Universidad de Cantabria
Departamento de Ingeniería Informática y Electrónica

Estudio de la planificabilidad y optimización de sistemas distribuidos de tiempo real basados en prioridades fijas y EDF

Tesis Doctoral
Juan María Rivas Concepción
Santander, mayo de 2015

Universidad de Cantabria
Departamento de Ingeniería Informática y Electrónica

Estudio de la planificabilidad y optimización de sistemas distribuidos de tiempo real basados en prioridades fijas y EDF

Memoria
Presentada para optar al grado de DOCTOR por la Universidad de Cantabria por

Juan María Rivas Concepción

Universidad de Cantabria
Departamento de Ingeniería Informática y Electrónica

Estudio de la planificabilidad y optimización de sistemas distribuidos de tiempo real basados en prioridades fijas y EDF

Memoria
presentada para optar al grado de Doctor por la Universidad de Cantabria, dentro del programa oficial de postgrado en Ciencias, Tecnología y Computación, por

Juan María Rivas Concepción

El Director:

Dr. J. Javier Gutiérrez García
Profesor Titular de Universidad

Declara:
Que el presente trabajo ha sido realizado en el Departamento de Electrónica y Computadores y en el Departamento de Ingeniería Informática y Electrónica de la Universidad de Cantabria, bajo su dirección, y reúne las condiciones exigidas para la defensa de tesis doctorales.

Santander, mayo de 2015

Fdo. Juan María Rivas Concepción

Fdo. J. Javier Gutiérrez García

Dicen que a veces te llegan momentos que te cambian la vida. El mío llegó en cuarto de carrera con la asignatura optativa Lenguajes de Alto Nivel, del profesor J. Javier Gutiérrez. Un año después estaba haciendo el proyecto fin de carrera con él, y desde entonces ha sido mi guía por el mundo de la investigación, culminando con esta tesis. Como no podía ser de otra manera, mi primer agradecimiento es a él, mi director, por toda la confianza que depositó en mi desde el principio, por toda su ayuda y paciencia durante todo este tiempo, sin la cual nada de esto hubiera sido posible.

Así mismo, también me gustaría expresar mi agradecimiento a todos los que con sus conocimientos y saber hacer me ayudaron a crecer en este mundo, Carlos, Michael, Julio, José María, Mario. No podría haber tenido mejores mentores. También reconocer a Jose Angel Herrero su paciencia y disponibilidad ante todas mis dudas y problemas que me surgían con el *cluster*.

A mi familia, por su paciencia y apoyo durante estos años. A mi madre que seguro que intenta leerse esta tesis entera; a mi hermano, que seguro que también la leerá, y probablemente la entenderá mejor que yo; y a mi padre, que aunque no pueda verlo, seguro que estaría orgulloso de mi.

No podía olvidarme de mis compañeros del despacho, pasillo, cafés y paellas, presentes y pasados. Cada uno dejáis una pequeña marca en mí para siempre. Son muchos los buenos momentos compartidos con vosotros, me habéis ayudado mucho a que todo este tiempo haya pasado volando.

Y también a mis amigos, demasiados como para citar aquí, no vaya a ser que se me olvide alguno, pero que todos y cada uno de vosotros sabéis quienes sois. Estáis desperdigados por todo el mundo, pero siempre estáis ahí cuando os necesito.

Gracias a todos.

Santoña, 13 de mayo de 2015.

Per aspera ad astra

Tabla de Contenidos

Resumen de la Tesis

Los sistemas distribuidos de tiempo real están adquiriendo una importancia creciente en aplicaciones actuales en campos con gran repercusión en la sociedad como la automatización industrial, la robótica, la automoción, o la aviónica. Estos sistemas ejecutan flujos de principio a fin distribuidos cuyo correcto funcionamiento depende del cumplimiento de ciertos requisitos temporales o plazos impuestos en el sistema.

El *software* de estos sistemas está formado por actividades que se ejecutan de forma concurrente. La decisión de qué actividad ejecutar en cada momento recae en un planificador que se rige por un política de planificación que determina las reglas en base a las que se elige la actividad a ejecutar. Las políticas principalmente utilizadas en los sistemas de tiempo real basan su decisión en una prioridad asociada a cada actividad que puede ser fija (FP, *Fixed Priorities*), o dinámica basada en el plazo (EDF, *Earliest Deadline First*). La política FP es la más utilizada y la más estudiada en la literatura. Sin embargo, EDF tiene la capacidad de ejecutar cargas mayores cumpliendo los plazos, y su soporte está creciendo en la industria. Para cada política de planificación se define un cuerpo teórico, del que se pueden extraer dos tipos de técnicas fundamentales:

- Técnicas de asignación de parámetros de planificación: se encargan de asignar a las actividades el parámetro que utiliza la política de planificación para tomar su decisión, p.ej. la prioridad fija en FP, o el plazo de planificación en EDF.
- Técnicas de análisis de planificabilidad: se encargan de determinar si un sistema concreto, con una asignación dada de parámetros de planificación, va a cumplir los requisitos temporales impuestos.

Ambos tipos de técnicas, y toda la teoría necesaria para definirlas, fueron inicialmente desarrolladas para sistemas con un único procesador. Durante los últimos 40 años el cuerpo teórico ha ido evolucionando para abarcar cada vez más tipos de sistemas. Existen en la actualidad varias técnicas que resuelven la asignación de parámetros de planificación y el análisis de planificabilidad en sistemas distribuidos, en los que las actividades intercambian mensajes a través de redes de comunicación de tiempo real que se planifican con políticas similares a las que utilizan los procesadores. Estas técnicas poseen diferentes niveles de complejidad y calidad en sus resultados.

El trabajo llevado a cabo en esta tesis se centra en realizar un estudio en profundidad de las soluciones que producen las técnicas de análisis de planificabilidad y asignación de parámetros de planificación para sistemas distribuidos, como consecuencia del cual se obtienen una serie de resultados que pueden guiar al diseñador a identificar qué algoritmo de planificación o técnica debe aplicar a su problema. También se adaptan las técnicas de análisis y optimización existentes a su aplicación a un conjunto más amplio de sistemas, que permiten la mezcla de diferentes planificadores en los recursos procesadores del sistema distribuido, y se proponen nuevas técnicas de asignación de plazos locales de planificación para EDF.

En estudios previos sobre técnicas de análisis de planificabilidad aplicables a sistemas distribuidos, se asume que todos los procesadores del sistema utilizan la misma política de planificación. Eliminar esta restricción abre la puerta a la utilización de la política de planificación más adecuada en cada recurso procesador del sistema distribuido. En el

contexto de esta tesis, a los sistemas distribuidos con diversidad de políticas de planificación se les llama sistemas distribuidos heterogéneos, y en la misma se proponen métodos para el análisis de planificabilidad y para la asignación de parámetros de planificación en estos sistemas heterogéneos.

La propuesta de una nueva técnica de análisis u optimización, suele ir acompañada de un estudio en el que se compara su rendimiento con otras técnicas existentes, normalmente para un conjunto de circunstancias reducido y dirigido a destacar las bondades de la nueva técnica. En esta tesis se plantea realizar un estudio masivo tanto de técnicas de análisis de planificabilidad, como de asignación de parámetros de planificación para sistemas distribuidos. Se ha desarrollado la herramienta GEN4MAST para la generación automática de sistemas de test que hace uso de un supercomputador, gracias al cual se pueden explorar un gran número de situaciones de estudio que no podrían ser tratadas con un computador convencional en tiempos de ejecución razonables.

En la realización de este estudio intensivo de técnicas se comprueba que, bajo ciertas circunstancias, la planificación EDF en sistemas distribuidos ofrece un rendimiento muy pobre que no parece corresponderse con este tipo de planificador. En esta tesis se estudian los motivos de este comportamiento anómalo y se ofrecen propuestas que, aun pareciendo contrarias a la intuición, aportan una mejora clara en los resultados.

Finalmente, se incluye un estudio del efecto del *jitter* (activación retrasada de las actividades) en la planificabilidad de los sistemas distribuidos. El *jitter* es un fenómeno inherente a las actividades en los sistemas distribuidos, y ya se habían estudiado previamente los beneficios que se podían obtener al eliminarlo en la planificación FP. En esta tesis se extiende este estudio a la planificación EDF.

Todas las técnicas de análisis de planificabilidad y de asignación de parámetros de planificación desarrolladas en esta tesis han sido implementadas en el entorno de herramientas MAST, que se distribuye como *software* libre bajo licencia GPL.

Abstract

Real-time distributed systems are starting to get more attention in the industry, and are being used in applications with high impact in daily life, such as factory automation, robotics, automotive or avionics. These systems execute distributed end to end flows whose correctness depends on meeting certain timing requirements or deadlines imposed on the system.

The system software is composed of tasks that are executed concurrently. The decision of which task should execute at any moment falls on a scheduler, which follows a scheduling policy that sets the rules used to select a task for execution. The most widely used scheduling policies are based on a priority assigned to each task, which can be fixed (Fixed Priorities scheduling, or FP), or dynamic based on the deadline (Earliest Deadline First scheduling, or EDF). FP is nowadays the most popular and best studied scheduling policy. However, EDF can reach higher loads in the system while meeting the deadlines, and its presence in the industry is currently increasing. For each scheduling policy, a series of techniques are defined, that can be divided into two fundamental groups:

- Scheduling parameters assignment techniques, that assign a scheduling parameter to each task, which is used by the scheduling policy to make its selection, e.g., a fixed priority for FP or a scheduling deadline for EDF.
- Schedulability analysis techniques, that determine whether a system, with a specific assignment of scheduling parameters, meets all of its deadlines or not.

Both kind of techniques, and their associated theoretical framework in which they are built upon, were initially developed for systems with only one processor. During the last 40 years, the research efforts have evolved the theoretical body to support a wider range of systems. Nowadays there are techniques available that perform the schedulability analysis and scheduling parameters assignment for distributed real-time systems, where tasks exchange messages transmitted through real-time networks that are scheduled with policies similar to those of the processors. These techniques offer different levels of complexity and quality in their results.

The work carried out in this thesis focuses on performing an exhaustive study of the solutions produced by both the schedulability analysis techniques, and the scheduling parameters assignment techniques, for distributed systems. The results obtained in this study can help a real-time system designer to identify the algorithm or scheduling policy to be applied. Additionally, the current analysis and optimization techniques are adapted for their application to a wider range of systems, where each processing resource could use a different scheduling policy. New techniques for the assignment of local scheduling deadlines for EDF are also proposed.

Previous studies on schedulability analysis techniques for distributed systems assumed that every processing resource was scheduled by the same scheduling policy. Eliminating this restriction enables using the most appropriate scheduling policy in each processing resource in the distributed system. In the context of this thesis, a distributed system with a variety of scheduling policies is called a heterogeneous distributed system, and methods to analyze it and assign scheduling parameters are proposed.

New analysis or optimization techniques usually include a study where their performance is compared to previous techniques, usually over a reduced set of attributes that tends to highlight the benefits of the new technique. In this thesis a wide study of the performance of schedulability analysis and scheduling parameters assignment techniques for distributed systems is carried out. For this purpose, the GEN4MAST tool has been developed, which performs the automatic generation of examples, and is able to leverage the computing power of a supercomputer. This capability enables the execution of broad studies that could not be carried out in reasonable amounts of time in a common PC.

The GEN4MAST tool has enabled performing a massive study which found that, under certain circumstances, the EDF scheduling policy shows a very poor performance in distributed systems, which is an unexpected behavior for this kind of scheduling policy. This anomalous performance is studied in this thesis, and new methods are proposed that, although seemingly contrary to intuition, can improve the performance significantly.

Finally, a study of the influence of the release jitter in the schedulability of distributed systems is included. Release jitter is a phenomenon inherent to tasks in distributed systems. The benefits of removing it in the context of the schedulability of FP systems had been previously studied. In this thesis this study is extended to EDF systems.

All the schedulability analysis and scheduling parameters assignment techniques proposed in the thesis have been implemented in the MAST suite of tools, which is distributed under the GPL license as free software.

Lista de Publicaciones

El trabajo de investigación incluido en esta tesis ha dado lugar a las siguientes publicaciones:

- J.M. Rivas, J.J. Gutiérrez and M. González Harbour, "GEN4MAST: A Tool for the Evaluation of Real- Time Techniques Using a Supercomputer," Proc. of the 3rd International Workshop on Real-Time and Distributed Computing in Emerging Applications (REACTION), págs. 41-47, Rome (Italy), 2014, ISBN 978-84-697-1736-3. (http://hdl.handle.net/10016/19692).
- J.M. Rivas, J.J. Gutiérrez, J.C. Palencia and M. González Harbour, "Deadline Assignment in EDF Schedulers for Real-Time Distributed Systems," In press, IEEE Transactions on Parallel and Distributed Systems, DOI: 10.1109/TPDS.2014.2359449. (http://dx.doi.org/10.1109/TPDS.2014.2359449).
- J.M. Rivas, J. J. Gutiérrez, M. González Harbour and J.C. Palencia, "Sobre la asignación de parámetros de planificación en sistemas distribuidos de tiempo real," IV Simposio de Sistemas de Tiempo Real in the IV Congreso Español de Informática (CEDI), Madrid (Spain), págs. 11-18, 2013, ISBN: 978-84-695-8334-0.
- M. González Harbour, J.J. Gutiérrez, J.L. Medina, J.C. Palencia, J.M. Drake, J.M. Rivas, P. López Martínez and C. Cuevas, "MAST: Bringing Response-Time Analysis into Real-Time Systems Engineering," Workshop on Real-time systems: the past, the present, and the future," York (UK), CreateSpace Independent Publishing Platform, págs. 42-59, 2013, ISBN:978-1482707786.
- J.M. Rivas, J.J. Gutiérrez and M. González Harbour, "Fixed Priorities or EDF for Distributed Real-Time Systems?," Proc. of the WiP Session of the 33rd IEEE Real-Time Systems Symposium (RTSS WiP), San Juan (Puerto Rico), 2012. In ACM SIGBED Review, vol. 10, no 2, págs. 21-21, 2012. (http://dx.doi.org/10.1145/2518148.2518159)
- J.M. Rivas, J.J. Gutiérrez, J.C. Palencia and M. González Harbour, "Schedulability analysis and optimization of heterogeneous EDF and FP distributed real-time systems," Proc. of the 23rd Euromicro Conference on Real-Time Systems (ECRTS), Porto (Portugal), págs. 195-204, 2011, ISBN 978-1-4577-0643-1. (http://dx.doi.org/10.1109/ECRTS.2011.26).
- J.M. Rivas, J.J. Gutiérrez, J.C. Palencia and M. González Harbour, "Optimized Deadline Assignment and Schedulability Analysis for Distributed Real-Time Systems with Local EDF Scheduling," Proc. of the 8th International Conference on Embedded Systems and Applications (ESA), Las Vegas (Nevada, USA), págs. 150-156, 2010, ISBN ISBN 1-60132-141-4.

Proyectos de Investigación Relacionados

La presente tesis ha sido desarrollada en el marco de los siguientes proyectos de investigación:

THREAD: Soporte integral para sistemas empotrados de tiempo real distribuidos y abiertos.
Proyecto de investigación coordinado del Plan Nacional financiado por el Ministerio de Educación y Ciencia.
Ref.: TIC2005-08665-C03-02.

RT-Model: Plataformas de tiempo real para diseño de sistemas empotrados basado en modelos.
Proyecto de investigación coordinado del Plan Nacional financiado por el Ministerio de Ciencia e Innovación.
Ref.: TIN2008-06766-C03-03.

HI-PartES: Sistemas empotrados particionados de alta integridad.
Proyecto de investigación coordinado del Plan Nacional financiado por el Ministerio de Ciencia e Innovación y los fondos FEDER.
Ref.: TIN2011-28567-C03-02.

Lista de Acrónimos

- DS: *Deadline Scaling.*
- EDF: *Earliest Deadline First.*
- ED: *Effective Deadline.*
- EQF: *Equal Flexibility.*
- EQS: *Equal Slack.*
- FP: *Fixed Priorities.*
- GC-EDF: *Global-Clock EDF.*
- GPL: (GNU) *General Public License*.
- GSD: *Global Scheduling Deadlines.*
- HOPA: *Heuristic Optimized Priority Assignment.*
- HOSDA: *Heuristic Optimized Scheduling Deadlines Assignment.*
- HOSPA: *Heuristic Optimized Scheduling Parameters Assignment.*
- LC-EDF: *Local-Clock EDF.*
- MARTE: *Modeling and Analysis of Real-Time and Embedded Systems.*
- MAST: *Modeling and Analysis Suite for Real-Time Applications.*
- NPD: *Normalized Proportional Deadlines.*
- OMG: *Object Management Group.*
- PD: *Proportional Deadlines.*
- UD: *Ultimate Deadlines.*
- UMP: Utilización Máxima Planificable.

Definiciones y Términos

- Términos del modelo MAST:

PR_i	Recurso procesador i-ésimo.
Γ_i	Flujo e2e i-ésimo.
N_i	Número de actividades en el flujo e2e Γ_i.
τ_{ij}	Actividad j-ésima del flujo e2e Γ_i.
e_i	Evento externo que activa el flujo e2e Γ_i.
P_{ij}	Prioridad fija de la actividad τ_{ij}.
SD_{ij}	Plazo global de planificación de la actividad τ_{ij}.
Sd_{ij}	Plazo local de planificación de la actividad τ_{ij}.
C_{ij} , Cb_{ij}	Tiempos de ejecución de peor y mejor caso de la actividad τ_{ij}, respecticamente.
R_{ij} , Rb_{ij}	Tiempos de respuesta de peor y mejor caso de la actividad τ_{ij}, respectivamente.
R_i , Rb_i	Tiempos de respuesta de peor y mejor caso del flujo e2e Γ_i, respectivamente.
J_{ij}	*Jitter* de activación de peor caso de la actividad τ_{ij}
D_i	Plazo de principio a fin del flujo e2e Γ_i.
D_{ij} , d_{ij}	Plazo global y local de la actividad τ_{ij}, respectivamente.
ϕ_{ij}	*Offset* de la actividad τ_{ij}.
U_k	Utilización del recurso procesador PR_k.

- Sistema Raíz: Sistema de test generado por la herramienta GEN4MAST en el que faltan por establecer los tiempos de ejecución de peor y mejor caso de las actividades.
- Serie de Utilizaciones: Serie de sistemas de test generados por la herramienta GEN4MAST a partir de un Sistema Raíz, en el que los tiempos de ejecución se aumentan de forma progresiva de forma que se abarque un rango de utilizaciones pre-establecido.
- Utilización Máxima Planificable: Utilización máxima que es planificable alcanzada en una Serie de Utilizaciones, tras aplicar una combinación de técnicas de análisis de planificabilidad y asignación de parámetros de planificación a los sistemas de la serie.

Lista de Figuras

1 Introducción

1.1 Sistemas de tiempo real

Los sistemas de tiempo real forman parte de nuestro día a día, aunque muchas veces no seamos conscientes de ello. Los podemos encontrar en aplicaciones tales como el control de la inyección o de estabilidad en los coches, en los sistemas de aviónica y control del tráfico aéreo, procesado digital de señal, bases de datos, o aplicaciones multimedia como compresión/descompresión de video.

En términos generales llamamos sistemas de tiempo real a aquellos sistemas computacionales en los que su correcto funcionamiento no depende sólo de la propia solución calculada, sino también del instante en el que ésta se obtiene [STA88]. En estos sistemas, tanto el obtener un resultado erróneo, como obtenerlo tarde, puede conllevar serias consecuencias.

Un ejemplo de la importancia del requisito temporal en los sistemas de tiempo real se puede encontrar en una fábrica en la que brazos robóticos son controlados automáticamente por computador. El computador debe ser consciente en todo momento del entorno y mover cada brazo acordemente. Si no consigue detener o mover adecuadamente a tiempo un brazo, podría colisionar con algún otro objeto en la planta, hecho que podría producir graves perjuicios materiales, e incluso personales.

No todas las aplicaciones de tiempo real poseen el mismo nivel de criticidad. Por ejemplo, en una aplicación multimedia, el no cumplimiento de los requisitos temporales podría implicar la pérdida de un fotograma en el video, hecho que, aparte de degradar la calidad del visionado, no produciría mayores repercusiones. En este tipo de sistemas de tiempo real, llamados de tiempo real no estricto, un retraso disminuye la calidad del resultado, pero no lo invalida por completo. El estudio de este tipo de sistemas se suele centrar en determinar estadísticamente la probabilidad de no cumplir los plazos.

En cambio, en un sistema como el del piloto automático de un avión, un retraso en el cómputo y aplicación de los resultados puede provocar un accidente, con posible pérdida de vidas humanas. A este conjunto de sistemas en los que un resultado retrasado se considera inválido se les conoce como de tiempo real estricto.

Los sistemas de tiempo real pueden adoptar distintas arquitecturas, aunque en general están formados por un determinado número de computadores gobernados por sistemas operativos de tiempo real, en los que se ejecutan las actividades requeridas, pudiendo hacer uso además de diferentes tipos de dispositivos de entrada/salida como sensores, servos, unidades de disco, redes de comunicaciones, etc.

Normalmente los requisitos temporales que se imponen a un sistema de tiempo real hacen referencia a la cantidad de tiempo que tienen para obtener una respuesta correcta, o también llamado tiempo de respuesta. A este requisito se le llama plazo.

El poder determinar que un sistema vaya a cumplir todos sus plazos siempre es de vital importancia en el campo de los sistemas de tiempo real estricto. Existen principalmente dos tipos de técnicas para garantizar el cumplimiento de los plazos:

- Técnicas *off-line*: Antes de poner en funcionamiento el sistema, se conocen todas sus características relevantes como su carga computacional o el comportamiento temporal. Con toda esta información se utiliza alguna técnica matemática para determinar cuál es el peor caso que puede darse en la ejecución, y si en dicho caso se cumplirían los plazos. Si en ese peor caso el sistema cumple sus plazos, se dice que el sistema es planificable.
- Técnicas *on-line*: La carga computacional no se conoce de antemano, sino que se va añadiendo en tiempo de ejecución. Antes de añadir una actividad nueva, se realiza un análisis que puede ser similar al realizado en las técnicas *off-line*, considerando que la nueva actividad se está ejecutando en concurrencia con las pre-existentes. Si se determina que el nuevo sistema sigue siendo planificable, se acepta la nueva actividad.

Esta memoria tratará únicamente sistemas en los que determinamos la planificabilidad de manera *off-line*, y por lo tanto, tendremos de antemano un conocimiento absoluto de las características de los sistemas que trataremos.

Desde el punto de vista del número de procesadores y la arquitectura en la que se interconectan, distinguimos tres tipos de sistemas de tiempo real:

- Sistemas monoprocesadores, en los que sólo existe un único procesador en el que se ejecutan todas las actividades, típicamente de forma independiente, salvo en el acceso a recursos compartidos.
- Sistemas multiprocesadores, compuestos por varios procesadores estrechamente conectados, de forma que se puede establecer un estado global en el sistema de una manera no costosa. La comunicación entre las actividades situadas en diferentes procesadores típicamente se lleva a cabo mediante mecanismos como memoria compartida o buses de datos.
- Sistemas distribuidos, en los que diferentes computadores se interconectan a través de una o varias redes de comunicación para perseguir una finalidad común. La principal diferencia con los sistemas multiprocesadores es que la conexión entre los diferentes procesadores se realiza de manera menos estrecha, por lo que resulta más costoso mantener actualizado un estado global en el sistema. La comunicación entre actividades situadas en distintos computadores se realiza

mediante el envío de mensajes transmitidos a través de las redes de comunicación que los unen.

1.2 Sistemas distribuidos de tiempo real

Durante las últimas décadas el coste de los microprocesadores y de las redes de comunicación ha ido disminuyendo de manera constante, hecho que unido a la cada vez mayor demanda computacional a la que se debe atender, está creando una tendencia hacia sistemas de computación distribuidos. Las principales ventajas de este tipo de esquemas de computación pueden resumirse en los siguientes puntos: [BUR09b]

- Posibilidad de obtener mayor rendimiento por la paralelización del trabajo.
- Mayor fiabilidad usando redundancia en los elementos del sistema.
- Localización de la capacidad de cómputo allá donde se utilice.
- Oportunidad de mejora del rendimiento aumentando el número de recursos de computación, o de los elementos de comunicación.

Sistemas distribuidos sin carácter de tiempo real los encontramos en *clusters* o *grids* de computación, las redes de los cajeros automáticos, o bases de datos distribuidas. Un ejemplo de las claras ventajas de la computación distribuida se observa en los llamados proyectos de computación distribuida participativa, en los cuales cualquier usuario puede añadir su propio PC a una red internacional. Un caso de este tipo de proyecto es *Folding@Home* [FOLAH], que mediante este esquema distribuido participativo logra crear un sistema con una capacidad total de cómputo de más de 40 petaFLOPS para llevar a cabo simulaciones biomédicas.

En el ámbito de los sistemas de tiempo real, los sistemas distribuidos gozan de gran aceptación en diversas aplicaciones. Un claro ejemplo lo encontramos en el mundo de la aviónica. Los aviones poseen multitud de computadores para llevar a cabo funciones como el control de vuelo, piloto automático, monitorización o alerta de colisiones. Estos computadores hacen uso de la información que proviene de diversas fuentes, como sensores de velocidad del aire, presión atmosférica, o temperatura, y a su vez aplican sus resultados en otros sistemas como los actuadores de los alerones. Computadores, sensores y actuadores se interconectan mediante una red para formar un sistema distribuido cuya complejidad es cada vez más elevada. Esta tendencia hacia sistemas más complejos se observa en la evolución de las redes de comunicación usadas en los sistemas de aviónica. El estándar ARINC 429 de 1977 [SPI00], usado por ejemplo en el Boeing 767 permite comunicaciones unidireccionales de hasta 100 kb/s en su modo más rápido. En 1994, el estándar ARINC 629 [ARI629] usado casi en exclusiva en el Boeing 777, define un bus bidireccional con transferencias de hasta 2 Mb/s. En la actualidad el estándar ARINC 664 Part 7 [AFDX], también conocido como AFDX, define una red Ethernet determinista que permite topologías más complejas y mayor ancho de banda, y es usado entre otros en el Airbus A350 y el Boeing 787.

Otra aplicación de los sistemas distribuidos de tiempo real con gran relevancia en la industria la podemos encontrar en el sector de la automoción. La demanda de mayor confort, eficiencia energética y seguridad hace que cualquier vehículo moderno posea decenas de procesadores que desempeñan funciones tales como el control de estabilidad o la administración de la inyección en el motor. Para dar solución al problema de

interconectar todos estos sistemas de una manera eficiente y con coste bajo, Bosch diseñó el bus CAN [CAN], convertido en el estándar *de facto* en la industria durante los últimos 20 años. Por otra parte, el bus Flex Ray [FLEXR] fue ideado por un consorcio liderado por BMW como sustituto del bus CAN, con el objetivo de dar soporte a las mayores transferencias de datos necesarias en las funciones electrónicas modernas. Pese a su bondades, su mayor complejidad y coste en comparación al probado bus CAN, unido a su reciente creación, provoca que el uso de Flex Ray se limite en la actualidad a vehículos de gama alta.

Existen aplicaciones emergentes que potencialmente pueden entenderse como un sistema distribuido, como por ejemplo los sistemas *many-core.* La cada vez mayor miniaturización utilizada en la fabricación de circuitos integrados está dando lugar a la aparición de procesadores cuya arquitectura integra decenas o incluso centenares de unidades de proceso *(cores)*. Para aprovechar la capacidad de un sistema *many-core*, el *software* que se ejecuta debe estar compuesto por multitud de actividades ejecutando concurrentemente. La comunicación entre las actividades se realiza normalmente mediante mecanismos de paso de mensajes, por lo que su concepción se asemeja a lo que entendemos por un sistema distribuido convencional.

1.3 Planificación en sistemas de tiempo real

Desde el punto de vista computacional, el trabajo que llevan a cabo los sistemas de tiempo real se divide en unidades que se ejecutan de forma concurrente a las que llamamos actividades. Una actividad puede consistir en una serie de instrucciones ejecutando en un procesador, un mensaje transmitido por una red de comunicación, la adquisición de un dato en un sensor, etc. Independientemente del tipo de trabajo que realice, decimos que la actividad se ejecuta en un recurso procesador (procesador, red, etc.).

Las actividades concurrentes del sistema se ejecutan en los recursos procesadores de acuerdo al orden establecido por algún algoritmo que se implementa en un módulo llamado planificador. El planificador por lo tanto se encarga de decidir en cada momento qué actividades deben ejecutarse en cada recurso procesador, de acuerdo a un criterio que se conoce como política de planificación.

Existen principalmente dos paradigmas que definen el funcionamiento básico del planificador: sistemas gobernados por tiempo [KOP98][KOP11], y sistemas gobernados por eventos [LIJ00][BUT11][KLE93].

1.3.1 Sistemas gobernados por tiempo

En el paradigma de los sistemas gobernados por tiempo, las actividades se ejecutan en instantes de tiempo predefinidos. Generalmente la planificación se lleva a cabo mediante una tabla en la que se especifican los instantes de tiempo en los que se deben ejecutar cada una de las actividades del sistema. Esta tabla se construye antes de poner en funcionamiento el sistema, y de forma que todas las actividades cumplan sus plazos (si tal solución existe). Debido a que el orden en el que todas las actividades van a ser

ejecutadas se decide de antemano, éste es un paradigma de planificación *off-line* o estático (se planifica en tiempo de compilación del sistema).

La principal ventaja de los sistemas gobernados por tiempo es que poseen un comportamiento único y completamente determinista, definido en su totalidad en la tabla de planificación. Por lo tanto, la determinación del cumplimiento de todos los plazos viene implícita en la planificación. Otra ventaja de este paradigma es que una vez que el sistema entra en funcionamiento, la política de planificación es trivial y de complejidad reducida, ya que consiste en simplemente un despachador de actividades que realiza una búsqueda en la tabla de planificación. Por lo tanto, su coste computacional en tiempo de ejecución es reducido.

Un caso sencillo y extensamente utilizado de sistema gobernado por tiempos es el de los ejecutivos cíclicos. En estos sistemas se define una única actividad periódica en la que se ejecutan de manera secuencial todas las acciones que se desean realizar.

Mientras que los sistemas gobernados por tiempo aportan una solución válida para entornos deterministas, existen multitud de situaciones en el que tal aproximación resulta demasiado restrictiva e inflexible, al no poder adaptarse con eficacia ante posibles evoluciones del entorno, o posteriores revisiones de los programas. Para tales situaciones, los sistemas gobernados por eventos pueden resultar más apropiados.

1.3.2 Sistemas gobernados por eventos

En los sistemas gobernados por eventos, las actividades se ejecutan como respuestas a eventos. La llegada de un evento provoca la activación de una (o varias) actividades, que se dice que han sido activadas. Ninguna actividad puede ejecutarse antes de activarse. El planificador decide en tiempo de ejecución qué actividad, de entre todas las activas, debe ejecutarse en cada momento. Este esquema de ejecución se corresponde por lo tanto con un tipo de planificación *on-line* (se planifica en tiempo de ejecución del sistema).

Los eventos representan estímulos en el sistema para los cuales se requiere una respuesta. El origen de los eventos puede ser periódico (p.ej. un reloj), o no periódicos (p.ej. un operario activando un botón, o una interrupción en un sensor por la captura de nuevos datos).

En los sistemas gobernados por eventos, el mecanismo más utilizado en la planificación de las actividades del sistema es el establecimiento del orden de ejecución de acuerdo a un valor que denota la prioridad de cada actividad. En los momentos en los que el planificador actúa (instantes de decisión), éste selecciona para ser ejecutada a la actividad con mayor prioridad de entre todas las actividades activas.

En función de los instantes en los que decidan, los planificadores se pueden clasificar en dos tipos distintos: planificadores expulsores o no expulsores [LIJ00]. Cuando es no expulsor, el momento de la decisión de planificación ocurre cuando las actividades terminan su ejecución. Por lo tanto, una vez que una actividad comience a ejecutar ésta no será interrumpida hasta haber finalizado, aunque haya actividades de mayor prioridad esperando. Esto puede provocar un retraso innecesario en las actividades de mayor prioridad.

En los planificadores expulsores, el momento de la decisión de planificación ocurre cada vez que una actividad se activa, finaliza su ejecución, o a intervalos regulares

gobernados por un reloj (*ticker*). Con este tipo de planificación nos aseguramos de que, salvo en situaciones en las que alguna actividad no pueda ser expulsada, en todo momento se esté ejecutando la actividad de mayor prioridad, de entre todas las actividades activas.

Al lapso de tiempo entre la llegada del evento y la finalización de la ejecución de su actividad asociada se le conoce como el tiempo de respuesta de dicha actividad. Este tiempo de respuesta está compuesto por el propio tiempo de ejecución de la actividad, y cualquier otra circunstancia que le impida ejecutarse (p.ej. la ejecución de una actividad de prioridad superior, o el bloqueo en el acceso a un recurso compartido).

El orden concreto en el que se van a ejecutar las actividades, y por lo tanto sus tiempos de respuesta, se desconoce antes de poner en funcionamiento el sistema. Así, salvo para sistemas muy sencillos, es imposible determinar los tiempos de respuesta de las actividades en el peor caso mediante la prueba del sistema. Para garantizar el cumplimiento de todos los plazos del sistema existen técnicas analíticas que se basan en las utilizaciones del sistema o en el cálculo de cotas superiores de los tiempos de respuesta de las actividades. Estás técnicas, que normalmente se aplican *off-line*, toman como entrada las características temporales relevantes del sistema (tiempos de ejecución, características de activación de los eventos, etc.), y para una política de planificación dada, determinan si los plazos van a poder cumplirse o no. La aplicación de estas técnicas analíticas requiere un modelado previo del sistema en el que se capturan todos los parámetros necesarios.

Si el análisis de planificabilidad dictamina que un sistema es planificable, significa que existe garantía de que todas las actividades en el peor de los casos van a tener tiempos de respuesta inferiores a los plazos impuestos.

La principal ventaja de los sistemas gobernados por eventos es su mayor flexibilidad. En los sistemas gobernados por tiempo, cualquier cambio requiere la reconstrucción de la tabla de planificación, hecho que puede complicar el mantenimiento de los sistemas más complejos [SHA04]. En cambio, en un sistema basado en prioridades, cualquier cambio implica únicamente llevar a cabo un análisis de planificabilidad para determinar si el sistema sigue siendo planificable. La principal desventaja de los sistemas gobernados por eventos radica en la mayor complejidad de implementar un planificador *on-line* en comparación con un simple despachador de actividades utilizado en los sistemas gobernados por tiempo.

Por último, en los sistemas de tiempo real las actividades se pueden sincronizar en el acceso a recursos compartidos (p.ej., datos o dispositivos de entrada salida). El acceso a estos recursos está normalmente protegido por mecanismos que garantizan la exclusión mutua, y con objeto de evitar inversiones de prioridad no acotadas se deben utilizar protocolos de sincronización de tiempo real. Estos protocolos permiten acotar los tiempos de bloqueo de las actividades, mejorando la planificabilidad de los sistemas.

Los bloqueos por acceso a recursos compartidos son situaciones que sólo conciernen a los sistemas gobernados por eventos, ya que los sistemas gobernados por tiempo se suelen construir de forma que se hace un acceso serializado de los recursos compartidos, evitando así las situaciones en las que una actividad permanezca bloqueada.

A lo largo de esta breve introducción a los sistemas gobernados por eventos se han identificado tres características fundamentales de éstos:

- La planificación normalmente se realiza *on-line* utilizando una **política de planificación basada en prioridades**.
- La planificabilidad se determina mediante técnicas analíticas (**análisis de planificabilidad**).
- El análisis de planificabilidad necesita un **modelo del sistema** que capture las características que influirán en el comportamiento temporal del mismo.

En las próximas secciones de este capítulo introductorio se ahondará en estos tres conceptos que resultan fundamentales en el desarrollo de esta tesis. Se hará un especial hincapié en su aplicación en sistemas distribuidos. En la Sección 1.4 se tratará el problema de la captura de las características de un sistema en un modelo, en la Sección 1.5 se tratarán las políticas de planificación más utilizadas, y en las Secciones 1.6 y 1.7 se introducirán los conceptos básicos sobre el análisis de planificabilidad.

1.4 Modelo de sistema distribuido: MAST

El modelo de un sistema de tiempo real es una estructura de datos abstracta en la que se almacena información relevante que lo describe, de forma que pueda ser procesado por una herramienta *software*, como por ejemplo una herramienta de análisis de planificabilidad, una herramienta que optimice las características del sistema, o un simulador que muestre el comportamiento del sistema en función del tiempo.

Desde un punto de vista más general, los modelos forman el eje central de la metodología de desarrollo de sistemas MDE (*Model Driven Engineering*) [SCH06]. Esta metodología se basa en la formalización en modelos de toda la información que se utilice en el proceso de desarrollo de un sistema. El procesado de la información se realiza a través de transformaciones entre modelos, que también se formalizan en modelos. Un entorno MDE aporta claras ventajas en el desarrollo de sistemas de tiempo real [CUE13][TER06], que se pueden resumir en los siguientes puntos:

- La abstracción del sistema en un modelo permite al desarrollador centrarse en la información relevante, con independencia de la plataforma utilizada.
- Se pueden especificar diferentes modelos que describan con mayor precisión diferentes aspectos de un mismo sistema.
- Permite la aceleración del proceso de diseño de sistemas de tiempo real.

Con objeto de poder desarrollar sistemas de tiempo real en un entorno MDE, la OMG (*Object Management Group*) definió el estándar MARTE (*Modeling and Analysis of Real Time and Embedded systems*) [MARTE]. Este estándar reúne todos los conceptos necesarios para poder modelar un sistema de tiempo real, y poder aplicar las técnicas más utilizadas en ellos, como el análisis de planificabilidad.

Con unos objetivos parejos a los perseguidos por MARTE, MAST [MAST] [MAST2] define un modelo con el que capturar el comportamiento temporal de sistemas de tiempo real. Además, aporta una serie de herramientas de *software* libre con las que aplicar diferentes técnicas a los sistemas modelados con MAST, como el análisis de planificabilidad, la optimización de sus parámetros de planificación, o la simulación de sus respuestas temporales.

El modelo MAST está alineado con el estándar MARTE, y en su versión 2 (MAST-2) [MAST2] se concibe para poder ser utilizada dentro de un entorno MDE. Adicionalmente, suministra herramientas que permiten transformaciones de modelos entre MAST y MARTE, para una mayor interoperabilidad. A continuación se resumen los principales elementos de modelado disponibles en MAST, junto con la nomenclatura que se utilizará en esta tesis.

El modelo MAST concibe los sistemas de tiempo real como sistemas reactivos, en los que el *software* reacciona ante eventos (internos o externos) ejecutando alguna actividad, denotada por el símbolo τ.

Las actividades pueden representar segmentos de código ejecutándose en un procesador, o mensajes transmitidos por una red de comunicación. Tanto a los procesadores como a las redes los denominamos recursos procesadores (*PR*). Independientemente de si representa código ejecutable, o un mensaje, diremos que las actividades de ejecutan en los recursos procesadores. Las actividades se localizan de manera estática, esto es, siempre se ejecutarán en el mismo recurso procesador (no hay migración de código). Asumimos que cada recurso procesador posee un planificador. En el ámbito de los sistemas distribuidos, a los recursos procesadores también se les denomina nodos.

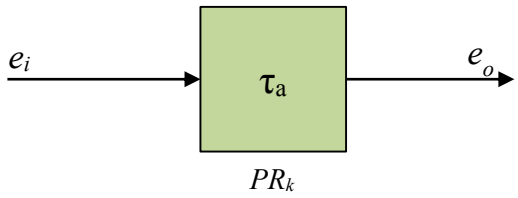


Figura 1-1 Modelo de activación de una actividad en MAST.

Una actividad se dice que se activa cuando se pone a disposición del planificador para ser ejecutada. En el modelo MAST las actividades se activan mediante la llegada de un evento de entrada. Cuando finaliza su ejecución, la actividad dispara un evento de salida. En la Figura 1-1 se representa la activación de una actividad (τ_a) mediante la llegada de un evento de entrada e_i. Cuando finaliza su ejecución, la actividad lanza el evento de salida e_o. La actividad se ejecuta en el recurso procesador PR_k.

Las aplicaciones distribuidas suelen estar compuestas por una serie de actividades que pueden ser independientes, o también estar relacionadas de manera que se van ejecutando secuencialmente en los recursos procesadores del sistema. Así por ejemplo, al sistema puede llegar un evento externo que activa una actividad en un nodo procesador; cuando ésta finaliza su ejecución puede enviar un mensaje a través de una red de comunicaciones, y éste activaría otra actividad en otro nodo procesador, y así sucesivamente. En el modelo MAST este funcionamiento se modela agrupando estas actividades en una entidad que llamamos flujo de principio a fin, o flujo e2e (*end-to-end flow*). Definimos el flujo *i*-ésimo Γ_i como la secuencia de actividades (τ_{i1}, τ_{i2},…, τ_{iNi}) que se van ejecutando sucesivamente en los diferentes recursos procesadores en los que se localizan, siendo N_i el número de actividades contenidas en el flujo Γ_i. Estas actividades representan las secuencias de código y mensajes que se van sucediendo en el sistema como respuesta al evento externo que llegó al sistema. Al flujo e2e que sólo tiene una actividad le llamamos flujo e2e simple.

Las diferentes actividades de un flujo e2e se van enlazando mediante eventos internos. Así, la primera actividad del flujo Γi (τ_{i1}) se activa mediante la llegada de un evento externo (e_i). Cuando esta primera actividad finaliza su ejecución, activa un evento interno que sirve a su vez de activación a la siguiente actividad del flujo. El proceso se repite sucesivamente hasta que finaliza la última actividad del flujo e2e.

La carga del sistema se modela mediante la especificación de los eventos externos que inician los flujos e2e. El modelo permite definir eventos externos periódicos o esporádicos que representan una carga acotada en el sistema, bien por el período o bien por una cota inferior al ritmo de llegadas. En caso de una actividad periódica τ_{ij} , su periodo se denota con T_i. Si fuera una actividad esporádica, T_i especifica el tiempo mínimo entre llegadas del evento.

En los flujos e2e de MAST, los requisitos temporales se pueden especificar de tres maneras distintas:

- Plazo de principio a fin (D_i), que representa el plazo de tiempo que tiene para ejecutarse el flujo Γ_i completo, desde la llegada del evento externo e_i, hasta la finalización de su última actividad τ_{iNi}.
- Plazo local (d_{ij}), que representa el plazo de tiempo que tiene para ejecutarse la actividad τ_{ij} desde su propia activación.
- Plazo global (D_{ij}), que representa el plazo de tiempo que tiene para ejecutarse la actividad τ_{ij}, contado desde la llegada del evento e_i que activó el flujo e2e en el que reside.

En la Figura 1-2 se representa un ejemplo sencillo de un flujo e2e (Γ_1) con tres actividades (τ_{11}, τ_{12}, τ_{13}), situadas en los recursos procesadores (PR_1, PR_2, PR_3) respectivamente. Se muestran además los tres tipos de requisitos temporales descritos previamente.

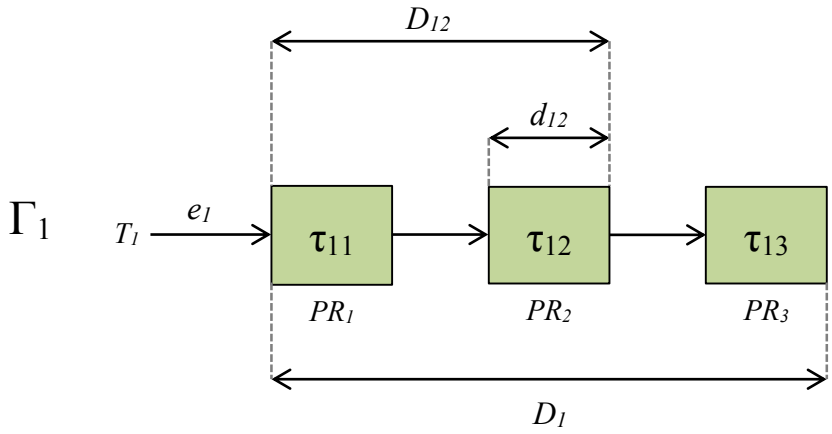


Figura 1-2 Ejemplo de flujo e2e con 3 actividades ejecutando en 3 recursos procesadores

Cada actividad τ_{ij} tiene asociados unos tiempos de ejecución de peor y mejor caso (C_{ij} *y* Cb_{ij} respectivamente). Estos valores especifican respectivamente el peor y mejor tiempo de ejecución que requiere la actividad, si esta se ejecutara sin ningún tipo de interrupción o interferencia externa, en un recurso procesador dado. Por lo tanto, el valor de este parámetro depende únicamente de la complejidad de la propia actividad, y de la velocidad del recurso procesador sobre el que se ejecuta.

Definimos el tiempo de respuesta global de una actividad τ_{ij} como el tiempo transcurrido desde la llegada del evento externo e_i hasta que τ_{ij} finaliza su ejecución. Cada activación de τ_{ij} puede tener un tiempo de respuesta distinto en función del estado del sistema. Definimos el tiempo de respuesta global de peor caso de τ_{ij} como una cota superior de sus posibles tiempos de respuesta, y lo denominaos R_{ij}. En esta tesis no se van a considerar tiempos de respuesta locales referidos a eventos internos, por lo que en adelante todas las referencias a tiempos de respuesta serán globales.

El tiempo de respuesta de peor caso del flujo Γ_i se corresponderá con el de su última actividad, R_{iNi}, aunque para simplificar lo denotaremos como R_i. Se dice que un sistema es planificable si cumple:

$$R_i \leq D_i \qquad \forall \Gamma_i \mid \exists D_i \tag{1-1}$$

$$R_{ij} \leq D_{ij} \qquad \forall \tau_{ij} \mid \exists D_{ij} \tag{1-2}$$

En esta definición de planificabilidad del sistema sólo se tienen en cuenta los plazos globales, que son los que se van a poder comprobar con los tiempos de respuesta que se obtengan.

De igual manera se define el tiempo de respuesta de mejor caso Rb_{ij} como una cota inferior del tiempo de respuesta de la actividad τ_{ij}.

El efecto de utilizar recursos compartidos se modela mediante los tiempos de bloqueo. Definimos B_{ij} como el tiempo de bloqueo de peor caso que puede experimentar una actividad τ_{ij} por el acceso a recursos compartidos. Este tiempo de bloqueo depende del protocolo de sincronización utilizado.

Adicionalmente, el modelo MAST soporta la especificación de *offsets* en las actividades. Un *offset* ϕ_{ij} especifica que la actividad τ_{ij} debe activarse como mínimo ϕ_{ij} unidades de tiempo después de la llegada del evento externo e_i.

La posible variabilidad en la activación de una actividad se conoce como *jitter* de activación. Definimos J_{ij}, como la mayor variación posible en el instante de activación de τ_{ij}. En un flujo e2e todas las actividades después de la primera (τ_{i1}) pueden tener un *jitter* de activación distinto de cero, debido a la variación en la finalización de la actividad anterior.

Cabe notar que en el modelo MAST los parámetros temporales del sistema no se especifican en una unidad de tiempo concreta. Se asume que todos los valores referencian la misma unidad de tiempo.

En el contexto del modelo MAST se definen una serie de conceptos que se utilizarán a lo largo de este trabajo, por lo que se detallan a continuación.

La utilización de un recurso procesador se define como la suma de las utilizaciones de las diferentes actividades alojadas en dicho recurso (C_{ij}/T_i). Así, llamamos U_k a la utilización del recurso procesador PR_k y se calcula con la siguiente ecuación:

$$U_k = \sum_{\forall \tau_{ij} \in PR_k} \frac{C_{ij}}{T_i} \tag{1-3}$$

Definimos la utilización de un sistema distribuido como la media de las utilizaciones de los recursos procesadores que lo componen. La utilización de un sistema con M recursos procesadores se define por lo tanto con la siguiente ecuación:

$$U = \frac{1}{M}\sum_{\forall k} U_k \tag{1-4}$$

Otro concepto importante es el *slack* del sistema. El *slack* de un sistema planificable se define como el porcentaje máximo por el cual se pueden aumentar los tiempos de ejecución de peor caso (C_{ij}) de todas las actividades del sistema de forma que se mantenga la planificabilidad. Por otra parte, el *slack* de un sistema no planificable se define como el porcentaje por el cual se deben reducir los tiempos de ejecución de peor caso de todas las actividades, de forma que se alcance la planificabilidad. El *slack* es positivo si el sistema es planificable y negativo si no lo es. Adicionalmente, en MAST se definen otros tipos de *slack*, que afectan parcialmente a un recurso procesador, un flujo e2e, o a una sola actividad.

El modelo que se utiliza en esta tesis representa un subconjunto del modelo MAST, cuyo modelo completo soporta situaciones adicionales que no serán objeto de este trabajo, tales como la definición de múltiples eventos mediante los cuales las actividades pueden lanzar más de un evento de salida, o que una actividad necesite más de un evento para activarse.

1.5 Políticas de planificación basadas en prioridades

La política de planificación es el algoritmo que utiliza el planificador para elegir la actividad que se debe ejecutar de entre aquellas que están listas para ejecutar. En un política basada en prioridades, el planificador elige para la ejecución a la actividad con la mayor prioridad.

Las políticas de planificación basadas en prioridades pueden categorizarse en dos tipos dependiendo del significado que se le da al valor de la prioridad: prioridades fijas, y prioridades dinámicas. A continuación se introducirán ambos tipos de políticas.

1.5.1 Planificación por prioridades fijas

En una política basada en prioridades fijas el valor de la prioridad no varía con la ejecución de las actividades del sistema. Normalmente se establece el valor de la prioridad de cada actividad del sistema antes de la puesta en funcionamiento del mismo. A la prioridad fija de la actividad τ_{ij} se le denota como P_{ij}. Por simplicidad, a los sistemas gobernados por planificadores expulsores de prioridades fijas los llamaremos simplemente sistemas de prioridades fijas, o FP (del inglés *Fixed Priorities*).

Una forma sencilla de implementar un planificador FP consiste en crear una cola de actividades listas para ser ejecutadas, en el que las actividades quedan ordenadas por su prioridad. Cuando se produce un instante de decisión, el planificador selecciona para ser ejecutada a la primera actividad de la cola (actividad de mayor prioridad). Cada vez que una nueva actividad pasa a estar lista para ser ejecutada, o haya sido expulsada antes de finalizar su ejecución, el planificador la introduce en la cola de actividades listas en la posición correspondiente a su prioridad.

La planificación FP goza de una gran aceptación en la industria, siendo la política de planificación más usada en sistemas planificados por prioridades. Está incluido en el estándar POSIX [POSIX], que es implementado en sistemas operativos de tiempo real como VxWorks [VXWRK], MaRTE [ALD01], ShARK [SHARK] o RTLINUX [RTLIN]. Otros sistemas operativos que también soportan FP son Windows CE [WINCE] o FreeRTOS [FRTOS].

La política FP también es soportada en lenguajes de programación como Ada [TUC12], o la especificación de tiempo real del lenguaje Java (RTSJ) [RTSJ][JAVRTS][PERC][JAMVM]. Además, ha sido utilizada en redes de comunicación como CAN [CAN], o redes de tipo Ethernet como AFDX [AFDX].

Un caso especial que debe tratar el planificador FP es cómo actuar cuando exista más de una actividad con la misma prioridad. Para estas situaciones, POSIX aporta dos soluciones:

- SCHED_FIFO: De entre todas las actividades con la misma prioridad, se ejecuta primero la que se activó antes. Hasta que no finaliza, no se ejecuta la siguiente actividad con la que comparte prioridad.
- SCHED_RR: Se lleva a cabo una planificación *Round Robin* [LIJ00] entre las actividades con la misma prioridad. La planificación *Round Robin* consiste en ir asignando alternativamente rodajas de tiempo a la ejecución de cada actividad afectada, hasta que todas hayan finalizado.

1.5.2 Planificación por prioridades dinámicas

En oposición a FP, las prioridades de las actividades también pueden plantearse de forma que posean un carácter dinámico, esto es, que el valor de las prioridades varíe a lo largo de la ejecución del sistema. Liu [LIJ00] identifica dos políticas de planificación de prioridades dinámicas principales para sistemas de tiempo real: EDF (*Earliest Deadline First*) [LIU73], y LST (*Least Slack Time*) [MOK83].

En la política EDF, las prioridades se asignan de forma inversamente proporcional a los plazos absolutos de las activaciones de las actividades. Así, se define td_{ij}^{k} como el plazo absoluto de la *k*-ésima activación de la actividad τ_{ij}, según la siguiente ecuación:

$$td_{ij}^{k} = t_{ij}^{k} + D_{ij} \tag{1-5}$$

donde t_{ij}^{k} es el instante de la *k*-ésima activación de τ_{ij}.

En cada momento de decisión, el planificador elige para ejecutar a la actividad que posee el plazo absoluto más cercano. Por simplicidad, y de manera pareja a lo descrito para sistemas FP, a los sistemas expulsores planificados por prioridades dinámicas EDF los llamaremos simplemente sistemas EDF.

Un planificador EDF puede implementarse de manera similar a FP, esto es, creando una cola de actividades pendientes de ejecución, en la que en esta ocasión las actividades se ordenan en función de su plazo absoluto.

La política EDF es la más utilizada entre las políticas de prioridades dinámicas, aunque su aceptación aún dista de la observada para sistemas FP. Sin embargo, la

planificación EDF posee una serie de propiedades referentes a su mayor capacidad de planificación respecto a FP que le hacen especialmente atractivo, y que serán tratadas en la sección sobre el análisis de planificabilidad de sistemas EDF (ver Sección 1.6.2).

En la actualidad podemos encontrar implementaciones de planificadores EDF para tiempo real en lenguajes de programación como Ada [TUC12] o la especificación de tiempo real de Java [RTSJ]; en protocolos de comunicación sobre bus CAN [PED02], o redes de propósito general [DIN01]; en sistemas operativos de tiempo real tales como SHaRK [SHARK], ERIKA [ERIKA], MaRTE [ALD01], OSEK/VDX (a nivel de aplicación) [DIE08]; en otros sistemas operativos de propósito general como Linux desde la versión 3.14 con el planificador SCHED_DEADLINE [FAG09], o también en modificaciones de dicho sistema operativo como AQuoSA [PAP08].

La otra política de planificación con prioridades dinámicas destacada es LST, también conocida como LLF (*Least Laxity First*). Con LST la actividad con mayor prioridad es aquella con menos *slack*, entendiendo el *slack* en este caso como lo definido en la siguiente ecuación:

$$Slack(t)_{ij}^{k} = td_{ij}^{k} - t - C'_{ij}(t) \tag{1-6}$$

donde td_{ij}^{k} es el plazo absoluto de la *k*-ésima activación de τ_{ij}, *t* es el instante de tiempo actual, y $C'_{ij}(t)$ es la cantidad de tiempo de ejecución restante en la actividad. Por lo tanto, la prioridad puede cambiar tanto entre diferentes activaciones, como incluso dentro de una misma activación.

1.5.3 Planificación en sistemas distribuidos

A la hora de planificar sistemas con más de un recurso procesador, se plantean una serie de particularidades que necesitan ser tenidas en cuenta.

En los sistemas multiprocesadores, en los que se considera que los diferentes procesadores están estrechamente conectados, se presentan dos tipos de planificación en función de cómo se localicen las actividades en los diferentes procesadores [DAV11]:

- Planificación particionada: las actividades se asignan estáticamente a un procesador, como por ejemplo en [BAR08].
- Planificación global: las actividades pueden migrar de un procesador a otro en el transcurso de la ejecución, como por ejemplo con [BER09].

La principal ventaja de la planificación particionada es su menor complejidad, y la ausencia de sobrecargas añadidas por las migraciones de las actividades. Además, asignar estáticamente actividades a procesadores permite la utilización de la extensa teoría sobre sistemas monoprocesadores. En contraposición, permitir migrar de procesador a las actividades permite un mejor aprovechamiento de los recursos del sistema, y facilita la adaptación del mismo ante cambios en la carga computacional.

La posibilidad de poder realizar una planificación global en sistemas multiprocesadores surge del fuerte acoplamiento entre los distintos procesadores, lo que permite que el mantenimiento de un estado global del sistema pueda llevarse a cabo con bajo coste computacional. Sin embargo, los sistemas distribuidos carecen de esta estrecha

interconexión entre los procesadores, por lo que en ellos se tiende a esquemas de planificación particionados, de forma que las actividades y mensajes se localizan estáticamente en alguno de los recursos procesadores. De esta forma cada uno de los recursos procesadores del sistema se planifica de manera independiente, pudiendo hacer uso de las políticas de planificación tradicionales como FP o EDF, tanto en los procesadores como en las redes de comunicación.

1.5.3.1 Planificación LC-EDF y GC-EDF

La planificación EDF ha sido ampliamente estudiada para sistemas monoprocesadores desde el artículo seminal de Liu y Layland [LIU73], y su comportamiento minuciosamente comparado con FP, por ejemplo en [BUT05]. EDF ha sido igualmente objeto de investigación en sistemas multiprocesadores, como se extrae del recopilatorio llevado a cabo por Davis y Burns [DAV11].

Sin embargo, en la actualidad no existe un cuerpo teórico equivalente para sistemas distribuidos EDF de tiempo real. Este vacío no se puede suplir utilizando las nociones aplicables a los sistemas multiprocesadores particionados EDF debido a dos factores fundamentales que los diferencian: (1) la sincronización de los relojes entre los distintos recursos procesadores, y (2) la asignación del requisito temporal de principio a fin a un flujo e2e y no a cada actividad por separado. A continuación se tratan ambos problemas, y cómo los manejamos en el transcurso de esta tesis.

Las actividades en EDF se planifican mediante su plazo, también conocido como plazo de planificación. Sin embargo, en un sistema distribuido las actividades típicamente se agrupan en flujos e2e, en los que el plazo se establece típicamente para todo el flujo, por lo que cada actividad carece de un plazo con el que ser planificada. Este problema se soluciona definiendo técnicas que asignan a cada actividad su plazo de planificación particular. Este plazo no representa un requisito temporal, y es utilizado únicamente para tomar decisiones de planificación. La asignación de plazos de planificación se suele realizar mediante la aplicación de alguna técnica que reparte los requisitos temporales establecidos entre las actividades.

En los sistemas multiprocesadores, al estar los diferentes procesadores estrechamente conectados, poseen típicamente una noción global del tiempo, pero esto no siempre es cierto en los sistemas distribuidos, en los que esta noción global de tiempo requiere la implementación de mecanismos de sincronización de relojes. Un planificador EDF elige para la ejecución a la actividad con el menor plazo absoluto. En sistemas monoprocesadores (ver Sección 1.5.2) el plazo absoluto se calcula como la suma entre el instante de activación de la actividad y su propio plazo, pero en sistemas distribuidos podemos encontrar dos tipos de planificación EDF, en función de la disponibilidad de una referencia global de tiempo.

El primer tipo de planificación EDF en sistemas distribuidos lo llamamos LC-EDF (*Local-Clock* EDF). En LC-EDF el plazo absoluto para planificación toma como referencia el instante de activación de la propia actividad. Como para este cálculo sólo se requiere información temporal local al propio recurso procesador, no se requiere sincronización entre los diferentes relojes del sistema. Definimos td_{ij}^{k} como el plazo absoluto de la *k*-ésima activación de la actividad τ_{ij}, que se calcula según la siguiente ecuación:

$$td_{ij}^{k} = t_{ij}^{k} + Sd_{ij} \tag{1-7}$$

donde t_{ij}^{k} es el instante de la *k*-ésima activación de τ_{ij}, y Sd_{ij} el plazo local de planificación de dicha actividad.

Al segundo tipo de planificación EDF en sistemas distribuidos lo llamamos GC-EDF (*Global-Clock EDF*). En GC-EDF, el plazo absoluto para planificación toma como referencia el instante de activación del flujo e2e. Como este evento ha podido ocurrir en otro recurso procesador distinto, se requiere sincronización en los relojes del sistema. Definimos td_{ij}^{k} como el plazo absoluto de la *k*-ésima activación de la actividad τ_{ij}, calculado según la siguiente ecuación:

$$td_{ij}^{k} = t_{i}^{k} + SD_{ij} \tag{1-8}$$

donde t_{i}^{k} es el instante de la *k*-ésima activación de la primera actividad del flujo Γ_i, y SD_{ij} el plazo global de planificación de la actividad τ_{ij}, referido al evento externo.

Es importante hacer notar que los plazos de planificación, tanto locales como globales, no representan requisitos temporales, sino que son valores utilizados exclusivamente para planificar las actividades. Adicionalmente, sus valores son independientes de los posibles plazos locales (d_{ij}) o globales (D_{ij}) que pueda tener impuestos el sistema como requisitos temporales.

Existen diversos métodos con los que sincronizar los relojes en un sistema distribuido. Estos se pueden dividir en dos tipos [KOP87][KOP04]: métodos de sincronización interna, y métodos de sincronización externa.

En los métodos de sincronización interna se define un protocolo distribuido por el cual los diferentes recursos procesadores se envían periódicamente mensajes con el estado actual de su reloj, de forma que cada recurso procesador recibe la información de los demás relojes en el sistema. Con esta información, y conociendo además los tiempos de transmisión de los mensajes, cada recurso procesador corrige su propio reloj utilizando alguna función. El tiempo de transmisión de estos mensajes puede poseer cierta variabilidad que afecta negativamente al proceso de sincronización de los relojes. Esta variabilidad se debe a factores como retrasos no deterministas en el sistema operativo o en el nivel de aplicación, y se pueden reducir utilizando soluciones *hardware* como la definida en el estándar de sincronización de relojes IEEE 1588 [EID06].

Los métodos de sincronización externa consisten en la utilización de un elemento especializado que transmite de forma periódica la referencia de tiempo actual. Un ejemplo podría ser un dispositivo conectado a la red de satélites GPS, o un reloj atómico. Este tipo de sincronización es unidireccional, en el sentido de que el elemento especializado envía los mensajes a los diferentes recursos procesadores, que únicamente reciben y procesan esta información.

Cuando se utiliza un protocolo de sincronización hay que tener en cuenta las posibles sobrecargas añadidas en el sistema, en forma de nuevos mensajes transmitidos en las redes de comunicación, o en nuevas actividades que se ejecutan en los procesadores que calculan las correcciones en los relojes, además de los posibles problemas que pueden ir asociados a la precisión y fiabilidad de la sincronización.

1.5.4 Parámetros de planificación

La política de planificación toma su decisión en base al parámetro de planificación que poseen las actividades. Hemos identificado tres tipos distintos de parámetros de planificación en sistemas distribuidos, que completan el modelo de sistema utilizado en esta tesis:

- Prioridad fija (P_{ij}) para planificación FP.
- Plazo local de planificación (Sd_{ij}) para planificación LC-EDF, en el que el plazo toma como referencia de tiempos el instante de activación de la propia actividad τ_{ij}, en el recurso procesador en el que está alojada.
- Plazo global de planificación (SD_{ij}) para planificación GC-EDF, que toma como referencia el instante de activación de la primera actividad del flujo e2e al que pertenece τ_{ij}.

Los parámetros de planificación son valores que se asignan de manera *off-line* a las actividades. El problema de calcular qué parámetros de planificación asignar a un sistema distribuido para que se cumplan todos los requisitos temporales es de gran interés en el campo de los sistemas de tiempo real, y es llevado a cabo por las técnicas de asignación de parámetros de planificación, también conocidas como técnicas de optimización.

Durante las últimas décadas se han ido proponiendo diferentes métodos para resolver la asignación de parámetros de planificación, y aún hoy en día sigue siendo un campo con gran actividad por los márgenes de mejora que aún existen en sistemas complejos como los distribuidos. En las Secciones 1.6 y 1.8 se introducirán las principales técnicas de asignación existentes en la actualidad para sistemas monoprocesadores y distribuidos, respectivamente.

1.6 Análisis y asignación de parámetros de planificación en sistemas monoprocesadores

En las secciones precedentes se fueron introduciendo algunos conceptos fundamentales en el estudio de los sistemas de tiempo real. En particular, se describió cómo las políticas de planificación actúan en base a los parámetros de planificación de las actividades (prioridades fijas en FP, y plazos de planificación en EDF), y cómo estos parámetros se deben calcular mediante el uso de alguna técnica de asignación de parámetros de planificación. Se comentó además cómo en los sistemas gobernados por eventos la planificabilidad del sistema se puede verificar de manera *off-line* aplicando alguna técnica de análisis de planificabilidad.

En la presente sección se introducirán las principales técnicas de análisis y optimización de sistemas monoprocesadores. que sientan las bases para las técnicas aplicables en sistemas distribuidos. Estas operan sobre una configuración estática del sistema que ha sido modelado.

1.6.1 Sistemas con prioridades fijas: Teoría RMA

La teoría RMA (*Rate Monotonic Analysis*) es un cuerpo matemático para el análisis de planificabilidad de actividades planificadas por prioridades fijas en sistemas monoprocesadores. Tiene sus orígenes en el algoritmo de asignación de prioridades fijas RM (*Rate Monotonic*) de Liu y Layland [LIU73].

El algoritmo RM original es aplicable a sistemas de un único procesador en los que todas las actividades son periódicas e independientes (flujos e2e simples, y no se comparten recursos). En este tipo de sistemas el algoritmo RM asigna prioridades fijas a las actividades de forma inversamente proporcional a sus periodos, de manera que la actividad con periodo más corto es la más prioritaria. Liu y Layland demostraron que entre los algoritmos de prioridades fijas, RM es óptimo cuando los plazos de las actividades son iguales a sus respectivos periodos. Esto significa que si existe alguna asignación de prioridades fijas que hace el sistema planificable, una asignación RM también lo consigue.

Una condición suficiente de planificabilidad para un sistema monoprocesador con utilización U y n actividades periódicas e independientes planificadas con RM, se expresa con la siguiente ecuación, que constituye un test de planificabilidad [LIU73]:

$$U \leq n\left(2^{1/n} - 1\right) \tag{1-9}$$

Este trabajo inicial de Liu y Layland, aunque muy restrictivo en cuanto al tipo de sistemas sobre los que se puede aplicar, sirvió como base para posteriores trabajos que abrieron el espectro de sistemas analizables.

El test de planificabilidad descrito por la ecuación (1-9) establece una condición suficiente para la planificabilidad, pero no necesaria. El límite de utilización ($n \rightarrow \infty$) que establece el test es aproximadamente del 69.3%. Sin embargo, en pruebas con conjuntos grandes de actividades generadas de manera aleatoria, se obtienen utilizaciones planificables en torno al 88% [LEH89]. Para poder determinar con mayor precisión la planificabilidad, se define un test exacto de planificabilidad [HAR87] [JOS86]. En el test exacto, o también conocido como test de tiempos de respuesta, se calculan los tiempos de respuesta de peor caso de todas las actividades que forman parte del sistema y para determinar la planificabilidad se comparan con los plazos impuestos. Gracias a este tipo de test se pueden alcanzar utilizaciones mayores, y además dan al diseñador del sistema una idea del margen que posee el sistema para cumplir sus plazos.

Por otra parte, en 1982, Leung y Whitehead [LEU82] consideraron el caso en el que los plazos pudieran ser menores que sus periodos, y propusieron la asignación *Deadline Monotonic* (DM), con la que se asignan las prioridades fijas de acuerdo a los plazos, de forma que las actividades con menores plazos siempre tienen mayor prioridad. Demostraron que el algoritmo DM es óptimo entre los algoritmos de prioridades fijas cuando los plazos de las actividades son iguales o menores que sus respectivos periodos.

Para el caso de plazos superiores al periodo, Lehoczy [LEH90] propone una serie de test de planificabilidad suficientes pero no necesarios basados en la utilización. Tindell *et al* [TIN94d] extendieron el análisis de tiempos de respuesta para el caso de plazos arbitrarios, y Audsley [AUD91] propuso un algoritmo que es óptimo en la asignación de prioridades fijas y plazos arbitrarios.

Los recursos compartidos son otro elemento que debe ser tenido en cuenta en el análisis de planificabilidad. Las actividades acceden a los recursos compartidos en exclusión mutua, hecho que provoca la aparición de tiempos de bloqueo. En el acceso a los recursos compartidos se utilizan protocolos de sincronización que limitan el efecto de inversión de prioridad no acotada. Así por ejemplo, el Protocolo de Techo de Prioridad Inmediato (*Inmediate Priority Ceiling Protocol*, o IPCP) [LAM80][SHA90c] establece que la actividad debe acceder a un recurso compartido ejecutando con la prioridad máxima de entre todas las prioridades de las actividades que puedan hacer uso de dicho recurso. Este mecanismo está implementado en el lenguaje Ada [TUC12] y el estándar POSIX [POSIX], y es además el protocolo utilizado en la herramienta de análisis MAST [MASTh].

Otro protocolo de sincronización para acceso a recursos compartidos es el Protocolo de Techo de Prioridad (*Priority Ceiling Protocol*, o PCP) [SHA90]. En PCP se asigna a cada recurso compartido una prioridad llamada techo de prioridad del recurso, que es una cota superior de las prioridades de todas las actividades que acceden a dicho recurso. Una actividad no puede acceder a un recurso compartido a menos que su prioridad sea estrictamente superior al techo máximo de todos los recursos compartidos en uso por otras actividades. Adicionalmente se define un mecanismo de herencia de prioridad, que establece que cada actividad accede a su recurso compartido ejecutando con su propia prioridad, y hereda la prioridad de las actividades de prioridad superior a las que va bloqueando. Con PCP se evitan situaciones de bloqueo mutuo, y se reduce el número total de bloqueos.

Aunque el modelo original de Liu y Layland asume que las actividades son perfectamente periódicas, también puede ser aplicado a sistemas con actividades no periódicas si se puede limitar el tiempo mínimo entre activaciones [LIJ00]. Sin embargo, para actividades aperiódicas con tiempo entre llegadas no acotado, hay que tener en cuenta consideraciones especiales con objeto de limitar el impacto que estas actividades pueden tener sobre el resto del sistema.

Una primera solución la aporta el Servidor de Muestreo Periódico (*Polling Server*) [SHA86], que consiste en una actividad periódica con una prioridad fija y un tiempo de ejecución (capacidad) determinados. En cada una de sus activaciones, el Servidor de Muestreo Periódico comprueba si hay alguna actividad aperiódica pendiente de ejecución, y las ejecuta por un tiempo máximo definido por su capacidad. Al comportarse como una actividad periódica normal, es analizable por las técnicas RMA tradicionales. El periodo y capacidad del Servidor de Muestreo Periódico deben calcularse *off-line* de manera que puedan acomodar todo el trabajo aperiódico que deba atenderse.

El principal problema del Servidor de Muestreo Periódico reside en que puede provocar tiempos de respuesta innecesariamente largos en las actividades no periódicas. Por ejemplo, si su capacidad se ve sobrepasada, no se puede volver a ejecutar el trabajo aperiódico hasta el siguiente periodo del Servidor de Muestreo Periódico, aunque el sistema no esté ejecutando ninguna otra actividad. Para solventar ineficiencias como esta, se definen servidores más avanzados en los que se establecen reglas para el consumo y rellenado de la capacidad, como por ejemplo el Servidor Diferido (*Deferrable Server*) [STR95] o el Servidor Esporádico (*Sporadic Server*) [SPR89]. Este último está incluido en el estándar POSIX como una política de planificación disponible [POSIXb].

1.6.2 Sistemas con prioridades dinámicas EDF

Los sistemas planificados por prioridades fijas nos permiten, de una forma relativamente sencilla, alcanzar utilizaciones altas en los sistemas manteniendo la planificabilidad. Con una planificación por prioridades dinámicas es posible utilizar los recursos procesadores de una manera más eficiente aún.

Un sistema monoprocesador con flujos e2e simples e independientes, y con plazos iguales a los periodos, es planificable con el algoritmo EDF sí y sólo sí tiene una utilización menor o igual al 100% [LIU73] :

$$\sum_{\forall i} \frac{C_{i1}}{T_i} \leq 1 \tag{1-10}$$

Además, se demuestra que EDF es un algoritmo óptimo entre todos los algoritmos expulsores cuando las actividades son independientes [DER74].

Mientras que en sistemas FP se definen diferentes métodos para calcular las prioridades fijas de las actividades, como por ejemplo los criterios *Rate Monotonic* o *Deadline Monotonic* (ver Sección 1.6.1), en EDF las actividades se planifican a partir de los requisitos temporales impuestos en las actividades. Aunque estos requisitos puedan venir especificados por las características intrínsecas del sistema, existen trabajos [BAL08][BAR99] que muestran cómo reducciones en los plazos de las actividades pueden conseguir aumentar el rendimiento en el sistema, por ejemplo reduciendo los tiempos de respuesta o los *jitters*.

Para determinar la planificabilidad en el caso general de plazos arbitrarios, Baruah *et al* proponen el Criterio de la Demanda de Procesador (*Processor Demand Criterion*) [BAR90][BAR90b]. El criterio se basa en la idea de que la demanda de procesador en un intervalo de tiempo determinado no puede exceder el tiempo disponible. Partiendo de esta idea, un sistema de actividades periódicas es planificable por EDF si tiene una utilización menor del 100% y se cumple:

$$\forall L > 0 \ , \ \sum_{\forall i} \left\lfloor \frac{L + T_i - SD_{i1}}{T_i} \right\rfloor C_i \leq L \tag{1-11}$$

Los puntos en los que se debe comprobar el test de la ecuación (1-11) no exceden $\max\left\{SD_{11}, SD_{21}, ..., \mathrm{SD}_{n1}, L^*\right\}$, donde:

$$L^* = \frac{\sum_{\forall i} \left(T_i - SD_{i1}\right) \frac{C_{i1}}{T_i}}{1 - U} \tag{1-12}$$

y donde U es la utilización del sistema.

Al igual que para el caso de sistemas con prioridades fijas, para sistemas EDF también se definen técnicas que tratan con fenómenos adicionales que completan el espectro de sistemas analizables. Un método habitual para el manejo de actividades aperiódicas es el llamado Servidor de Ancho de Banda Total (*Total Bandwidth Server*) [SPU96b], con el que a cada activación de las actividades aperiódicas se le asigna un plazo de planificación de forma que el trabajo no periódico no supere un nivel de carga máximo definido. Otro método es el Servidor de Ancho de Banda Constante (*Constant Bandwidth Server*) [ABE04], con el que se pueden reservar fracciones de tiempo de procesador para ejecutar

actividades aperiódicas con demanda computacional no acotada, sin que afecten a las actividades con requisito temporales estrictos.

El método más habitual para acotar los tiempos de bloqueo por acceso a recursos compartidos en EDF es el protocolo SRP (*Stack Resource Protocol*) [BAK91], que también es aplicable en sistemas FP. Con SRP en conjunción con EDF, a cada actividad se le asigna, además de una prioridad dinámica de acuerdo a su plazo absoluto (prioridad dinámica de EDF), un nivel de expulsión (*preemption level)* de forma inversamente proporcional a su plazo de planificación. Adicionalmente, a cada recurso compartido se le asigna un techo de expulsión (*preemption ceiling)* que es el nivel de expulsión máximo de entre todas las actividades que acceden a dicho recurso. Se define el techo del sistema como el máximo de los techos de expulsión de todos los recursos actualmente en uso. Una actividad planificada según SRP con EDF sólo comenzará a ejecutarse si su prioridad dinámica es la mayor entre todas las actividades activas (planificación EDF normal), y además su nivel de expulsión es mayor que el techo del sistema. Con SRP se garantiza que si una actividad comienza su ejecución, no va a bloquearse por acceder a un recurso compartido. Por ello, el bloqueo que se considera en el análisis realmente es el tiempo que la actividad permanece esperando a que la regla de lanzamiento de SRP se cumpla. Recientemente se ha propuesto un protocolo alternativo llamado DFP (*Deadline Floor Inheritance Protocol*) [BUR14], que propone un esquema más intuitivo, y prescinde del nivel de expulsión.

A pesar de que con EDF se consiguen planificar mayores utilizaciones de los recursos, históricamente FP ha sido la política basada en prioridades más utilizada en aplicaciones de tiempo real, debido principalmente a su mayor soporte en sistemas operativos comerciales. Adicionalmente, a FP se le suelen atribuir una serie de ventajas sobre EDF que favorecen su uso, tales como una mayor simplicidad de implementación, menores sobrecargas en tiempo de ejecución, o un comportamiento más predecible en situaciones de sobrecarga. Buttazzo [BUT05] relativiza estas ventajas:

- La política FP es más sencilla de implementar únicamente en aquellos sistemas operativos de tiempo real que no soporten de forma nativa características temporales, como periodos o plazos.
- La necesidad de calcular los plazos absolutos en la planificación EDF añade sobrecargas en tiempo de ejecución que no están presentes en FP. Sin embargo, este hecho se ve contrarrestado por el menor número de cambios de contexto (expulsiones de actividades por otras de mayor prioridad) que se producen con EDF.
- En momentos de sobrecarga permanente, con EDF las actividades se ejecutan como si tuvieran un periodo mayor, pero ninguna actividad deja de ejecutarse. Con FP las actividades de menor prioridad quedarían totalmente inhibidas. Si la sobrecarga es transitoria, en FP se asegura el cumplimiento de los plazos de las actividades con prioridad mayor que la actividad que generó la sobrecarga, mientras que en EDF cualquier actividad podría perder su plazo. Sin embargo, esta ventaja de FP sólo sería efectiva si se conociera de antemano cuál es la actividad que va a generar la sobrecarga.

1.7 Análisis de planificabilidad en sistemas distribuidos

Para determinar la planificabilidad de un sistema distribuido se deben analizar cada uno de sus recursos procesadores (procesadores y redes). Utilizar un esquema particionado, por el cual cada actividad se localiza de forma estática en un recurso procesador, nos permite que el análisis de cada recurso pueda llevarse a cabo con técnicas similares a las aplicadas en los sistemas monoprocesadores, aunque con modificaciones para tratar dos características presentes en los sistemas distribuidos, (1) la imposición de un orden de precedencia en la ejecución de las actividades dentro de un flujo e2e, y (2) el análisis en las redes de comunicación.

1.7.1 Análisis en redes de comunicación

Las redes de comunicación son recursos procesadores que desde el punto de vista del *hardware* poseen una arquitectura muy distinta con respecto a la de un procesador. Sin embargo, existen redes tales como el bus CAN [CAN], Futurebus+ [SHA90b], o el *Token Ring* (IEEE 802.5) [TOKR], que utilizan esquemas de comunicación basados en prioridades fijas sin expulsiones, que se pueden analizar aplicando directamente el análisis de tiempos de respuesta para sistemas monoprocesadores en los mensajes [TIN94c][STR88].

Alternativamente, también es habitual el uso de redes de propósito general, que mediante la aplicación de algún protocolo o mecanismo de acceso, adquieren un comportamiento predecible, y que permite el uso de políticas de planificación tradicionalmente usadas en los procesadores, como FP o EDF. De esta manera, a estas redes se las puede considerar como un procesador más en el sistema en el que se aplican los análisis de tiempo de respuesta tradicionales. Por ejemplo, en la especificación IEEE 802.1Qbb [1QBB] se define un método de control de flujo por prioridades para Ethernet conmutado (*switched Ethernet*) que permite el uso de esta red de propósito general en aplicaciones de tiempo real. De forma similar, el estándar ARINC-664 Part 7, también llamado AFDX [AFDX], define una red de tiempo real basada en Ethernet conmutado, para la comunicación entre sistemas de aviónica. En esta red se especifican dos niveles de prioridad en los conmutadores (*switches*), y existen técnicas que permiten la integración de su análisis en el conjunto de un sistema distribuido completo [GUT14]. Otro mecanismo para utilizar la red Ethernet en aplicaciones de tiempo real es mediante el protocolo RT-EP [MAR05], que es un protocolo de paso de testigo que permite la asignación de prioridades en los mensajes, y no requiere ninguna modificación en el *hardware*. Tomando como base RT-EP, se define el protocolo AR-TP [URE06] para sistemas de alta integridad, con el que se consiguen velocidades de transmisión mayores.

Existen también trabajos que utilizan EDF a la hora de planificar mensajes en redes, aunque su uso tenga aún un carácter experimental. Por ejemplo, en [PED02] se propone un planificador EDF sobre el bus CAN, reportando mejoras en la utilización planificable de la red, y una mejora en la respuesta de los mensajes con mayores periodos. En [DIN01] se estudia la aplicación del algoritmo EDF en redes *Token Ring* y Ethernet.

En general el uso de estos protocolos nos permiten modelar los mensajes en la red como una actividad ejecutando en un procesador, en la que el tiempo de ejecución de peor caso es el tiempo de transmisión de peor caso del mensaje. Adicionalmente, se

deben tener en cuenta otros factores, como los bloqueos que se presentan por el hecho de que los paquetes en los que se pueden trocear los mensajes no sean expulsables, o la posible limitación del número de prioridades, como por ejemplo en el *Token Ring*, en el que sólo se definen ocho niveles, o en AFDX con dos niveles de prioridad.

1.7.2 Análisis de flujos de principio a fin

El análisis de un flujo e2e debe tener en cuenta los *jitters* inherentes a la ejecución de las actividades en el flujo e2e, y en su caso también los *offsets* si se han especificado. Audsley *et al* [AUD93] definen un análisis de tiempos de respuesta para actividades periódicas planificadas por FP y afectadas por *jitter* de activación, para el caso de plazos inferiores al periodo. Este análisis es extendido por Tindell *et al* [TIN94d] para plazos arbitrarios. Sin embargo, estos análisis no son aplicables directamente a sistemas distribuidos, ya que asumen que los *jitters* de activación son valores constantes y conocidos de antemano. En un flujo e2e todas las actividades excepto la primera en el flujo presentan un *jitter* de activación que es debido a la variabilidad en el tiempo de respuesta de la actividad precedente. A su vez el tiempo de respuesta de cualquier actividad depende de este *jitter* de activación. Tindell y Clark [TIN94] resuelven esta interdependencia *jitter*-tiempo de respuesta mediante la técnica de análisis llamada holística, definida para sistemas con planificación FP.

En el análisis holístico, a cada recurso procesador se le considera independiente de los demás, y sobre cada uno de ellos se aplican las técnicas de análisis de actividades periódicas FP afectadas por *jitter*. Tindell y Clark muestran que la interdependencia *jitter*-tiempo de respuesta puede resolverse de manera iterativa: el análisis comienza con unos valores iniciales para *jitters* y tiempos de respuesta, en cada iteración se calculan nuevos tiempos de respuesta aplicando las fórmulas de análisis, y con estos valores se actualizan los *jitters* de todas las actividades. La monotonicidad del tiempo de respuesta y del *jitter* llevan al análisis iterativo a converger a una solución.

A partir del trabajo de Tindell y Clark para sistemas con planificación FP, se proponen nuevas técnicas de análisis para sistemas distribuidos planificados por otras políticas de planificación. Así, Spuri [SPU96] propone una técnica de análisis para actividades periódicas afectadas por *jitter* de activación y planificadas por EDF, en la que los plazos de planificación toman como referencia el mismo punto temporal que los tiempos de respuesta y los *jitters*. Por lo tanto, y aunque Spuri no lo especifique de manera explícita, este análisis se adapta a lo que consideraríamos un análisis holístico para sistemas distribuidos GC-EDF, resolviendo la interdependencia *jitter*-tiempo de respuesta de la misma manera que lo hacen Tindell y Clark en el análisis holístico para FP. La adaptación del análisis holístico de Spuri a sistemas distribuidos planificados por LC-EDF se realizó en [RIV10].

El análisis holístico aporta una visión del sistema que es inherentemente pesimista, ya que no tiene en cuenta que las relaciones temporales entre diferentes respuestas son interdependientes, pudiendo así construir una situación que es peor que el peor caso real en un sistema distribuido. Sin embargo, este pesimismo implica que los resultados que se obtienen son cotas superiores a los tiempos de respuesta de peor caso, que permiten garantizar la planificabilidad de un sistema en el caso de que se cumplan los plazos.

Con objeto de reducir el nivel de pesimismo del análisis holístico, se propone una familia de técnicas de análisis basadas en *offsets*. Tindell [TIN94b] desarrolló una técnica

que permite calcular de forma exacta los tiempos de respuesta de peor caso en un sistema planificado por FP, compuesto por actividades periódicas y activadas por *offsets*. El problema asociado a esta técnica de análisis exacta es que es computacionalmente intratable, salvo para ejemplos triviales. Sin embargo, sirven como base para la definición de otras técnicas de análisis que, aunque no siendo exactas, sí que son tratables computacionalmente. Así, en [TIN94b] también se propone una aproximación al análisis exacto que posee una complejidad polinomial con respecto al número de actividades en el sistema, y aunque los resultados no son exactos, son menos pesimistas que los obtenidos por el análisis holístico. Palencia y González Harbour [PAL98] modificaron esta técnica para soportar *offsets* mayores que los periodos, y también propusieron una aproximación para integrar el efecto de los tiempos de respuesta de mejor caso dentro un flujo e2e que acota el pesimismo.

En los resultados expuestos en [PAL98], se observa que los tiempos de respuesta obtenidos por la técnica holística son en promedio el doble que los obtenidos por la técnica basada en *offsets*, para el conjunto de sistemas estudiados. Con esta comparación se quiere dejar patente la importancia del nivel de pesimismo que puede introducir la técnica de análisis que se use, ya que unos resultados excesivamente pesimistas pueden provocar que se dé por no planificables sistemas que sí lo son.

Utilizando como base el análisis holístico para GC-EDF de Spuri, Palencia y González Harbour [PAL03] adaptaron los conceptos del análisis basado en *offsets* para aplicarlo en sistemas GC-EDF. La definición del análisis basado en *offsets* para LC-EDF se presentó recientemente [DIA14].

Una descripción más detallada de las técnicas de análisis para sistemas distribuidos aquí introducidas, y que serán utilizadas en esta tesis, se presentará en las Secciones 2.1 y 2.2 para sistemas FP y EDF respectivamente.

Las técnicas de análisis de planificabilidad para sistemas distribuidos aquí introducidas son herederas del análisis original de Tindell y Clark para sistemas FP, en la que se aporta una visión holística del sistema, es decir, el sistema es analizado como un todo teniendo en cuenta la dependencia que existe entre actividades a través de los flujos e2e. Existen otras metodologías de análisis que se rigen por visiones distintas del sistema. Así por ejemplo, además de la visión holística, en [PER09] se destacan otras dos concepciones, que son:

- Análisis composicionales: Este análisis se basa en la aplicación de análisis locales en cada actividad, y en la propagación de los resultados [HEN05].
- Análisis basado en autómatas temporales [ALU94].

1.8 Asignación de parámetros de planificación en sistemas distribuidos

Otro problema fundamental a tratar en los sistemas distribuidos consiste en la asignación de los parámetros de planificación a todas las actividades del sistema. Los algoritmos de asignación presentados en la teoría RMA, y que son óptimos para sistemas monoprocesadores, no son aplicables a los sistemas distribuidos debido a la interdependencia que presentan sus respuestas temporales. Para sistemas distribuidos se

requieren técnicas más avanzadas que tengan en cuentan no sólo el periodos o el plazo de cada actividad (el plazo se puede especificar a la totalidad del flujo, a alguna actividad intermedia, etc.), sino también el tiempo de respuesta o el *jitter* de activación.

El objetivo de la técnica de asignación de parámetros de planificación es el de encontrar una combinación de parámetros de planificación para todas las actividades de forma que el sistema sea planificable. Este es un problema NP-difícil [BUR91][TIN92] salvo en un grupo de sistemas triviales con poco interés práctico, por lo que se descarta el uso de métodos de búsqueda por fuerza bruta.

Como solución al problema de la asignación de prioridades fijas sistemas distribuidos FP, conjuntamente con la asignación de actividades a recursos procesadores, Tindell *et al* [TIN92] propusieron la utilización de la técnica de propósito general llamada templado simulado (*simulated annealing*). El templado simulado es una técnica de optimización que se utiliza para diversos ámbitos [KIR83], cuya principal característica es que posee la capacidad de evitar converger a soluciones que representan mínimos locales (soluciones que no son las mejores de forma global). Esto lo consigue realizando saltos aleatorios en el espacio de soluciones, pero en el que a medida que transcurre el proceso de optimización, se reduce la probabilidad de aceptar saltos a soluciones peores. Esta técnica necesita de una técnica de análisis de planificabilidad para verificar si las soluciones alcanzadas son válidas. En [GUT95b] se propone una modificación al templado simulado propuesto por Tindell *et al*, con el fin de poder distinguir una solución planificable de otra que no lo es. Con esta modificación se puede utilizar este algoritmo para optimizar una solución ya planificable, iterando sobre ella.

Con objeto de mejorar los resultados obtenidos por el templado simulado, tanto en calidad de resultados, como en el tiempo empleado en conseguirlos, se propuso el algoritmo HOPA [GUT95] para sistemas FP. HOPA (*Heuristic Optimized Priority Assignment*) es un algoritmo heurístico e iterativo que hace uso de uno los factores más determinantes en el comportamiento temporal de los sistemas distribuidos, el tiempo de respuesta de peor caso, y lo hace además desde dos puntos de vista ortogonales: el tiempo de respuesta de cada actividad respecto del flujo e2e al que pertenece, y el tiempo de respuesta de cada actividad respecto del recurso procesador en el que está alojada. Por lo tanto, es un algoritmo que para su funcionamiento depende de una técnica de análisis con la que obtener los tiempos de respuesta de peor caso. Al igual que el algoritmo del templado simulado implementado en [GUT95b], posee la capacidad de optimizar una solución ya planificable, iterando sobre ella. En la comparación de ambas técnicas realizada en [GUT95] se muestra que HOPA es dos órdenes de magnitud más rápido que el templado simulado, y que es capaz de planificar mayor número de sistemas y con mejores soluciones (mayor *slack*).

Otro camino alternativo para resolver el problema de la asignación de prioridades fijas en sistemas FP lo proponen Azketa *et al* [AZK11] mediante el uso de un algoritmo genético. Los algoritmos genéticos [HOL92] implementan una búsqueda heurística implementando mecanismos que se asemejan a los observados en la selección natural de las especies. A partir de una solución inicial, se crea una población de soluciones "hijas". Las soluciones más "sanas" de esta población se eligen para crear la población de la siguiente generación. De esta forma, las peores soluciones se eliminan progresivamente, y se converge hacia poblaciones con mejores soluciones. La implementación de Azketa *et al* consigue buenos resultados si como solución inicial se suministra una asignación como la obtenida por HOPA. La contrapartida es que el algoritmo genético posee un coste computacional mucho mayor en comparación con HOPA.

Como son de propósito general, tanto el templado simulado como los algoritmos genéticos, no tienen la capacidad de hacer uso de ninguna propiedad de los sistemas distribuidos de tiempo real con la que guiar la obtención de sus resultados. Esto provoca que sean unos algoritmos inherentemente más lentos que otros creados específicamente para la asignación de parámetros de planificación, como HOPA.

La asignación de plazos de planificación representa *a priori* un problema más complejo. Las prioridades fijas son valores discretos que se seleccionan entre un rango disponible de prioridades, por ejemplo, 256 niveles de prioridades. Sin embargo, la resolución de los plazos es mucho más elevada. Por ejemplo, si queremos asignar plazos de planificación en el rango entre 100ms. y 200 ms. con una resolución de 1 microsegundo, existen 100000 diferentes valores que podemos asignar a dicho plazo de planificación. Cuando combinamos un número no trivial de actividades, el número de combinaciones posibles convierte a la asignación de plazos de planificación en un problema más complejo que el de asignación de prioridades fijas.

Un método sencillo para afrontar el problema de la asignación de plazos de planificación consiste en el reparto del plazo de principio a fin entre las actividades del flujo e2e al que se impone. Existen diversos métodos para llevar a cabo este reparto. De entre todos ellos Liu [LIJ00] destaca los siguientes:

- UD: plazos de planificación iguales al plazo de principio a fin del flujo e2e.
- ED: plazos de planificación iguales al plazo de principio a fin del flujo e2e, menos la suma de los tiempos de ejecución de peor caso de las actividades posteriores en el flujo e2e.
- PD: plazos de planificación de acuerdo a un reparto proporcional al tiempo de ejecución de la actividad.
- NPD: plazos de planificación de acuerdo a un reparto proporcional al tiempo de ejecución de la actividad, normalizado a la utilización del recurso procesador en el que resida.

Estos repartos son algoritmos no iterativos, y por lo tanto carecen de la capacidad de mejorar la solución que obtienen (sea planificable o no lo sea). Con el objetivo de mejorar el rendimiento de estas técnicas de asignación de plazos de planificación, se propuso el algoritmo HOSDA (*Heuristic Optimized Scheduling Deadline Assignment*). HOSDA comienza con un reparto de plazos como los descritos previamente, y adopta los mismos mecanismos heurísticos del algoritmo HOPA para mejorar las asignaciones de manera iterativa, hasta encontrar una solución planificable. Este algoritmo fue desarrollado como trabajo previo a esta tesis, y se define tanto para sistemas con planificación LC-EDF [RIV08][RIV10], como para sistemas con planificación GC-EDF [RIV09].

En esta sección se han presentado a modo de introducción una serie de algoritmos de asignación de parámetros de planificación para sistemas FP, GC-EDF y LC-EDF. Estos algoritmos son de gran importancia en el transcurso de esta tesis, y serán descritos con más detalle en las Secciones 2.3 y 2.4.

1.9 Herramientas de análisis y optimización

Las técnicas de análisis de planificabilidad y asignación de parámetros de planificación carecen de utilidad si no se implementan en herramientas *software* con las que poder aplicarlas. Estas herramientas típicamente definen un modelo con el que describir los sistemas, y una sintaxis con la que trasladar este modelo a un fichero de entrada. Sobre el sistema descrito en el fichero de entrada se aplican las técnicas que requiera el usuario, y se devuelven los resultados requeridos (estado de la planificabilidad, tiempos de respuesta, asignación de prioridades, etc.). Ejemplos de estas herramientas usadas en ámbitos industriales o de investigación son los siguientes:

- Cheddar [SIN04][CHEDD]: Es una herramienta gratuita y de código abierto que integra el análisis de planificabilidad y simulación de sistemas de tiempo real. Incluye soporte para las políticas de planificación FP (RM y DM), EDF y LLF. Soporta flujos e2e, para los cuales implementa el análisis holístico de Tindell *et al* [TIN94]. Los sistemas se pueden definir en el lenguaje de descripción AADL [FEI06], o en el lenguaje propio de Cheddar.
- RAPID-RMA [RAPID]: Herramienta comercial y de código cerrado para el análisis de sistemas de tiempo real, con soporte para políticas de planificación FP (RM y DM) y EDF. El análisis de flujos e2e periódicos queda limitado a sistemas con un único procesador.
- SymTA/S [HAM04][HEN05][SYMT]: Es una herramienta comercial y de código cerrado, que implementa una gran variedad de estándares utilizados en la industria de la aviación y la automoción. Permite el análisis de planificabilidad de sistemas monoprocesadores, multiprocesadores y distribuidos mediante un método composicional. Cada actividad se analiza con un análisis local, y los resultados se propagan de acuerdo a un modelo concreto de eventos. Además, los modelos de eventos se pueden transformar para cada actividad dependiendo de los requisitos de cada componente. El análisis local soporta FP, EDF, *Round Robin* y planificación estática.
- UPPAAL [UPPA]: UPPAAL es una herramienta comercial (gratuita para un uso académico) que permite modelar, simular y verificar sistemas de tiempo real, haciendo uso de la teoría de autómatas temporales [ALU94]. Está especialmente indicado para analizar la validez de protocolos de comunicación. Su licencia no permite modificar su código fuente salvo expresa autorización de los autores.

La alternativa que se utilizará en esta tesis se presenta con la herramienta de análisis y optimización MAST (*Modeling and Analysis Suite for real-Time applications*) [MAST] [MASTh], que va asociada al modelo del mismo nombre para la descripción de sistemas distribuidos de tiempo real, introducido en la Sección 1.4. MAST constituye un conjunto de herramientas programadas en lenguaje Ada que permiten modelar, analizar y optimizar sistemas de tiempo real tanto monoprocesadores, como multiprocesadores y distribuidos. En el ámbito de los sistemas distribuidos, MAST integra entre otras las técnicas de análisis y asignación de parámetros de planificación que se han introducido previamente para FP, GC-EDF y LC-EDF. Una descripción más detallada de la herramienta de análisis y optimización de MAST se llevará a cabo en la Sección 2.5.

El entorno MAST se lleva desarrollando desde 2001 en el grupo de Computadores y Tiempo Real de la Universidad de Cantabria, y se distribuye como *software* libre [MASTh] con licencia GPL. Esta licencia facilita, e incluso alienta, a que cualquier

usuario pueda adaptar y añadir funcionalidades dentro de MAST, de forma que con el tiempo la herramienta evolucione, adaptándose a las necesidades de los investigadores y desarrolladores de sistemas de tiempo real. En conjunto ofrece una colección de las técnicas más avanzadas de análisis de planificabilidad y optimización aplicables a sistemas distribuidos, que unido a su accesibilidad, hacen de MAST el entorno idóneo para el desarrollo de esta tesis.

1.10 Objetivos

A lo largo de este capítulo se han introducido los conceptos fundamentales que rigen los sistemas de tiempo real distribuidos gobernados por eventos, y las políticas de planificación más utilizadas. Se presentaron además dos problemas fundamentales que afectan al comportamiento de este tipo de sistemas: el análisis de planificabilidad para determinar si un sistema cumplirá los requisitos temporales impuestos, y la asignación de parámetros de planificación que permitirá hacer los sistemas planificables. En la literatura de sistemas de tiempo real se han ido proponiendo multitud de técnicas que afrontan estos dos problemas. En esta introducción se presentó una selección representativa de ellas con diferentes niveles de complejidad y calidad en sus resultados.

En general, los esfuerzos de los investigadores en el campo de los sistemas distribuidos de tiempo real se han centrado en ampliar cada vez más su cuerpo teórico, generalizando los conceptos ya asentados de sistemas monoprocesadores. Sin embargo, pese a todos los avances en la comprensión de este tipo de sistemas que se han venido consiguiendo durante los últimos años, identificamos dos áreas en las que aún existen carencias, que creemos son importantes de cara a un mejor entendimiento y soporte de los sistemas distribuidos de tiempo real:

1. Con objeto de facilitar el análisis de los sistemas distribuidos, se suele asumir que todos los recursos procesadores del sistema son planificados por la misma política de planificación [SHA04]. Esta visión es la que adopta por ejemplo el análisis holístico para sistemas distribuidos FP, y todas las técnicas que derivan de él. Sin embargo, esta visión se puede ampliar para poder modelar sistemas más realistas en los que los diferentes recursos procesadores pueden ser de naturaleza heterogénea no sólo en cuanto a los procesadores o redes que los componen, sino también en las políticas con las que se planifican. Las técnicas existentes hasta la elaboración de esta tesis no permiten el análisis y optimización global de sistemas distribuidos en los que los recursos procesadores usan diferentes políticas de planificación. Eliminar esta restricción permitiría analizar y optimizar un conjunto más amplio y realista de sistemas distribuidos.
2. Otra carencia observada en el cuerpo teórico de los sistemas distribuidos de tiempo real hace referencia a la falta de un marco global en el que comparar el rendimiento de las diferentes técnicas de análisis y optimización. Los diferentes trabajos en los que se presentan cada una de estas técnicas suelen venir acompañados de un estudio de rendimiento en el que se comparan normalmente con alguna de las técnicas precedentes. Sin embargo, estos estudios suelen tener un carácter limitado tanto en la variedad de técnicas estudiadas, como en el abanico de situaciones probadas. Como consecuencia, un diseñador de sistemas distribuidos de tiempo real carece de una noción global de las diferencias en el rendimiento real de las diferentes políticas de

planificación, técnicas de optimización, o de las técnicas de análisis. Poseer dicho conocimiento sería de gran utilidad a la hora de tomar decisiones en la etapa de diseño del sistema.

Una vez centrado el ámbito general de trabajo de esta tesis, pasamos a plantear los objetivos específicos que se van a desarrollar.

1.10.1 Soporte de sistemas distribuidos heterogéneos FP+EDF

En primer lugar abordaremos el soporte de sistemas distribuidos en el que se elimina la restricción de que todos los recursos procesadores deban ser planificados por la misma política. A este tipo de sistemas los denominaremos sistemas distribuidos heterogéneos y permitirán el uso de las políticas FP y EDF.

Mientras que la política FP continúa siendo la más utilizada en procesadores y redes, empieza a haber una mayor aceptación de la política EDF, dada su mayor capacidad de planificación, y la relativización de sus aspectos negativos respecto de FP [BUT05], lo que hace prever que el uso de EDF pueda aumentar en el futuro. Mientras que la planificación EDF en procesadores ha sido ampliamente estudiada, su aplicación en redes de comunicación es aún limitada. Por este motivo, el uso de EDF en sistemas distribuidos puede estar limitado a los procesadores, mientras que las redes de comunicación, o cualquier otro recurso procesador heredado, podría seguir utilizando planificadores FP.

Por lo tanto, como primer objetivo de esta tesis nos plantearemos el soporte de los sistemas distribuidos heterogéneos en el que coexisten recursos procesadores FP y EDF. Este soporte se dará tanto en el análisis de planificabilidad, como en la asignación de parámetros de planificación. Los nuevos métodos que se presentarán serán integrados dentro de la herramienta de análisis y optimización de MAST, puesto que ya integra un rico soporte de sistemas distribuidos, tanto en modelado, análisis, y asignación de parámetros de planificación, además de estar bajo el amparo de la licencia GPL, que facilita este tipo de modificaciones.

1.10.2 Uso de un supercomputador para la evaluación de técnicas de análisis y optimización de sistemas distribuidos de tiempo real

Mientras que el comportamiento de los sistemas monoprocesadores y multiprocesadores ha sido estudiado extensamente [BUT05] [DAV11] [BER09] [BAR08], no existe un cuerpo de conocimiento similar para sistemas distribuidos. Esta carencia es especialmente importante en el caso de los sistemas distribuidos EDF, para los cuales en este capítulo introductorio formalizamos dos subtipos de planificación en función de la disponibilidad de sincronización entre los relojes, LC-EDF y GC-EDF. Esta distinción no había sido planteada hasta el momento

Así pues, el segundo objetivo de esta tesis es el de ayudar a completar el conocimiento sobre el comportamiento de los sistemas distribuidos de tiempo real, estudiando el rendimiento de las diferentes políticas de planificación a través de sus técnicas de análisis y asignación de parámetros de planificación asociadas, con el objetivo de contrastar y

extender las conclusiones incluidas en los trabajos en las que fueron presentadas. Se hará especial hincapié en comprobar la influencia en la planificabilidad de disponer de relojes sincronizados en los sistemas distribuidos EDF.

Para la consecución de este objetivo introduciremos el uso de un supercomputador para la evaluación de las técnicas bajo estudio, gracias al cual podremos llevar a cabo evaluaciones más completas que las que se podrían efectuar en un computador convencional. El estudio se realizará sobre conjuntos sintéticos y masivos de ejemplos que abarcarán sistemas con una gran variedad de características arquitecturales (números de recursos procesadores, flujos e2e, actividades, etc.), temporales (diferentes relaciones periodo/plazo, cargas computacionales variables, etc.), o de la plataforma (políticas de planificación, sincronización de relojes, etc.).

1.10.3 Estudio de la influencia del *jitter* en sistemas distribuidos de tiempo real

Tal como hemos presentado en este capítulo, el *jitter* es un efecto inherente a los flujos e2e, que además para sistemas distribuidos planificados por FP se demuestra que induce un efecto negativo en la planificabilidad de los mismos [GUT96]. Sin embargo, no existe un estudio similar de los efectos del *jitter* sobre sistemas distribuidos planificados por EDF, ya sea con sincronización de relojes o sin ella.

En consecuencia, aprovechando la infraestructura de evaluación mediante un supercomputador planteada en el punto anterior, efectuaremos un estudio que evalúe el efecto del *jitter* en sistemas distribuidos EDF. Con objeto de aportar unos resultados completos, también se incluirán los sistemas planificados por FP en este estudio, de forma que puedan compararse con los resultados obtenidos para EDF para los conjuntos masivos de sistemas que se generarán.

1.10.4 Optimización de la asignación de plazos locales de planificación en sistemas distribuidos LC-EDF

En este capítulo introductorio se ha presentado HOSDA como un algoritmo de asignación de plazos de planificación que puede operar tanto con planificación LC-EDF como con planificación GC-EDF. Como parte del desarrollo del segundo objetivo planteado, se observó que las técnicas de asignación de plazos locales de planificación para LC-EDF obtenían un rendimiento especialmente pobre en comparación con FP y GC-EDF.

Estos resultados nos motivaron a plantear un nuevo objetivo específico en esta tesis: profundizar más en el comportamiento de las técnicas aplicables a LC-EDF, y el estudio de nuevas técnicas de asignación de plazos de planificación para LC-EDF que puedan mejorar los resultados de las existentes.

1.11 Organización de la memoria

La memoria de esta tesis queda organizada de la siguiente manera. En el **Capítulo 2** se presentan con más detalle las características de las técnicas de análisis y asignación de parámetros de planificación que se han presentado en este capítulo introductorio, y que serán utilizadas a lo largo de esta tesis. El capítulo finalizará con una descripción más detallada de las herramientas de análisis y optimización del entorno MAST, en el que se implementan las técnicas previamente detalladas.

En el **Capítulo 3** se propone un método con el que combinar las diferentes técnicas de análisis y asignación de parámetros de planificación con el fin de soportar sistemas heterogéneos en los que coexistan recursos procesadores FP y EDF. Se detalla además su implementación dentro del entorno de herramientas MAST.

En la **Capítulo 4** se aborda el desarrollo de la herramienta GEN4MAST, diseñada para generar sistemas de test que permitan llevar a cabo estudios masivos de la planificabilidad en sistemas distribuidos mediante el uso de un supercomputador. Se incluye además el estudio de la influencia de los diferentes métodos de generación de carga que implementa GEN4MAST en los resultados obtenidos al aplicar las técnicas de análisis y optimización. En los capítulos sucesivos se hace uso extensivo de GEN4MAST.

En el **Capítulo 5** se lleva a cabo un estudio comparativo del rendimiento de las diferentes técnicas de análisis de planificabilidad para sistemas distribuidos. El estudio se realiza para las tres políticas de planificación de sistemas distribuidos disponibles en MAST: FP, GC-EDF y LC-EDF.

En el **Capítulo 6** se compara el rendimiento de las técnicas de asignación de parámetros de planificación en sistemas distribuidos implementadas en MAST. En este capítulo se identifica la anomalía en el rendimiento de las técnicas de asignación de plazos para LC-EDF cuyo estudio se incorporó a los objetivos de la tesis. Esta anomalía se estudia con mayor profundidad en el **Capítulo 7**, en el que también se proponen nuevas técnicas de asignación de plazos de planificación con objeto de mejorar el rendimiento de LC-EDF.

El **Capítulo 8** se dedica al estudio de la influencia de la eliminación del *jitter* en sistemas distribuidos planificados con las tres políticas objeto de estudio en esta tesis: FP, GC-EDF y LC-EDF. El estudio se lleva a cabo mediante la comparación de rendimiento en situaciones libres de *jitter* de las diferentes técnicas de análisis y asignación de parámetros de planificación.

En el **Capítulo 9** se aportan las conclusiones de este trabajo, y también se identifican posibles caminos a seguir en futuras investigaciones. Para finalizar, se incluye la bibliografía utilizada en esta tesis.

2 Técnicas y Herramientas Utilizadas

Para la consecución de los objetivos de esta tesis se utilizan una serie de técnicas de análisis de planificabilidad y asignación de parámetros de planificación. El uso que haremos de estas técnicas se puede agrupar en dos ámbitos diferenciados:

1. Integración de las técnicas para dar soporte a sistemas distribuidos heterogéneos, en los que cada recurso procesador puede tener una política de planificación distinta (FP o EDF).
2. Estudio comparativo a gran escala del rendimiento tanto de las técnicas de análisis como de las de asignación de parámetros de planificación.

Antes de afrontar estos objetivos es necesario conocer los detalles del funcionamiento de estas técnicas, así como del entorno de herramientas MAST en el que están implementadas. Éste es el objetivo del presente capítulo.

El capítulo queda organizado como se describe a continuación. Las Secciones 2.1 y 2.2 describen las técnicas de análisis de planificabilidad para sistemas distribuidos FP y EDF respectivamente. Las Secciones 2.3 y 2.4 tratan sobre las técnicas de asignación de prioridades fijas y plazos de planificación respectivamente. Todas las técnicas que aquí se describen están disponibles en el conjunto de herramientas MAST para el modelado, análisis y optimización de sistemas de tiempo real que se utiliza en esta tesis. Una descripción más detallada de esta herramienta se lleva a cabo en la sección 2.5.

2.1 Análisis de planificabilidad en sistemas distribuidos FP

Las técnicas de análisis de planificabilidad para sistemas distribuidos FP calculan los tiempos de respuesta de peor caso de las actividades que se ejecutan. Todas estas actividades deben tener asignada una prioridad fija para poder ser planificados. Si los tiempos de respuesta de peor caso calculados son menores o iguales que los plazos impuestos en el sistema, se puede determinar que el sistema es planificable en cualquier circunstancia. Las técnicas de referencia para desempeñar esta labor, y que van a ser utilizadas en este trabajo, son las que identificamos como:

- Análisis holístico para FP.
- Análisis basado en *offsets* de fuerza bruta para FP.
- Análisis basado en *offsets* para FP.
- Análisis basado en *offsets* con relaciones de precedencia para FP.
- Análisis basado en *offsets slanted* para FP.

2.1.1 Análisis holístico para FP

La primera aproximación al análisis de planificabilidad de sistemas distribuidos FP es el denominado análisis holístico (*holistic analysis*) [TIN94], llamado así porque toma una visión holística del sistema, en la que se analiza el sistema distribuido en su conjunto, aunque considerando a cada recurso procesador de manera independiente. Este tratamiento del sistema da lugar a resultados inherentemente pesimistas, construyendo una situación que es peor que el peor caso real, al no tenerse en cuenta la dependencia entre las activaciones de las diferentes actividades dentro de un mismo flujo e2e.

El análisis holístico se basa en la adaptación del test exacto para su aplicación en flujos e2e. En el test exacto, el tiempo de respuesta de peor caso de cada actividad se calcula creando el instante crítico que produzca el periodo de ocupación de peor caso de cada actividad. El periodo de ocupación de una actividad τ_{ab} es un intervalo de tiempo en el que el recurso procesador en el que reside está ocupado ejecutando $\tau_{ab,}$ o actividades de prioridad igual o superior. En sistemas FP con actividades sin *offsets* se demuestra que el instante crítico se construye activando todas las actividades del recurso procesador en el mismo instante [LIU73]. Bajo estas circunstancias, asumiendo además que los plazos no son mayores que los periodos y que las actividades son perfectamente periódicas, el tiempo de respuesta de peor caso de τ_{ab} se calcula como el menor tiempo de respuesta que cumple la siguiente ecuación:

$$R_{ab} = B_{ab} + C_{ab} + \sum_{\forall i \, \mathrm{j} \in \mathrm{hp}(ab)} \left\lceil \frac{R_{ab}}{T_i} \right\rceil C_{ij} \qquad (2\text{-}1)$$

donde B_{ab} es el bloqueo máximo que puede sufrir la actividad τ_{ab} por otras de menor prioridad, y *hp(ab)* representa el conjunto de actividades que residen en el mismo recurso procesador que τ_{ab}, y poseen mayor o igual prioridad.

La ecuación (2-1) se expande para soportar dos situaciones que deben ser tenidas en cuenta a la hora de analizar flujos e2e. En primer lugar Tindell *et al* [TIN94d] eliminan la

restricción de que los plazos sean menores que los periodos. Cuando los plazos son mayores que los periodos, el cálculo del tiempo de respuesta de peor caso de cada actividad debe tener en cuenta que dentro del periodo de ocupación puede aparecer más de una activación de la propia actividad bajo análisis. En segundo lugar, Audsley *et al* [AUD93] actualizan el análisis para soportar actividades con *jitter* de activación. Con estas dos extensiones, Tindell y Clark [TIN94] definen el análisis holístico para sistemas distribuidos FP.

En el análisis holístico, el tiempo de respuesta de peor caso de una actividad τ_{ab} se calcula a partir de la ecuación iterativa siguiente, que utiliza la formulación utilizada en [PAL97], en la que la primera activación de las actividades es la activación número 1 (en la nomenclatura original de Tindell *et al* la numeración de las actividades comienza en 0):

$$w_{ab}^{n+1}(p) = pC_{ab} + B_{ab} + \sum_{\forall ij \in hp(ab)} \left\lceil \frac{J_{ij} + w_{ab}^{n}(p)}{T_i} \right\rceil C_{ij} \qquad (2\text{-}2)$$

donde:

- $w_{ab}(p)$: Valor de convergencia de la ecuación $\left(w_{ab}^{n+1}(p) = w_{ab}^{n}(p)\right)$, que representa el tiempo de finalización de peor caso de la *p*-ésima activación de la actividad τ_{ab} desde el instante crítico. Se puede inicializar con un valor $w_{ab}^{0}(p) = pC_{ab}$.
- *hp(ab)*: Conjunto de actividades que pueden expulsar a τ_{ab}, esto es, las actividades que residen en el mismo recurso procesador y que poseen una prioridad mayor o igual.
- B_{ab}: Bloqueo máximo de la actividad τ_{ab}.

La ecuación (2-2) se aplica de forma sucesiva para las diferentes activaciones de τ_{ab}, p=[1,2,3,…]. Se deben probar las activaciones que cumplan con la siguiente ecuación:

$$w_{ab}(p) \leq pT_{ab} \qquad (2\text{-}3)$$

Una vez calculados los tiempos de finalización de las sucesivas activaciones, el tiempo de respuesta de peor caso de la actividad τ_{ab} se calcula según la siguiente ecuación:

$$R_{ab} = \max_{p=1,2,3,\ldots} \left(w_{ab}(p) - (p-1)T_{ab}\right) + J_{ab} \qquad (2\text{-}4)$$

Se comprueba que el tiempo de respuesta de peor caso depende del *jitter*. El *jitter* a su vez depende del tiempo de respuesta de peor caso, ya que es la variabilidad en el instante de finalización de la actividad precedente en el flujo e2e. Esta variabilidad se representa con la siguiente ecuación, para una actividad τ_{ij} dada:

$$J_{ij} = R_{ij-1} - Rb_{ij-1} \qquad (2\text{-}5)$$

La interdependencia del *jitter* con el tiempo de respuesta se soluciona aplicando el cálculo del *jitter* y los tiempos de respuesta de forma iterativa [TIN94]. En la primera iteración el término de *jitter* se establece a valor 0, y se procede al cálculo de los tiempos de respuesta de peor caso iniciales. Con estos tiempos de respuesta se calculan los nuevos *jitters* utilizando la ecuación (2-5), y se vuelven a calcular los tiempos de respuesta de peor caso. Este proceso se repite de forma sucesiva hasta que los valores de *jitter* y

tiempos de respuesta de peor caso converjan. Las características de monotonicidad del tiempo de respuesta y el *jitter* aseguran la convergencia [PAL97].

Poseer una estimación ajustada de la cota inferior de los tiempos de respuesta, esto es, de los tiempos de respuesta de mejor caso, reduciría los términos de *jitter* en el análisis, y por lo tanto, se obtendrían igualmente tiempos de respuesta de peor caso más ajustados. Si se dispone de una estimación de los tiempos de ejecución de mejor caso de las actividades (Cb_{ij}), una manera sencilla de calcular una estimación del tiempo de respuesta de mejor caso de una actividad es como la suma de los tiempos de ejecución de mejor caso de las actividades precedentes en el flujo e2e. Utilizando este criterio, una estimación del *jitter* de una actividad τ_{ij} dada se puede calcular con la siguiente ecuación:

$$J_{ij} = R_{ij-1} - \sum_{k=1}^{j-1} Cb_{ik} \qquad (2\text{-}6)$$

2.1.2 Técnicas basadas en *offsets* para FP

El análisis holístico calcula los tiempos de respuesta de peor caso de las actividades asumiendo que la peor situación surge al activar todas las actividades con prioridad superior en el mismo instante (instante crítico). Esta idea es pesimista en los sistemas distribuidos, ya que el orden de precedencia en la ejecución de las actividades puede impedir que en la realidad ocurra tal instante crítico.

Las técnicas de análisis de planificabilidad basadas en *offsets* se crean con el objetivo de reducir el pesimismo de la técnica holística en el cálculo de los tiempos de respuesta de peor caso. Cuando las actividades poseen *offsets*, hay que tener en cuenta que el instante crítico puede no incluir la activación simultánea de todas las actividades con prioridad superior, ya que los *offsets* pueden hacer imposible que ciertas actividades puedan activarse a la vez. Mediante esta noción se pueden encontrar instantes críticos menos pesimistas.

Los análisis basados en *offsets* estudian la contribución de cada actividad τ_{ij} sobre la respuesta de peor caso del resto de actividades de menor prioridad en el mismo recurso procesador, dividiendo las activaciones en tres posibles grupos [PAL98]:

- *Set 0*: Activaciones que ocurren antes del instante crítico, y cuyo *jitter* no puede retrasarlas hasta el periodo de ocupación. Por lo tanto, las activaciones englobadas en este grupo no se tienen en cuenta en el análisis.
- *Set 1*: Activaciones que ocurren antes o en el instante crítico, y cuyo *jitter* puede retrasarlas hasta justo el instante crítico.
- *Set 2*: Activaciones que ocurren después del instante crítico.

Una vez encuadradas las sucesivas activaciones de τ_{ij} en alguno de los tres grupos, el cálculo de los términos de *jitter* que provoca la contribución de peor caso de la actividad τ_{ij} en las actividades de menor prioridad en el mismo recurso procesador se lleva a cabo mediante el siguiente teorema:

> **Teorema 1 en** [PAL98] (adaptado de [TIN94b])**:** La contribución de peor caso de la actividad τ_{ij} ocurre cuando las activaciones del *Set 1* se ven retrasadas por una cantidad de *jitter* tal que las hace coincidir con el instante crítico, y cuando las activaciones del *Set 2* no experimentan ningún retraso.

Mediante el Teorema 1, Tindell [TIN94b] define una serie de ecuaciones con las que se pueden calcular de forma exacta los tiempos de respuesta cuando se considera que los *offsets* son constantes. El problema principal de este análisis es que se vuelve intratable para cualquier sistema, salvo que sea muy sencillo, ya que la actividad que genera la contribución de peor caso sobre un flujo dado es desconocida, y por lo tanto, se deben comprobar todas las posibles combinaciones de actividades, hasta encontrar la que produce el peor caso.

En el mismo trabajo [TIN94b], Tindell propone una aproximación a este análisis exacto que produce resultados ligeramente pesimistas. Esta aproximación consiste en que, para cada flujo e2e, se define una función que es una cota superior de todas las contribuciones de peor caso considerando que cada una de las actividades inicia el instante crítico. Con esta aproximación se reduce drásticamente el número de casos a tratar.

Los análisis basados en *offsets* de Tindell, tanto el exacto como el aproximado, tienen como requisito de funcionamiento que los *offsets* deben ser estáticos. Palencia y González Harbour [PAL98] extendieron estos análisis para darle a los *offsets* la capacidad de ser dinámicos. Con esta modificación se permite el análisis de sistemas distribuidos mediante las siguientes equivalencias en las que se hace uso de los tiempos de respuesta para calcular los *offsets* y *jitters* equivalentes:

$$\phi'_{ij} = Rb_{ij-1} \tag{2-7}$$

$$J'_{ij} = R_{ij-1} - Rb_{ij-1} \tag{2-8}$$

donde ϕ'_{ij} y J'_{ij} representan respectivamente el *offset* y el *jitter* equivalente de la actividad τ_{ij}. Para el cálculo del *jitter* (ecuación (2-8)) se puede utilizar una estimación del mejor caso como la descrita en la ecuación (2-6).

Al análisis exacto de Tindell modificado para dar soporte a sistemas distribuidos mediante *offsets* dinámicos lo llamamos **análisis basado en o*ffsets* de fuerza bruta**, (*offset based brute force analysis*), y al análisis aproximado con soporte para sistemas distribuidos lo llamamos **análisis basado en o*ffsets*** (*offset based analysis*).

Mäki-Turja y Nolin proponen en [MAK08] una mejora del análisis basado en *offsets*. Observan que las técnicas previas utilizan una función techo para calcular la interferencia de una actividad en las otras de menor prioridad, provocando una función escalonada que sobreestima dicha interferencia. En base a esta observación proponen utilizar una función distinta, que sustituye estos escalones por pendientes, y que obtiene una estimación menos pesimista. A este análisis mejorado lo llamamos **análisis basado en *offsets slanted*** (*offset based slanted analysis)*.

Estas técnicas basadas en *offsets* soportan el análisis de flujos e2e distribuidos, pero tienen en cuenta las relaciones de precedencia de la ejecución de las actividades sólo de una forma indirecta, utilizando los valores de los tiempos de respuesta intermedios calculados. En [PAL99], Palencia y González Harbour proponen una nueva técnica en la que estiman las relaciones de precedencia de una manera más directa, permitiendo así reducir el pesimismo del análisis. Este análisis se basa en el siguiente lema:

Lema 5-1 en [PAL99]: Sean τ_{ij} y τ_{ij+1} dos actividades consecutivas en un mismo flujo e2e, de tal forma que τ_{ij+1} se activa cuando τ_{ij} finaliza su ejecución. Si ambas actividades están localizadas en el mismo procesador, no puede haber un periodo de

ocupación con nivel de prioridad $pr = \min\left(prio(\tau_{ij}), prio\left(\tau_{ij+1}\right)\right)$ o inferior, cuyo comienzo coincida con la activación de τ_{ij+1} .

Mediante este lema se reduce el número de casos a considerar en el análisis, eliminando los que potencialmente podrían ser más pesimistas. Al análisis aproximado, añadiéndole esta consideración sobre la precedencia de las actividades, lo llamamos a**nálisis basado en o*ffsets* con relaciones de precedencias** (*offset based with precedence relationships)*.

2.2 Análisis de planificabilidad en sistemas distribuidos EDF

A continuación se describirán las técnicas de análisis de referencia para sistemas distribuidos EDF. Las técnicas que trataremos las identificamos cómo:

- Análisis holístico para GC-EDF.
- Análisis basado en *offsets* para GC-EDF.
- Análisis holístico para LC-EDF.

2.2.1 Análisis holístico para GC-EDF

Spuri [SPU96] propuso un análisis de tiempos de respuesta para actividades planificadas por EDF con plazos arbitrarios y afectadas por *jitter*. Aunque Spuri no lo especifique explícitamente, la inclusión del *jitter* permite utilizar su formulación para el análisis de flujos e2e, de manera similar a como Tindell propuso el análisis holístico para FP, asumiendo además que todas las actividades en el sistema son independientes. Adicionalmente, el modelo de sistema del análisis de Spuri se asemeja al de un sistema GC-EDF: los plazos con los que se planifican las actividades toman como referencia la activación de la propia actividad menos su *jitter* de entrada. Por todas estas razones, al análisis de Spuri también lo denominaremos **análisis holístico para GC-EDF** (*offset based analysis for GC-EDF*). Su formulación fue modificada por Palencia y González Harbour [PAL03], ofreciendo una nomenclatura más cercana a la utilizada en MAST.

En el análisis holístico para FP, el tiempo de respuesta de peor caso de una actividad τ_{ij} se encuentra después del instante crítico, cuando la activación de τ_{ij} coincide con la activación de todas las actividades de prioridad mayor después de haber experimentado un retraso (*jitter*) máximo. En esta situación, el instante crítico coincide con el comienzo del periodo de ocupación. En EDF esta propiedad no es cierta, pero el concepto del periodo de ocupación sigue siendo útil. En el análisis holístico para GC-EDF Se utiliza el siguiente teorema para encontrar el instante crítico de una actividad:

> **Teorema 1 en [PAL03]:** El tiempo de respuesta de peor caso de la actividad τ_{ij} se encuentra en un periodo de ocupación en el que el resto de las actividades del recurso procesador (y quizás la propia actividad bajo análisis) se activan simultáneamente en el comienzo del periodo de ocupación, después de haber experimentado su retraso (*jitter*) máximo.

Cabe notar que, al contrario que para el resto de actividades, activar la actividad bajo análisis al comienzo del periodo de ocupación puede no producir su tiempo de respuesta de peor caso.

A partir de las condiciones establecidas en el Teorema 1, se definen las ecuaciones con las que calcular los tiempos de respuesta de peor caso de todas las actividades del sistema. Siguiendo la formulación de [PAL03], la contribución de peor caso de una actividad τ_{ij} al periodo de ocupación de la actividad τ_{ab} bajo análisis es:

$$W_{ij}(t,D) = \min\left(\left\lceil \frac{t+J_{ij}}{T_i} \right\rceil, \left\lfloor \frac{J_{ij}+D-SD_{ij}}{T_i} \right\rfloor + 1\right)_0 C_{ij} \qquad (2\text{-}9)$$

donde D es el plazo absoluto de τ_{ab}, y t es la longitud del periodo de ocupación.

El tiempo de finalización de la p-ésima activación de la actividad bajo análisis τ_{ab}, si su primera activación ocurrió en el instante A, se calcula con la siguiente ecuación:

$$w_{ab}^{A}(p) = pC_{ab} + \sum_{(\forall ij \neq ab) \in PR_{ab}} W_{ij}\left(w_{ab}^{A}(p), D^{A}(p)\right) \qquad (2\text{-}10)$$

donde PR_{ab} es el recurso procesador en el que reside τ_{ab}, y $D^A(p)$ el plazo absoluto de su p-ésima activación:

$$D^{A}(p) = A - J_{ab} + (p-1)T_{ab} + SD_{ab} \qquad (2\text{-}11)$$

El tiempo de respuesta de la p-ésima activación de τ_{ab} cuya primera activación ocurrió en A se calcula como:

$$R_{ab}^{A}(p) = w_{ab}^{A}(p) - A + J_{ab} - (p-1)T_{ab} \qquad (2\text{-}12)$$

Los valores de A que se deben probar son:

$$A = \Psi_x - \left[(p-1)T_{ab} - J_{ab} + SD_{ab}\right] \qquad (2\text{-}13)$$

donde Ψ_x es cada elemento dentro del conjunto Ψ^* en el que se almacenan todos los instantes en los que los plazos absolutos de τ_{ab} coinciden con alguno de los plazos absolutos del resto de actividades en el periodo de ocupación, ignorando los valores que provocan que A se salgan del rango $[0,T_{ab}]$, ya que estos no forman parte de la p-ésima activación de τ_{ab}:

$$\Psi = \cup\left\{(p-1)T_{ij} - J_{ij} + SD_{ij}\right\} \qquad \forall p = 1...\left\lceil \frac{L+J_{ij}}{T_{ij}} \right\rceil \qquad (2\text{-}14)$$

$$\Psi^* = \left\{\Psi_x \in \Psi \mid (p-1)T_{ab} - J_{ab} + SD_{ab} \le \Psi_x \le pT_{ab} - J_{ab} + SD_{ab}\right\} \qquad (2\text{-}15)$$

donde L corresponde con el periodo de ocupación más largo, calculado como:

$$L = \sum_{\forall ij \in PR_{ab}} \left\lceil \frac{L+J_{ij}}{T_{ij}} \right\rceil C_{ij} \qquad (2\text{-}16)$$

El tiempo de respuesta de peor caso de la actividad τ_{ab} es el peor de todos los tiempos de respuesta calculados:

$$R_{ab} = \max\left(R_{ab}^{A}(p)\right) \quad \forall p = 1...\left\lceil \frac{L - J_{ab}}{T_{ab}} \right\rceil, \ \forall A \in \Psi^{*} \qquad (2\text{-}17)$$

La dependencia del *jitter* con el tiempo de respuesta de peor caso se resuelve de manera iterativa, de igual forma que en el análisis holístico para FP.

2.2.2 Análisis basado en *offsets* para GC-EDF

Al igual que ocurría en FP, la visión holística del sistema distribuido EDF añade pesimismo al cálculo de los tiempos de respuesta de peor caso. Con el fin de reducir este pesimismo, y de forma similar a lo anteriormente descrito para sistemas FP, se define una nueva técnica basada en los *offsets* de las actividades.

Para una actividad τ_{ab} bajo análisis, el análisis se basa en crear el instante crítico que produce el periodo de ocupación de peor caso, de manera similar al proceso que se sigue en el análisis holístico, con la salvedad de que con *offsets* se debe tener en cuenta que el periodo de ocupación puede no incluir la activación simultánea de todas las actividades, es decir, la existencia de *offsets* puede hacer imposible que algún conjunto de actividades puedan activarse simultáneamente. El periodo de ocupación de la actividad τ_{ab} se construye según el siguiente teorema.

> **Teorema 2 en [PAL03]:** La contribución de peor caso de un flujo Γ_i en el tiempo de respuesta de la actividad τ_{ab} se obtiene cuando la primera activación de alguna actividad τ_{ik} que ocurra en el periodo de ocupación coincida con el comienzo del periodo de ocupación, después de haber experimentado un retraso (*jitter)* máximo.

Se deben probar todos los posibles periodos de ocupación creados por combinación, haciendo coincidir cada actividad de cada flujo e2e con el comienzo del periodo de ocupación. Para calcular la contribución de cada una de estas actividades, categorizamos sus activaciones en alguno de los tres grupos descritos en la sección 2.1.2 para FP. Una vez clasificados en un grupo, la contribución de peor caso de una actividad τ_{ij} sobre τ_{ab} con plazo absoluto D se obtiene a partir del siguiente teorema:

> **Teorema 3 en [PAL03]:** Dada una activación de τ_{ab} con plazo absoluto D, y una fase Φ (distancia entre el flujo Γ_i y el comienzo del periodo de ocupación), la contribución de peor caso de τ_{ij} sobre el tiempo de respuesta de τ_{ab} ocurre cuando las activaciones del *Set 1* poseen una cantidad de retraso (*jitter*) tal que todas se activan al comienzo del periodo de ocupación, y cuando las activaciones del *Set 2* tienen un retraso (*jitter*) nulo.

Bajo las condiciones del Teorema 3, en [PAL03] se describen las ecuaciones para obtener el tiempo de respuesta de peor caso de las actividades del sistema. En una primera versión del análisis, se deben comprobar todos los periodos de ocupación posibles, y seleccionar aquel en el que los tiempos de respuesta han sido mayores. Cada periodo de ocupación se genera seleccionando en cada flujo e2e cuál de sus actividades es la que se activó en el comienzo del periodo de ocupación. Salvo para sistemas simples, el elevado número de combinaciones a probar hacen que esta búsqueda del peor periodo de ocupación sea computacionalmente intratable. Por ello, y siguiendo los mismos métodos descritos para FP en [PAL98], en [PAL03] se establece una cota superior a la interferencia de las actividades del flujo Γ_i sobre el periodo de ocupación. Con esta aproximación se obtienen resultados pesimistas, pero el número de casos a comprobar

ahora sólo tiene una dependencia polinomial con el número de actividades en el sistema. A esta aproximación la llamamos **análisis basado en *offsets* para GC-EDF** (*offsets based analysis for GC-EDF)*.

2.2.3 Análisis holístico para LC-EDF

En [RIV10] se adaptó el análisis holístico para GC-EDF para dar soporte a sistemas distribuidos EDF con relojes locales (LC-EDF). Ambas técnicas se basan en el mismo concepto de instante crítico y periodo de ocupación, adaptando la formulación para tener en cuenta que en LC-EDF los plazos de planificación hacen referencia a la activación de la propia actividad.

Este análisis se basa en la aplicación del siguiente teorema, que es similar al Teorema 1 de [PAL03] para GC-EDF:

Teorema 1 en [RIV10]: El tiempo de respuesta de peor caso de la actividad τ_{ab} se encuentra en un periodo de ocupación en el que cada actividad τ_{ij} (distinta a τ_{ab}) y situada en el mismo recurso procesador, se planifica de tal manera que su primera activación ocurrida dentro del periodo de ocupación sucede justo al comienzo de éste, después de haber experimentado su retraso máximo. Es decir, el comienzo del periodo de dicha actividad ocurre J_{ij} unidades de tiempo antes que el comienzo del periodo de ocupación, y además, el resto de actividades se activan con el retraso (*jitter*) mínimo que les hace comenzar dentro del periodo de ocupación.

A partir de las condiciones del Teorema 1 en [RIV10], la contribución de peor caso de una actividad τ_{ij} a un periodo de ocupación de longitud l cuando el plazo absoluto de la actividad bajo análisis τ_{ab} ocurre en el instante D es:

$$W_{ij}(l,D) = \min\left(P_l, P_D\right) C_{ij} \tag{2-18}$$

donde P_l es el número de activaciones de τ_{ij} en el periodo de ocupación:

$$P_l = \left\lceil \frac{l + J_{ij}}{T_i} \right\rceil \tag{2-19}$$

y P_D es el número de activaciones con plazo absoluto anterior o igual a D:

$$P_D = \begin{cases} 0 & D < Sd_{ij} \\ \left\lfloor \dfrac{J_{ij} + D - Sd_{ij}}{T_i} \right\rfloor & D \geq Sd_{ij} \end{cases} \tag{2-20}$$

El desfase de τ_{ab} con respecto al comienzo del periodo de ocupación se calcula teniendo en cuenta que su activación debe estar o bien en el comienzo del periodo de ocupación, o de forma que su plazo coincida con el de alguna otra actividad. El conjunto de instantes en el que los plazos absolutos coinciden son:

$$\Psi_{ij} = \cup\left\{\left(p-1\right)T_{ij} - J_{ij} + Sd_{ij}\right\} \quad \forall p = 1...\left\lceil \frac{L + J_{ij}}{T_{ij}} \right\rceil \tag{2-21}$$

donde L es el la longitud del periodo de ocupación más largo, calculado según la ecuación (2-16).

El conjunto de plazos absolutos de τ_{ab} es:

$$\Psi_{ab} = \cup\{(p-1)T_{ab} + Sd_{ab}\} \quad \forall p = 1..\left\lceil \frac{L}{T_{ab}} \right\rceil \tag{2-22}$$

Por lo tanto, el conjunto de instantes en el que se debe estudiar τ_{ab} es:

$$\Psi = \Psi_{ij} \cup \Psi_{ab} \tag{2-23}$$

Los instantes de activación de τ_{ab} se obtienen restando Sd_{ab} a cada valor de Ψ. El tiempo de finalización de la *p*-ésima activación de $\tau_{ab,}$ cuya primera activación tuvo un plazo $\psi \in \Psi$, se calcula con la siguiente ecuación:

$$w_{ab}^{\psi}(p) = pC_{ab} + \sum_{(\forall ij \neq ab) \in PR_{ab}} W_{ij}\left(w_{ab}^{\psi}(p), \psi\right) \tag{2-24}$$

El tiempo de respuesta de la *p*-ésima activación de $\tau_{ab,}$ cuya primera activación tuvo un plazo $\psi \in \Psi$, se calcula con la siguiente ecuación:

$$R_{ab}^{\psi}(p) = w_{ab}^{\psi}(p) - (\psi - Sd_{ab}) + J_{ab} \tag{2-25}$$

El tiempo de respuesta de peor caso de la actividad τ_{ab} es el peor de todos los tiempos de respuesta calculados:

$$R_{ab} = \max\left(R_{ab}^{\psi}(p)\right) \quad \forall p = 1...\left\lceil \frac{L}{T_{ab}} \right\rceil, \ \forall \psi \in \Psi^{*} \tag{2-26}$$

donde Ψ^{*} es el subconjunto de plazos absolutos incluidos en Ψ que recaen en el periodo actual de la actividad bajo análisis. El resto de plazos absolutos no deben ser tenidos en cuenta ya que corresponden a otra activación distinta (otro valor de *p*). Ψ^{*} se formaliza en la siguiente ecuación:

$$\Psi^{*} = \left\{\psi_x \in \Psi \mid (p-1)T_{ab} + Sd_{ab} \leq \psi_x \leq pT_{ab} + Sd_{ab}\right\} \tag{2-27}$$

2.3 Algoritmo HOPA para la asignación de prioridades fijas

La labor de una técnica de asignación de prioridades fijas en un sistema distribuido es la de asignar a todas las actividades del sistema una prioridad fija que los planificadores utilizarán para ordenar la ejecución de las actividades. Debido al elevado número de combinaciones de actividades y prioridades que pueden probarse, es fácil comprobar que encontrar la asignación óptima es un problema NP-difícil [BUR91][TIN92], y por lo tanto, computacionalmente intratable. Por ello se hace necesario disponer de alguna técnica que encuentre soluciones lo más próximas posibles a la solución óptima, pero que posea una complejidad computacional inferior.

El algoritmo HOPA (*Heuristic Optimized Priority Assignment*) [GUT95] se crea con el propósito de asignar prioridades fijas en sistemas distribuidos, y resulta básico en el desarrollo de esta tesis. Para conseguir optimizar los resultados, y conseguirlos en tiempos razonables, hace uso de uno de los factores más determinantes en el

comportamiento temporal de los sistemas, el tiempo de respuesta de peor caso de cada actividad, por lo que requiere el uso de una técnica de cálculo de dichos tiempos.

El funcionamiento del algoritmo HOPA se basa en el reparto del plazo de principio a fin de los flujos e2e entre las actividades que lo componen, y puede resumirse con el pseudocódigo mostrado en la Figura 2-1, que consta de los pasos que se explican a continuación.

```
Algoritmo HOPA is
begin
   Distribución del plazo de principio a fin entre las actividades
   loop
      Prioridades fijas Deadline Monotonic
      Cálculo de tiempos de respuesta de peor caso
      exit when se cumple algún requisito de parada
      Cálculo de nuevos plazos para cada actividad
   end loop
end HOPA
```

Figura 2-1 Pseudocódigo del algoritmo HOPA

1. Distribución del plazo de principio a fin: El primer paso del algoritmo HOPA consiste en la distribución del plazo de principio a fin de los flujos e2e entre las actividades que los componen. El resultado es que a cada actividad se le asigna una porción de dicho plazo, que llamamos plazo virtual (VD, *Virtual Deadline*). En HOPA, este reparto se hace de forma proporcional a los tiempos de ejecución de cada actividad, de acuerdo a la siguiente fórmula:

$$VD_{ij} = D_i \frac{C_{ij}}{\sum_{k=1}^{N_i} C_{ik}} \tag{2-28}$$

   donde N_i es el número de actividades en el flujo e2e Γ_i.

2. En el siguiente paso, los plazos virtuales se traducen a prioridades fijas. Para ello se utiliza el criterio *Deadline Monotonic*, de forma que un plazo virtual menor se traduce en una prioridad mayor.

3. Con la asignación de prioridades fijas obtenida, se calculan los tiempos de respuesta de peor caso mediante alguna de las técnicas de análisis de planificabilidad disponibles.

4. Una vez calculados los tiempos de respuesta de peor caso, el algoritmo HOPA se detiene cuando se cumple alguna de las siguientes situaciones :

   - ✓ En dos iteraciones consecutivas se ha obtenido la misma asignación de prioridades fijas, ya que en este caso, si se siguiera iterando, se obtendría la misma asignación indefinidamente.
   - ✓ El sistema es planificable.
   - ✓ Se supera un número máximo de iteraciones preestablecidas.

5. Si no se cumple ninguno de los criterios de parada, se procede a calcular nuevos plazos virtuales de acuerdo con la formulación descrita en [GUT95]. Dicha formulación hace uso de los tiempos de respuesta de peor caso calculados, de forma que se intenta asignar menor plazo virtual a aquellas actividades que estén más lejos de la planificabilidad. El cálculo de los nuevos plazos virtuales se puede controlar mediante dos parámetros que controlan el peso que tiene la "distancia" a la planificabilidad en el resultado final de dicho cálculo, tanto para los recursos procesadores (controlado por el parámetro k_r), como para los flujos e2e (controlado por el parámetro k_a). Estos plazos virtuales se ajustan a continuación para que sean conformes al plazo de principio a fin (su suma sea igual al plazo de principio a fin). Una vez que todas las actividades tengan asignado un nuevo plazo virtual, se vuelve al paso 2.

Cuando HOPA encuentra una asignación de prioridades fijas que hace al sistema planificable, el algoritmo tiene la capacidad de optimizar esta solución, de acuerdo con el criterio usado para determinar la bondad de una solución, como por ejemplo el *slack*. Para aprovechar esta característica, se puede establecer un número máximo de sobre-iteraciones a realizar sobre una solución planificable. Para ello, a los criterios de parada ya establecidos en el paso 4 del algoritmo, se añade el hecho de superar este número de sobre-iteraciones de optimización.

2.4 Asignación de plazos de planificación en sistemas distribuidos

En esta sección se van a describir las técnicas de asignación de plazos de planificación para sistemas distribuidos EDF que fueron presentadas en la introducción. Estas son, en orden ascendente de complejidad: UD, ED, PD, NPD, EQS, EQF y HOSDA. El objetivo de todos estos algoritmos es el de asignar a cada actividad un plazo de planificación con el que puedan operar los planificadores EDF, para obtener la planificabilidad del sistema en su conjunto. Tal y como se ha definido en la Sección 1.5.3.1 de esta memoria, en LC-EDF a estos plazos los llamamos plazos locales de planificación (Sd_{ij}), y en GC-EDF plazos globales de planificación (SD_{ij}).

Mientras que el concepto de prioridad fija es claro y utilizado de forma homogénea en los trabajos sobre sistemas de tiempo real, el de plazo de planificación en sistemas distribuidos resulta más ambiguo. Muchos de los algoritmos que se van a describir aquí fueron presentados originalmente sin tener en cuenta la distinción que existe entre lo que representa que un plazo de planificación sea local o global. Es por ello que resulta fundamental definir los criterios que utilizaremos para interpretar una asignación de plazos de planificación como local (para LC-EDF), o global (para GC-EDF).

Los plazos globales de planificación toman como referencia los instantes de activación del flujo e2e en cada una de sus instancias. Por lo tanto, es fácil comprobar que los valores de estos plazos globales deberían aumentar a lo largo del flujo e2e. Adicionalmente, el plazo global de la última actividad de un flujo e2e es por definición el plazo de principio a fin de dicho flujo. Estas dos características establecen el criterio para interpretar una asignación de plazos de planificación como global, que se formaliza en las siguientes dos ecuaciones:

$$SD_{ij} \leq SD_{ij+1} \tag{2-29}$$

$$SD_{iN_i} = D_i \tag{2-30}$$

Los valores de los plazos locales no poseen ninguna característica intrínseca que pueda ayudar a interpretar una asignación dada como local, ya que cualquier asignación de valores podría considerarse válida, incluso una global. Sin embargo, puesto que los plazos locales toman como referencia la activación de su propia actividad, resulta lógico establecer que la suma de los plazos locales en un flujo e2e dado sea igual a su plazo de principio a fin. Este es el criterio que se utiliza normalmente [SER10], y se formaliza en la ecuación (2-31) para plazos locales de planificación. Cuando una asignación de plazos locales de planificación cumple dicha ecuación, decimos que es una asignación local conforme a los plazos de principio a fin.

$$\sum_{j=1}^{N_i} Sd_{ij} = D_i \tag{2-31}$$

Una propiedad de las asignaciones de plazos locales de planificación conformes a los plazos de principio a fin es que pueden ser transformadas en una asignación global mediante la siguiente fórmula:

$$SD_{ij} = \sum_{k=1}^{j} Sd_{ik} \tag{2-32}$$

Cuando un algoritmo de asignación se diseñó para asignar únicamente plazos locales de planificación, y además estos son conformes con el plazo de principio a fin de acuerdo a lo especificado en la ecuación (2-31), se puede utilizar la ecuación (2-32) para la transformación a plazos globales de planificación para GC-EDF.

Cabe notar que las técnicas de análisis para sistemas distribuidos EDF no poseen de forma inherente ningún requisito que especifique qué propiedades deben tener los plazos de planificación. Las reglas de interpretación aquí descritas surgen de la intuición para dar una respuesta sencilla a la pregunta de qué define a una asignación de plazos de planificación como local o global, pero no representan ningún método de clasificación que deba ser cumplido de forma estricta.

2.4.1 *Ultimate Deadline* (UD) y *Effective Deadline* (ED)

Los dos primeros algoritmos que destaca Liu [LIJ00] para asignar plazos de planificación a las actividades de un flujo e2e son UD (*Ultimate Deadline*) y ED (*Effective Deadline*). Pese a que UD y ED son unos algoritmos creados con el objetivo de llevar a cabo en tiempo de ejecución la tarea de asignar plazos de planificación en sistemas de tiempo real no estricto [KAO93], no poseen ningún requisito en su formulación que impida su uso en sistemas con el modelo que utilizamos en este trabajo (tiempo-real estricto, y asignación estática *off-line* de plazos de planificación).

UD es el algoritmo más sencillo de ambos, puesto que simplemente asigna como plazo de planificación el valor del plazo de principio a fin del flujo e2e en el que reside la actividad, tal y como se especifica en la ecuación (2-33).

UD: $$SD_{ij} = D_i \tag{2-33}$$

El algoritmo ED en cambio resta al valor del plazo de principio a fin la suma de los tiempos de ejecución de peor caso de las actividades posteriores en el flujo e2e, tal y como se describe en la ecuación (2-34).

ED:
$$SD_{ij} = D_i - \sum_{k=j+1}^{N_i} C_{ik} \tag{2-34}$$

Tanto UD como ED cumplen con los requisitos descritos con las ecuaciones (2-29) y (2-30) para clasificarlas como asignaciones de plazos globales de planificación.

2.4.2 *Proportional Deadline* (PD) y *Normalized Proportional Deadline* (NPD)

Los otros dos algoritmos que destaca Liu [LIJ00] son PD (*Proportional Deadline*) y NPD (*Normalized Proportional Deadline*).

PD reparte el plazo de principio a fin de forma que el plazo asignado a cada actividad es proporcional a su tiempo de ejecución de peor caso en relación al tiempo de ejecución de peor caso total del flujo e2e. Se formaliza con la siguiente ecuación:

PD:
$$Sd_{ij} = D_i \frac{C_{ij}}{\sum_{k=1}^{N_i} C_{ik}} \tag{2-35}$$

NPD actúa de manera similar a PD, pero tiene en cuenta además para cada actividad el nivel de utilización del recurso procesador en el que reside en relación con el del flujo, según la siguiente ecuación:

NPD:
$$Sd_{ij} = D_i \frac{C_{ij} \cdot U_{ij}}{\sum_{k=1}^{N_i} C_{ik} \cdot U_{ik}} \tag{2-36}$$

donde U_{ij} representa la utilización del recurso procesador en el que reside la actividad τ_{ij}

Tanto PD como NPD tienen en común que reparten el plazo de principio a fin entre las actividades del flujo e2e de forma que se obtiene una asignación conforme a los plazos de principio a fin, propiedad que se establece con la ecuación (2-31).

2.4.3 *Equal Slack* (EQS) y *Equal Flexibility* (EQF)

En [KAO93][KAO97] se proponen otros dos algoritmos para repartir los plazos de principio a fin: EQS (*Equal Slack*), y EQF (*Equal Flexibility*). Estos algoritmos se propusieron para la asignación de plazos *on-line* a actividades en flujos e2e en sistemas de tiempo real no estricto, y el interés por ellos se ha venido centrando en entornos en los que se busca mejorar la calidad de servicio (*quality of service*) [YU05][SON01][WU13], pero no así garantizar el cumplimiento de plazos de tiempo estrictos. Aquí adaptamos su formulación para la asignación *off-line* de plazos de planificación, con el objetivo de estudiar nuevos métodos de asignación.

EQS asigna los plazos dividiendo equitativamente el *slack* entre las actividades del flujo e2e, entendiendo *slack* como la diferencia entre el plazo y el tiempo de respuesta de peor caso. Cabe notar que esta definición de *slack* es diferente a la utilizada en MAST. Como *a priori* no se conocen los tiempos de respuesta, utiliza los tiempos de ejecución de peor caso como una estimación a este tiempo de respuesta de peor caso. La ecuación de EQS en su formulación original se describe a continuación:

$$dl(T_i) = ar(T_i) + pex(T_i) + \frac{dl(T) - ar(T_i) - \sum_{j=i}^{m} pex(T_j)}{m - i + 1} \qquad (2\text{-}37)$$

donde los términos en los que se expresa tienen los siguientes significados de acuerdo a los conceptos definidos en el modelo MAST:

- $dl(T_i)$: plazo absoluto asignado a la actividad *i*-ésima del flujo e2e.
- $ar(T_i)$: instante de activación de la actividad *i*-ésima del flujo e2e.
- $pex(T_i)$: tiempo de ejecución de peor caso de la actividad *i*-ésima del flujo e2e.
- $dl(T)$: plazo absoluto de principio a fin del flujo e2e.
- m: número de actividades en el flujo e2e.

El algoritmo EQS original actúa de manera *on-line* asignando plazos absolutos en las actividades teniendo en cuenta los instantes de activación de éstas. Sin embargo, en MAST, al ser una herramienta de asignación *off-line*, los instante de activación de las actividades se desconocen. Para adaptar la ecuación (2-37) a la formulación de las técnicas de MAST, asumimos que los instantes de activación de las actividades ocurren en el instante 0, lo que significa que los plazos obtenidos toman como referencia el comienzo del flujo e2e. El algoritmo adaptado se muestra en la siguiente ecuación:

Adaptación de EQS:

$$SD_{ij} = C_{ij} + \frac{D_i - \sum_{k=j}^{N_i} C_{ik}}{N_i - j + 1} \qquad (2\text{-}38)$$

El algoritmo EQF original también se basa en el reparto del *slack* (definido de la misma manera que en EQS), pero lo realiza de manera proporcional a los tiempos de ejecución de cada actividad. Para ello define un nuevo concepto llamado "flexibilidad" (*flexibility*), que es la ratio entre el *slack* y el tiempo de respuesta (o de una estimación de éste). La ecuación de EQF en su formulación original se describe a continuación:

$$dl(T_i) = ar(T_i) + pex(T_i) + \left[dl(T) - ar(T_i) - \sum_{j=i}^{m} pex(T_j) \right] * \left[\frac{pex(T_i)}{\sum_{j=i}^{m} pex(T_j)} \right] \qquad (2\text{-}39)$$

donde los términos en los que se expresa son los utilizados en EQS.

Adaptamos la ecuación original de EQF para la asignación de plazos de planificación definidos según el modelo MAST. Para ello seguimos el mismo criterio seguido con EQS, asumiendo la activación de las actividades en el instante 0. La formulación de EQF adaptada a nuestro modelo se muestra en la siguiente ecuación:

Adaptación de EQF:

$$SD_{ij} = C_{ij} + \left[D_i - \sum_{k=j}^{N_i} C_{ik} \right] \left[\frac{C_{ij}}{\sum_{k=j}^{N_i} C_{ik}} \right] \quad (2\text{-}40)$$

En el transcurso de esta tesis, a estas adaptaciones de EQS y EQF que aquí presentamos las llamaremos simplemente EQS y EQF, aunque no sean exactamente los mismos algoritmos que los presentados en su trabajo original. Estos algoritmos adaptados no representan asignaciones locales conformes al plazo de principio a fin. Sin embargo, en el Capítulo 7 comprobaremos cómo estas adaptaciones son capaces de explotar el comportamiento anómalo que observamos en LC-EDF (desarrollado en el Capítulo 6). Adicionalmente, cabe notar que existen situaciones en las que estas adaptaciones de EQS y EQF no cumplen estrictamente con la ecuación (2-29) de plazo global ascendente. Aun así, las utilizaremos como asignaciones globales, con el objetivo de explorar las capacidades de planificación de nuevos métodos en sistemas GC-EDF.

2.4.4 Algoritmo HOSDA

Los algoritmos de asignación descritos anteriormente nos permiten resolver el problema de la asignación *off-line* de plazos de planificación de una manera relativamente sencilla y con un bajo coste computacional, ya que no requieren un uso iterativo de técnicas de cálculo de tiempos de respuesta. La contrapartida a esta sencillez es que, en el caso de no obtener una solución planificable, no poseen la capacidad de iterar sobre esta solución para intentar mejorarla y alcanzar la planificabilidad. En el campo de los sistemas de prioridades fijas, el algoritmo HOPA sí posee esta capacidad. Por ello, se definió el algoritmo HOSDA (*Heuristic Optimized Scheduling Deadline Assignment*) como una adaptación del algoritmo HOPA para sistemas EDF.

Se definen dos variantes del algoritmo HOSDA, el algoritmo HOSDA Local para LC-EDF [RIV10], y el HOSDA Global para GC-EDF [RIV09], ambos con un funcionamiento similar a HOPA.

Como se vio en la Sección 2.3, el algoritmo HOPA se basa en la obtención de una asignación de plazos virtuales en todas las actividades, y su posterior transformación en prioridades fijas utilizando el criterio *Deadline Monotonic*. El reparto inicial que se efectuaba era realmente una asignación local PD, y las posteriores iteraciones obtenían también plazos virtuales que eran asignaciones locales conformes. Por lo tanto, el algoritmo HOSDA Local se obtiene por aplicación del algoritmo HOPA en el que se elimina el paso 2 de traducción de plazos virtuales a prioridades fijas. Así pues, tanto los criterios de parada, como el cálculo de los nuevos plazo virtuales, que ahora son plazos locales de planificación, se realizan de la misma manera que la descrita para HOPA en la sección 2.3 y en [GUT95]. El esquema de funcionamiento de HOSDA Local se resume en el pseudocódigo mostrado en la Figura 2-2.

```
Algoritmo HOSDA_Local is
begin
   Asignación PD
   loop
      Cálculo de tiempos de respuesta de peor caso
      exit when se cumple algún requisito de parada
      Cálculo de los nuevos plazos locales
   end loop
end HOSDA_Local
```

Figura 2-2 Pseudocódigo del algoritmo HOSDA Local para LC-EDF

Con el algoritmo HOSDA Local se obtienen asignaciones locales conformes. El algoritmo HOSDA Global se aprovecha de esta propiedad, y utiliza la ecuación (2-32) para transformar los plazos de planificación locales en globales. Por lo tanto, los criterios de parada y método para calcular los nuevos plazos virtuales, que hay que transformar en globales, son los mismos que los planteados en HOSDA Local. El funcionamiento de HOSDA Global se resume en el pseudocódigo mostrado en la Figura 2-3.

```
Algoritmo HOSDA_Global is
begin
   Asignación PD
   Transformación local -> global
   loop
      Cálculo de tiempos de respuesta de peor caso
      exit when se cumple algún requisito de parada
      Cálculo de los nuevos plazos locales
      Transformación local -> global
   end loop
end HOSDA_Global
```

Figura 2-3 Pseudocódigo del algoritmo HOSDA Global para GC-EDF

2.5 Herramienta de análisis y optimización MAST

La herramienta de análisis y optimización de MAST es la herramienta, dentro del entorno MAST, encargada de llevar a cabo el análisis de planificabilidad, y la asignación de parámetros de planificación. Los sistemas sobre los que actúa se describen utilizando el modelo MAST, cuyos elementos más importante para el desarrollo de esta tesis se describieron en la Sección 1.4.

Los objetivos de esta tesis, planteados en la Sección 1.10, se desarrollan en torno a la herramienta de análisis y optimización de MAST. En primer lugar añadiremos el soporte para sistemas heterogéneos FP+EDF, modificando las técnicas implementadas dentro de la herramienta. En segundo lugar, se utilizará la herramienta de análisis y optimización de MAST dentro de un supercomputador para realizar un estudio comparativo masivo de las técnicas de sistemas distribuidos implementadas. Estos objetivos requieren modificar y extender la funcionalidad de dicha herramienta.

La herramienta de análisis y optimización de MAST se encuentra disponible en [MASTh], tanto en forma binaria como en código fuente. El código fuente se distribuye

con licencia GNU GPL, y está realizado en lenguaje Ada. El trabajo realizado en esta tesis parte de la versión 1.3.8.0 de dicha herramienta. En primer lugar, y como trabajo previo al desarrollo de los objetivos específicos de esta tesis, sobre esta versión se implementaron las siguientes técnicas de análisis de planificabilidad para sistemas distribuidos que ya estaban definidas, pero aún no habían sido incluidas en MAST:

- Análisis basado en *offsets* para GC-EDF [PAL03].
- Análisis basado en *offsets slanted* para FP [MAK08].
- Análisis basado en *offsets* de fuerza bruta para FP [TIN94b].

Tras la implementación de estas técnicas, a continuación se resume la funcionalidad incluida dentro de la herramienta de análisis y optimización de MAST, que además representa el punto de partida para la consecución de los objetivos planteados en esta tesis:

- Análisis de planificabilidad de sistemas monoprocesadores: análisis exacto de tiempos de respuesta de peor caso desarrollado originalmente por Harter [HAR87] y Joseph y Pandya [JOS86], extendido para soportar plazos arbitrarios y *jitter* [LEH90][TIN94d].
- Análisis de planificabilidad de sistemas distribuidos: implementa las técnicas descritas a en las Secciónes 2.1 y 2.2, resumidas en la Tabla 2-1.
- Asignación de prioridades fijas en sistemas monoprocesadores: para plazos menores o iguales a los periodos se implementa la asignación según el criterio *Deadline Monotonic* [LEU82]. Para plazos mayores a los periodos se implementa la técnica óptima desarrollada por Audsley [AUD91].
- Asignación de parámetros de planificación en sistemas distribuidos: implementa las técnicas resumidas en la Tabla 2-2.
- Cálculo de tiempos de bloqueo, soportando los protocolos IPCP [BAK91] y SRP [BAK91].
- Cálculo del *slack* de sistema, de recurso procesador, de flujo e2e, y de actividad. Estos conceptos están definidos en la Sección 1.4, y su cálculo se implementa con un algoritmo propio de MAST basado en búsqueda binaria.

	Holistic	**Offset Based**	**Offset Based Slanted**	**Offset Based w/pr**	**Offset Based Brute Force**
FP	✓ [TIN94]	✓ [TIN94b]	✓ [MAK08]	✓ [PAL99]	✓ [TIN94b]
GC-EDF	✓ [SPU96]	✓ [PAL03]			
LC-EDF	✓ [RIV10]				

Tabla 2-1 Disponibilidad inicial de técnicas de análisis de planificabilidad en sistemas distribuidos en MAST.

	Templado Simulado	**HOPA**	**HOSDA**	**PD**	**NPD**
FP	✓[GUT95b]	✓[GUT95]			
GC-EDF			✓[RIV09]	✓[LIJ00]	✓[LIJ00]
LC-EDF			✓[RIV10]	✓[LIJ00]	✓[LIJ00]

Tabla 2-2 Disponibilidad inicial de técnicas de asignación de parámetros de planificación en sistemas distribuidos en MAST.

La herramienta de análisis y optimización de MAST integra diferentes componentes que automatizan el uso de las técnicas citadas anteriormente. En la Figura 2-4 se muestra

un esquema general del funcionamiento de esta herramienta. El proceso de ejecución de las técnicas implementadas se resumen en los siguientes pasos:

1. El usuario especifica las técnicas a aplicar, y sobre qué sistema se aplicarán. La herramienta de análisis y optimización de MAST es un ejecutable de línea de comandos. Adicionalmente, MAST también incluye una herramienta gráfica que permite opcionalmente configurar visualmente la ejecución.
2. El *Parser* de MAST lee el fichero de entrada en el que se describe un sistema siguiendo el modelo MAST. Como resultado, el sistema queda almacenado en memoria, utilizando estructuras Ada, sobre las que se definen procedimientos de lectura y escritura.
3. Seguidamente se comprueba que el sistema cumpla con una serie restricciones y reglas de consistencia que aseguran su compatibilidad con las técnicas a aplicar. Por ejemplo, se verifica que ningún recurso procesador tenga una utilización por encima del 100%, o que las prioridades estén todas en el rango correcto impuesto por el planificador.
4. A continuación se aplican las técnicas requeridas por el usuario (análisis de planificabilidad y/o asignación de parámetros de planificación).
5. Una vez que las técnicas finalicen su ejecución, se crea un fichero de resultados en el que se almacenan los tiempos de respuesta calculados, junto con los *jitters* asociados, y opcionalmente, los valores del *slack* (de los flujos e2e, recursos procesadores, sistema, y/o actividad). Si se ha especificado realizar una asignación de parámetros de planificación, la herramienta de análisis y optimización de MAST también puede devolver un fichero descriptor de sistema similar al de entrada, pero con la nueva asignación.

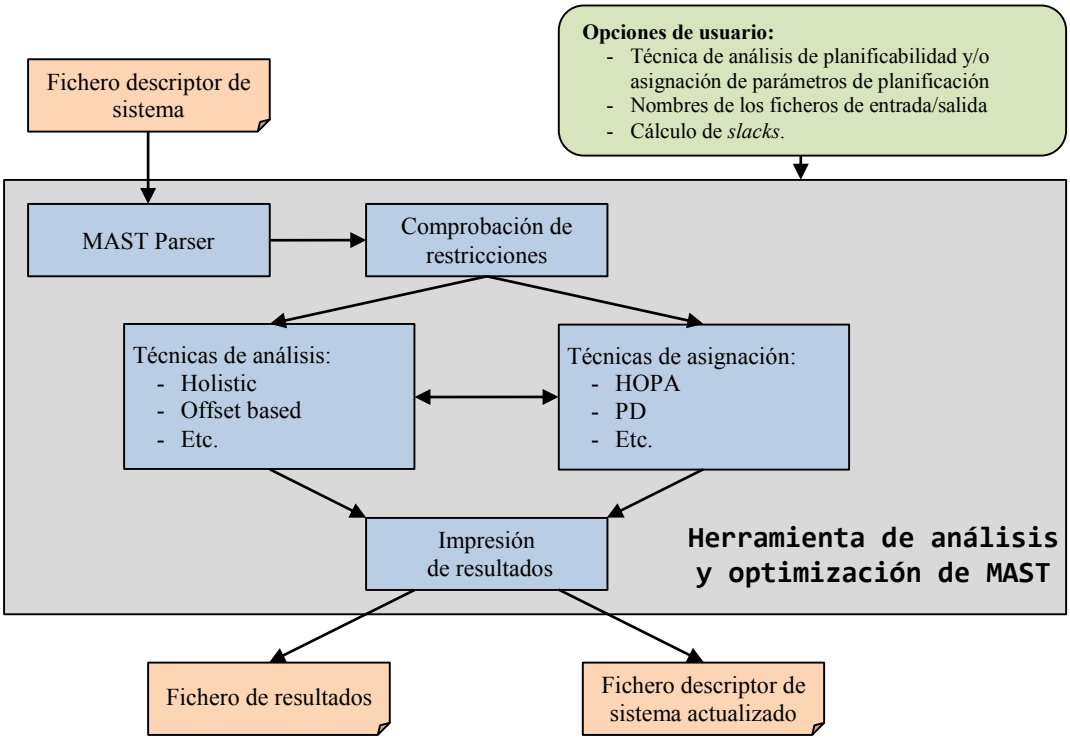


Figura 2-4 Esquema de funcionamiento de la herramienta de análisis y optimización de MAST.

3 Sistemas Distribuidos Heterogéneos FP+EDF

La elección de qué política de planificación utilizar depende de los requerimientos de la aplicación concreta. Existen situaciones en las que FP resulta la técnica apropiada, y en otras EDF es la que obtiene mejores resultados. Adicionalmente, otro caso que puede presentarse es el de un sistema en el que la solución óptima sea planificar parte de él con FP, y otra con EDF. La combinación de FP y EDF en un mismo sistema es un campo con especial interés para ser explorado, por las claras ventajas que conllevaría su soporte.

En los sistemas monoprocesadores, esta combinación puede lograrse utilizando planificación jerárquica, de manera que típicamente las actividades con mayor criticidad se planifican en un nivel de prioridad alto, y el resto de actividades podrían planificarse con EDF para un mayor aprovechamiento de los recursos, pero conceptualmente utilizando un nivel de prioridad fija bajo. De esta forma se consigue aislar las actividades críticas de la interferencia debida a las actividades EDF. La planificación jerárquica para sistemas monoprocesadores es un área ampliamente estudiada. Existen técnicas para analizar este tipo de esquemas [GON03], e implementaciones en lenguajes de programación como Ada [BUR09][TUC12].

En sistemas distribuidos es habitual encontrarse con sistemas en los que los diferentes recursos procesadores se dedican a tareas con tipos de carga o requerimientos muy diversos. En estas situaciones resultaría beneficioso utilizar en cada recurso procesador la política de planificación que mejor se adapte al trabajo que va a realizar. Pueden darse otras razones por las que se deban combinar diferentes políticas en un mismo sistema distribuidos. Por ejemplo, podemos tener algún módulo heredado que típicamente utiliza FP, o una red de comunicación FP como el bus CAN, que deban integrarse dentro de un sistema con procesadores en el que por motivos de rendimiento se ha elegido EDF como política de planificación. A los sistemas distribuidos en el que los distintos nodos poseen una política de planificación diferente los llamamos sistemas distribuidos heterogéneos,

en oposición a los sistemas homogéneos, en los que todos sus nodos se planifican con la misma política.

Las técnicas de análisis de planificabilidad para sistemas distribuidos operan sobre sistemas homogéneos en los que los recursos procesadores son todos FP o EDF, no existiendo ninguna técnica de análisis de planificabilidad que actúe sobre todo un sistema distribuido heterogéneo considerado de manera global. De la misma forma, las técnicas de asignación de parámetros de planificación en la actualidad sólo son aplicables sobre sistemas homogéneos.

En el presente capítulo propondremos una aproximación composicional que dará soporte para sistemas heterogéneos a dos niveles distintos:

- Análisis de planificabilidad: Se adaptarán las técnicas de análisis de sistemas homogéneos presentadas en el capítulo 2, para dar soporte a sistemas heterogéneos.

- Asignación de parámetros de planificación: Se integrarán las técnicas de asignación actuales para FP y EDF dentro de la nueva técnica HOSPA (*Heuristic Optimized Scheduling Parameters Assignment*) para sistemas heterogéneos.

Como podemos considerar que los sistemas homogéneos no son más que un caso particular de sistema heterogéneo en el que existe un único tipo de planificador, la metodología que aquí presentaremos también será aplicable a sistemas homogéneos, pero no aportará ninguna ventaja sobre las técnicas ya existentes para ellos.

El capítulo se dividirá de la siguiente forma. En primer lugar se abordará el problema del análisis de planificabilidad en sistemas distribuidos heterogéneos (Sección 3.1), para a continuación tratar la asignación de parámetros de planificación (Sección 3.2). Para finalizar se presentarán un caso de estudio en el que se pondrán a prueba las técnicas propuestas, y se mostrarán los beneficios obtenidos al combinar distintas políticas de planificación (Sección 3.3).

3.1 Análisis de planificabilidad para sistemas heterogéneos

Existen diferentes abstracciones con las que afrontar el análisis de planificabilidad de sistemas distribuidos. En [PER09] se destacan y comparan las cuatro más relevantes: visión holística (utilizada en MAST), los análisis composicionales de SymTA/S [HEN05][HAM04] y MPA-RTC [THI01], y el análisis basado en autómatas temporales.

Los análisis composicionales se basan en la combinación de modelos de análisis local sobre las actividades disparadas por eventos de entrada, y la propagación de eventos de salida. Esta visión es compatible con el análisis formal de rendimiento para sistemas distribuidos heterogéneos. SymTA/S [HAM06] es una herramienta comercial y de código cerrado que implementa un análisis composicional, con soporte para planificación FP expulsora y no expulsora, *Round Robin* o EDF, redes con acceso de tipo TDMA (*Time Division Multiple Access*), o bus CAN, y adaptación de modelo de eventos [RIC03]. Por otra parte, MPA-RTC no se apoya en la teoría de planificación clásica, sino que utiliza la técnica *Real-Time Calculus* [THI00], que extiende los conceptos de la técnica *Network*

Calculus [BOU01] para su aplicación en el análisis de flujos de eventos que recorren redes de computación y comunicación. Esta técnica es aplicada para soportar planificación FP, EDF y TDMA. La aproximación basada en autómatas temporales permite el soporte de sistemas distribuidos heterogéneos, pero sólo es aplicable en sistemas de tamaño reducido.

La visión holística que utilizan las técnicas de análisis de planificabilidad presentes en MAST no permiten la combinación de diferentes esquemas de planificación en un mismo sistema, salvo para la planificación jerárquica dentro de un mismo recurso procesador. En este capítulo modificaremos la abstracción holística para que pueda soportar el análisis de sistemas heterogéneos, y así alinearlo con las capacidades de las aproximaciones composicionales.

Las técnicas de análisis que siguen la visión holística sobre las que trabajaremos son las que forman parte de la herramienta MAST. Estas fueron introducidas en los Capítulos 1 y 2. A modo de recordatorio se enumeran a continuación:

- Análisis holístico (*holistic analysis*) para FP, GC-EDF y LC-EDF. No confundir con la visión holística del análisis que poseen todas las herramientas integradas en MAST.
- Análisis basado en *offsets* (*offset based analysis*) para FP y GC-EDF.
- Análisis basado en *offsets slanted* (*offset based slanted analysis*) para FP.
- Análisis basado en *offsets* con relaciones de precedencia (*offset based with precedende relationships analysis*) para FP.
- Análisis basado en *offsets* de fuerza bruta (*offset based brute force analysis*) para FP.

Todas estas técnicas integradas en MAST, y en general todas las que siguen la abstracción holística, tienen un esquema general de funcionamiento similar, que se puede simplificar en el siguiente pseudocódigo:

```
Algoritmo análisis
   inicialización tiempos de respuesta, jitters y offsets
   loop
      for each flujo e2e
         for each actividad en el flujo
            cálculo tiempo respuesta de actividad
            actualización de jitters y offsets
         end loop
      end loop
      exit when no hay cambios
   end loop
end análisis
```

Figura 3-1 Pseudocódigo del análisis de planificabilidad con visión holística del sistema.

Tal y como se observa en el pseudocódigo de la Figura 3-1, los análisis con visión holística comienzan asignando unos valores iniciales a los términos de tiempos de respuesta, *jitters* y *offsets* del sistema. A continuación se itera sobre cada actividad de cada flujo e2e del sistema, y se aplica la técnica de análisis designando a dicha actividad como la actividad bajo análisis. Una vez que se calcula el tiempo de respuesta de la actividad bajo análisis, se utilizan dichos resultados para actualiza los *jitters* y *offsets* de las actividades posteriores en el flujo e2e, y se procede a analizar la siguiente actividad en

el sistema. El ciclo del análisis finaliza cuando, tras recorrer todas las actividades del sistema, no hay variación en ningún término de tiempos de respuesta, *jitter* u *offset*.

Cada vez que se designa una actividad como la actividad bajo análisis, sólo se considera el recurso procesador en el que se sitúa, y los resultados se propagan al resto de actividades del flujo e2e, pudiendo éstos estar situados en otros recursos procesadores. Este comportamiento nos permite proponer un método composicional en el que extraemos del análisis tradicional la parte que se dedica al análisis de cada actividad, y lo aislamos en un módulo al que llamamos Análisis de Actividad. El aislamiento del Análisis de Actividad se completa estableciendo un procedimiento con el que cada uno de ellos comunica sus resultados a los demás, haciendo uso de términos a los que llamamos *jitters* y *offsets* heredados.

Aplicando nuestra metodología composicional, el análisis del sistema completo consiste en la ejecución para cada actividad de su correspondiente Análisis de Actividad, propagando los resultados mediante los *jitter* y *offsets* heredados hasta que converja a una solución estable. La independencia de los Análisis de Actividad permite que cada actividad pueda analizarse incluso con una técnica distinta, abriendo la puerta a la combinación de distintas políticas de planificación en los recursos procesadores de un mismo sistema, y por tanto también al uso de políticas de planificación diferentes en el mismo flujo e2e. A continuación formalizamos esta metodología composicional.

Conceptualmente, el Análisis de Actividad representa un módulo (Figura 3-2) que, para la actividad τ_{ij} perteneciente al flujo e2e Γ_i , calcula los tiempos de respuesta de peor y mejor caso (R_{ij}, Rb_{ij}) utilizando alguna de las técnicas disponibles (análisis holístico o basados en *offsets*). Para operar tan sólo necesita como entrada el parámetro de planificación de la actividad bajo análisis (SD_{ij}/Sd_{ij} para EDF, P_{ij} para FP), los *jitters* heredados (J'_{ij}), los *offsets* heredados (Φ'_{ij}), y la información del análisis actualizada del resto de actividades situadas en el mismo procesador.

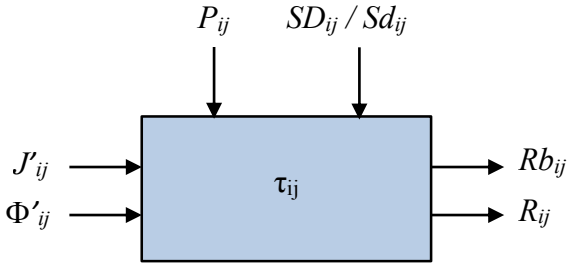


Figura 3-2 Parámetros de entrada y salida del análisis de actividad

La comunicación de los resultados entre los diferentes Análisis de Actividad mediante los *jitters* y *offsets* heredados se define siguiendo los principios establecidos según el modelo de eventos de MAST, en el que la actividad τ_{ij+1} se activa una vez que la actividad precedente τ_{ij} finaliza su ejecución. Además de por unos valores de *jitter* y *offset* iniciales caracterizados en el modelo de eventos del sistema, la activación de las actividades posteriores a la primera en el flujo también se pueden ver afectadas por la variabilidad de los tiempos de respuesta de las actividades precedentes. El instante de activación de la instancia *k*-ésima de la actividad τ_{ij} (j >1) depende de los tiempos de finalización de las actividades precedentes, y se puede acotar en el siguiente intervalo:

$$activacion \in \left[t_i^k + \max\left(Rb_{ij-1}, \phi_{ij}\right), t_i^k + J_{ij} + \max\left(R_{ij-1}, \phi_{ij}\right) \right] \quad (3\text{-}1)$$

donde t_i^k es el instante de la *k*-ésima activación de la primera actividad del flujo Γ_i.

El *jitter* heredado de la actividad τ_{ij}, J'_{ij} , se obtiene como la diferencia entre el tiempo de activación de mejor y de peor caso:

$$J'_{ij} = J_{ij} + \max\left(R_{ij-1}, \phi_{ij}\right) - \max\left(Rb_{ij-1}, \phi_{ij}\right) \quad (3\text{-}2)$$

El *offset* heredado de la actividad τ_{ij}, Φ'_{ij} , se define como el lapso de tiempo mínimo que debe transcurrir desde que se active el flujo e2e (t_{in}), hasta que la actividad τ_{ij} se active:

$$\Phi'_{ij} = \max\left(Rb_{ij-1}, \phi_{ij}\right) \quad (3\text{-}3)$$

El esquema general de ejecución del análisis para sistemas heterogéneos utilizando el método composicional propuesto (Figura 3-3) se asemeja al análisis tradicional para sistemas homogéneos. Los Análisis de Actividad se ejecutan iterativamente en todas las actividades del sistema, y el proceso finaliza cuando tras recorrer todo el sistema no se produce ningún cambio. La característica de monotonicidad del análisis se mantiene, por lo que la dependencia *jitter* y tiempo de respuesta de peor caso se resuelve de manera iterativa, al igual en los análisis para sistemas homogéneos.

El método composicional que proponemos permite que cada actividad pueda ser analizada con un Análisis de Actividad distinto, pudiendo así combinar distintas políticas de planificación o técnicas de análisis dentro de un mismo sistema. Una característica que deben tener los Análisis de Actividad es que sólo deben actuar sobre una única actividad, y relegar la propagación de los resultados al mecanismo definido con los *jitter* y *offsets* heredados. Esta restricción plantea limitaciones sobre el conjunto de técnicas de análisis que pueden descomponerse para ser utilizadas con este método composicional. Una de las técnicas que sufre estas limitaciones es el análisis basado en *offsets* con relaciones de precedencia.

Durante el proceso de ejecución del análisis basado en *offsets* con relaciones de precedencia, pueden ocurrir situaciones intermedias en las que los tiempos de respuesta de peor caso dentro de un flujo e2e no sean siempre ascendentes. Para paliar este problema, cuando se actualiza un tiempo de respuesta de peor caso, se realiza un barrido en el flujo e2e que comprueba y corrige estos problemas. Como consecuencia, al analizar una única actividad, se pueden modificar los tiempos de respuesta de peor caso de otras actividades dentro del mismo flujo e2e. Si se usara en un mismo sistema una técnica de análisis que no utiliza relaciones de precedencia junto con otra que sí las utiliza, se podrían producir casos en los que la convergencia nunca se alcanzaría. Por esta razón, cualquier implementación de nuestro método composicional debe restringir el uso de las técnicas que utilizan relaciones de precedencia con aquellas que no las utilizan.

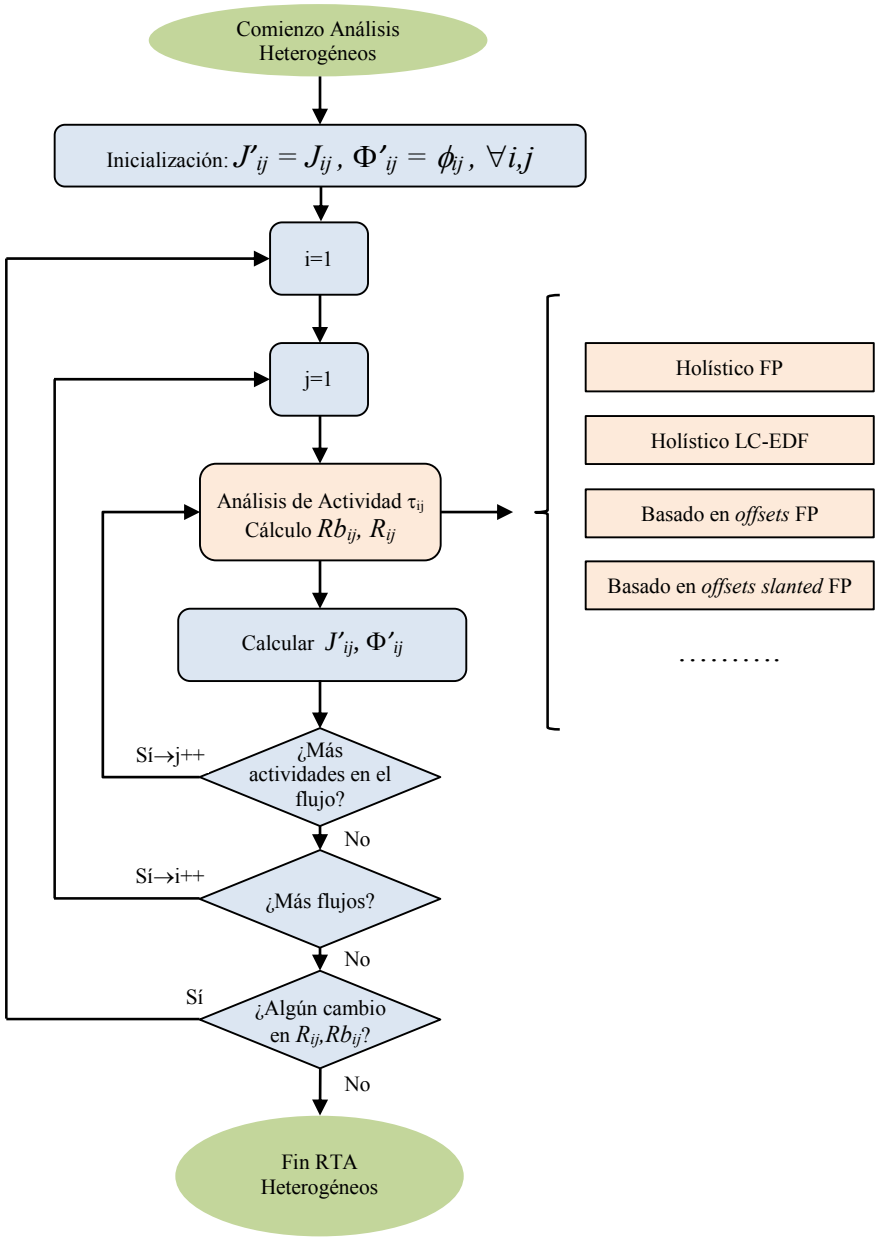


Figura 3-3 Algoritmo de análisis para sistemas heterogéneos.

3.1.1 Integración en MAST

Para poder explotar las capacidades de la metodología composicional propuesta, procedemos a implementarla dentro de la herramienta de análisis y optimización de MAST. Hay que tener presente que esta implementación se lleva a cabo para añadir a MAST el soporte para el análisis de sistemas heterogéneos, en los que cada recurso procesador puede ser analizado con una técnica distinta, acorde a la política de planificación que utilice.

Como primer paso en la integración en MAST se han implementado los diferentes procedimientos de Análisis de Actividad para cada una de las técnicas de análisis. Para ello definimos un prototipo de procedimiento en lenguaje Ada, *Task_Analysis_Tool* (ver

Figura 3-4) al que se deben adaptar estos Análisis de Actividad, y al que se le pasan los siguientes parámetros:

- The_System: Parámetro de entrada y salida en el que se almacena la descripción del sistema.
- I, J: Índices de la actividad τ_{ij} sobre la que actuará el Análisis de Actividad.
- Changes_Made: Parámetro booleano de salida. Se establece con valor True si el Análisis de Actividad ha actualizado el tiempo de respuesta de la actividad.
- Stop_Factor_When_Not_Schedulable: Cuando algún tiempo de respuesta es superior a Stop_Factor_When_Not_Schedulable*D_i, el análisis finaliza su ejecución. Este es un valor que puede establecer el usuario (por defecto es infinito), y se utiliza como método de interrumpir el análisis de sistemas con convergencias excesivamente lentas, o cuando sólo nos interesa saber si el sistema es planificable o no, en cuyo caso se usará el valor 1.
- Over_Analysis_Bound: Parámetro de salida que se establece a True cuando el análisis finaliza debido al parámetro Stop_Factor_When_Not_Schedulable.

```
type Task_Analysis_Tool is access procedure
     (The_System                       : in out Mast.Systems.System;
      I                                : Transaction_ID_Type;
      J                                : Task_ID_Type;
      Changes_Made                     : out Boolean;
      Stop_Factor_When_Not_Schedulable : in Positive := Positive'Last;
      Over_Analysis_Bound              : out Boolean);
```

Figura 3-4 Tipo de procedimiento Ada para el Análisis de Actividad

Los diferentes procedimientos de Análisis de Actividad se han implementado de acuerdo con el tipo definido *Task_Analysis_Tool*, y tomando como base las implementaciones existentes en MAST de las técnicas de análisis correspondientes para sistemas homogéneos.

El soporte para el análisis de sistemas heterogéneos se basa en la creación de un *array* con longitud igual el número de recursos procesadores, en el que cada elemento es un puntero a una función de Análisis de Actividad, de forma que cada recurso procesador tiene asignada una técnica de análisis distinta. La definición del tipo de este *array* se muestra en la Figura 3-5.

```
type Processor_Analysis_Accesses is array (1 .. Max_Processors) of Task_Analysis_Tool;
```

Figura 3-5 *Array* de procedimientos de Análisis de Actividad

Antes de realizar el análisis de un sistema es necesario rellenar el *array* de procedimientos de Análisis de Actividad de acuerdo con las políticas de planificación definidas para cada recurso procesador. Para ello se itera sobre cada recurso procesador del sistema, y dependiendo de qué política de planificación utilice, y el tipo de técnica de análisis que haya requerido el usuario, se completa el *array* con los punteros a los procedimientos adecuados. El usuario puede seleccionar un único tipo de técnica de análisis para todo el sistema. Los Análisis de Actividad implementados se detallan a continuación:

- *holistic_task_analysis*: Especifica un análisis holístico, disponible para recursos procesadores FP, GC-EDF y LC-EDF.
- *offset_based_task_analysis*: Especifica un análisis basado en *offsets*, disponible para recursos procesadores FP y GC-EDF.
- *offset_based_slanted_task_analysis*: Especifica un análisis basado en *offsets slanted*. Disponible para recursos procesadores FP.
- *offset_based_w_pr_task_analysis:* Especifica un análisis basado en *offsets* con relaciones de precedencia. Disponible para recursos procesadores FP.
- *offset_based_brute_force_task_analysis*: Especifica un análisis basado en *offsets* de fuerza bruta. Disponible para recursos procesadores FP.

En el caso de que el recurso procesador utilice una política de planificación no soportada por la técnica especificada por el usuario, se utilizará la técnica *holistic*.

Para analizar la totalidad del sistema, se define el procedimiento *Heterogenous_RTA* que implementa los pasos descritos en la Figura 3-3, haciendo uso del *array* de punteros a funciones de Análisis de Actividad. En la Figura 3-6 se muestra el pseudocódigo simplificado correspondiente a la implementación del procedimiento de análisis de sistemas heterogéneos. Los puntos destacados del diagrama de la Figura 3-3, que se indican en rojo en el pseudocódigo, son los siguientes: la inicialización de los *jitters* y *offsets*, la iteración en cada actividad del sistema, el análisis de la actividad, y la propagación de los *jitters* y *offsets* heredados.

```
procedure Heterogeneous_RTA
   (The_System                          : in out Mast.Systems.System;
    Stop_Factor_When_Not_Schedulable    : in Positive := Positive'Last)
is
   Access_to_Task_Analysis              : Processor_Analysis_Accesses;
   Changes_Made, Over_Analysis_Bound, Done       : Boolean := False
begin
   loop
      Done := True

      # se inicializa el array de punteros a funciones de Análisis de Actividad
      Initialize_Task_Analysis_Array(Access_to_Task_Analysis, The_System);
      # Se inicializan los jitters y offsets del sistema
      Initialize_Jitters_and_Offsets(The_System);

      # Se itera cada actividad τij del sistema
      for I in N_Flujos(The_System) loop
         for J in N_Actividades_en_flujo(I, The_System) loop

            # Se analiza la actividad τij con el tipo de analisis asignado a su procesador
            Access_to_Task_Analysis(Proc_Actividad(I,J))
                                    (The_System, I, J, Changes_Made,
                                     Stop_Factor_When_Not_Schedulable,
                                     Over_Analysis_Bound);

            # Se Calculan y propagan los jitters y offsets heredados
            Propagate_Jitters_and_Offsets(The_System, I, J);

            if Changes_Made then Done:= False; end if;

            exit when Over_Analysis_Bound;
         end loop;
         exit when Over_Analysis_Bound;
      end loop;
      exit when Done or Over_Analysis_Bound;
   end loop;
end Heterogeneous_RTA;
```

Figura 3-6 Pseudocódugo del procedimiento de análisis del sistema heterogéneo.

Desde la versión 1.4 de la herramienta de análisis y optimización de MAST, el análisis para sistemas distribuidos, sean heterogéneos o no, se realiza mediante el método composicional propuesto en esta sección, sustituyendo a las técnicas precedentes.

3.2 Asignación de Parámetros de Planificación en Sistemas Heterogéneos

Tal y como ya se estableció en los capítulos introductorios, la asignación de parámetros de planificación en un sistema distribuido de forma que éste cumpla con todos sus plazos es un problema NP-difícil, salvo para casos muy simples con poca utilidad práctica. Existen diversos métodos que se han propuesto para resolver este problema de una manera computacionalmente tratable, pero todos ellos fueron diseñados para actuar sobre sistemas homogéneos en los que sólo se utiliza una única política de planificación. En esta sección se abordará el problema de la asignación de parámetros de planificación en sistemas heterogéneos FP+EDF.

En un sistema heterogéneo, y en el contexto de esta tesis, nos encontraremos con diversos tipos de parámetros de planificación (prioridades fijas y plazos de planificación locales o globales) en un mismo sistema. La labor del algoritmo de asignación será la de asignar el parámetro de planificación adecuado en cada actividad, teniendo en cuenta el tipo de planificador que utilice, intentando aumentar la probabilidad de conseguir la planificabilidad del sistema.

Como hemos visto en el Capítulo 2, una forma habitual de resolver el problema de la asignación de parámetros de planificación en los sistemas distribuidos es mediante el reparto del plazo de principio a fin de los flujos e2e entre todas las actividades que lo componen, para a continuación utilizar este valor para el cálculo de la prioridad fija o del plazo de planificación. En primer lugar vamos a proponer cómo adaptar los algoritmos de asignación tradicionales (UD, ED, PD, NPD, EQS y EQF) [LIJ00][KAO93], para hacerlos compatibles con sistemas heterogéneos. Utilizando los fundamentos establecidos en esta adaptación, a continuación combinaremos los algoritmos heurísticos de asignación para sistemas homogéneos HOPA [GUT95] y HOSDA [RIV09][RIV10], creando el algoritmo heurístico HOSPA de asignación de parámetros de planificación de sistemas heterogéneos FP+EDF.

3.2.1 Adaptación de los algoritmos tradicionales de reparto del plazo de principio a fin

Los algoritmos de reparto del plazo de principio a fin toman el plazo de principio a fin del flujo e2e y, siguiendo algún tipo de criterio, lo dividen entre las actividades que lo componen, de forma que a cada actividad del sistema se le asigna un valor de plazo repartido. En el capítulo 2 se introdujeron los siguientes algoritmos de reparto: UD, ED, PD, NPD, EQS y EQF.

Las características de los plazos que obtienen estas técnicas limitan el tipo de planificación EDF en el que pueden ser utilizados. Los repartos PD y NPD, al ser conformes con el plazo de principio a fin, producen asignaciones que entendemos como

de plazos locales para LC-EDF. Adicionalmente, pueden transformarse sumando los plazos asignados en las actividades precedentes en el flujo e2e para obtener asignaciones de plazos globales de planificación compatibles con GC-EDF. Por otro lado, los repartos UD, ED, EQS y EQF obtienen asignaciones que consideramos como plazos de planificación globales para GC-EDF.

La asignación de parámetros de planificación para sistemas heterogéneos que proponemos se basa en la transformación de los plazos obtenidos por estos repartos, en parámetros de planificación apropiados para cada actividad dependiendo del planificador que utilice: prioridad fija para FP, plazo local de planificación para LC-EDF, y plazo global de planificación para GC-EDF.

Para formalizar el proceso de transformación reutilizamos el concepto de plazo virtual que se usaba en los algoritmos HOPA y HOSDA. Los plazos virtuales son los plazos asignados a todas las actividades de un sistema dado tras haber ejecutado alguno de los algoritmos de reparto de plazos de principio a fin. Al plazo virtual de la actividad τ_{ij} lo denotamos como VD_{ij} (*Virtual Deadline*). En el contexto de los sistemas heterogéneos, estos plazos virtuales no tienen ninguna validez como parámetro con el que planificar, deben ser transformados a un parámetro de planificación acorde con la política utilizada por la actividad.

Una vez que se ha llevado a cabo una asignación de plazos virtuales utilizando algún algoritmo de reparto, el siguiente paso consiste en transformar estos plazos virtuales en plazos de planificación, que podrán ser prioridades fijas, plazos globales de planificación, o plazos locales de planificación.

El proceso de transformación es la formalización en un mismo algoritmo de los criterios de interpretación de los plazos de planificación como local o global descritos en la Sección 2.4, y de las transformaciones llevadas a cabo en HOPA y HOSDA para convertir un reparto inicial en una parámetro de planificación. En la Tabla 3-1 se reúnen las transformaciones que adaptan las asignaciones de plazos virtuales llevadas a cabo por los algoritmos de reparto descritos, en los diferentes parámetros de planificación utilizados en FP, GC-EDF y LC-EDF. Mediante estas transformaciones, los algoritmos de reparto se hacen compatibles con otras políticas de planificación diferentes a aquellas para las que fueron diseñados originalmente, soportando ahora sistemas heterogéneos.

Para sistemas FP, la transformación consiste en aplicar el criterio *Deadline Monotonic* sobre la asignación de plazos virtuales, de igual manera que en el algoritmo HOPA. Para las asignaciones locales conformes al plazo de principio a fin (PD y NPD), la transformación a plazos locales de planificación para LC-EDF es directa; sin embargo, la transformación en plazos globales de planificación para GC-EDF se realiza aplicando la ecuación (2-32). Estos criterios para LC-EDF y GC-EDF son los utilizados en el algoritmo HOSDA (utiliza una asignación inicial PD por defecto).

Los algoritmos UD y ED obtienen asignaciones de plazos virtuales que cumplen con las ecuaciones (2-29) y (2-30), por lo que se consideran asignaciones de plazos globales de planificación, y no requieren transformación para su uso en GC-EDF. Los algoritmos EQF y EQS cumplen con la ecuación (2-30), pero no siempre con la ecuación (2-29) de plazos globales ascendentes. Utilizamos las asignaciones de EQS y EQF en GC-EDF de forma directa, con el objetivo de probar nuevos métodos de asignación en esta política de planificación.

Los valores obtenidos por UD, ED, EQS y EQF también pueden ser usados en sistemas LC-EDF, aunque como ya habíamos anticipado no son asignaciones lógicas al

no ser conformes con los plazos de principio a fin. Su uso en LC-EDF en esta tesis se incluye para estudiar nuevos métodos de asignación que intenten explotar los resultados anómalos que observamos en LC-EDF, y que se detallan en los Capítulos 6 y 7.

<table>
<tr><th></th><th>LC-EDF</th><th>GC-EDF</th><th>FP</th></tr>
<tr><td>PD</td><td rowspan="2">$Sd_{ij} = VD_{ij}$</td><td rowspan="2">$SD_{ij} = \sum_{k=1}^{j} VD_{ik}$</td><td rowspan="4">Transformación
Deadline Monotonic:
$VD_{ij} \Rightarrow P_{ij}$</td></tr>
<tr><td>NPD</td></tr>
<tr><td>UD</td><td rowspan="3">$Sd_{ij} = VD_{ij}$ (*)</td><td>$SD_{ij} = VD_{ij}$</td></tr>
<tr><td>ED</td><td rowspan="2">$SD_{ij} = VD_{ij}$ (**)</td></tr>
<tr><td>EQS</td><td></td></tr>
<tr><td>EQF</td><td colspan="3"></td></tr>
</table>

(*) No es una asignación local conforme al plazo de principio a fin.
(**) No cumplen siempre con la ecuación (2-29) de asignación global ascendente

Tabla 3-1 Transformaciones de plazos virtuales a parámetros de planificación.

3.2.2 Algoritmo de asignación HOSPA

Los algoritmos de reparto de plazo adaptados a sistemas heterogéneos descritos en la sección anterior nos proveen de una solución sencilla al problema de la asignación de parámetros de planificación en este tipo de sistemas. Sin embargo, no son algoritmos iterativos y por lo tanto carecen de la capacidad de mejorar una asignación inicial. Para sistemas homogéneos hemos presentado algoritmos heurísticos que sí poseen esta capacidad: HOPA [GUT95] para FP, y HOSDA para GC-EDF [RIV09] y LC-EDF [RIV10]. En esta sección integraremos estos algoritmos heurísticos dentro de una única técnica, HOSPA (*Heuristic Optimized Scheduling Parameter Assignment*), para la asignación y optimización de parámetros de planificación en sistemas heterogéneos.

HOSPA se basa en los mismos fundamentos que HOPA y HOSDA, haciendo uso de los mismos factores que influyen en la respuesta temporal del sistema para buscar y optimizar de forma iterativa una solución de parámetros de planificación. Sobre esta base introducida por HOPA y HOSDA, se añade la formalización de las transformaciones de plazos virtuales a parámetros de planificación descritas en la Sección 3.2.1.

El esquema general del funcionamiento de HOSPA se resume en la Figura 3-7. El algoritmo está compuesto por los siguientes pasos, heredados de los algoritmos HOPA y HOSDA:

1. Comienza llevando a cabo un reparto de plazos virtuales entre todas las actividades del sistema. Al igual que HOPA y HOSDA, lleva a cabo una asignación inicial conforme a los plazos de principio a fin, por defecto PD.
2. Se itera cada actividad, y se transforma su plazo virtual en el parámetro de planificación acorde a la política de planificación usada en su recurso procesador. Para ello se utiliza el criterio de transformación descrito en la Tabla 3-1 para asignaciones de plazos virtuales conformes al plazo de principio a fin (como por ejemplo PD).
3. Una vez que todas las actividades posean un parámetro de planificación, se calculan los tiempos de respuesta de peor caso de todas las actividades, haciendo uso de un análisis para sistemas heterogéneos.

4. Con los tiempos de respuesta de peor y mejor caso calculados, el algoritmo HOSPA comprueba si debe finalizar utilizando los mismos criterios de parada que HOPA y HOSDA.
5. Si no se cumple ningún criterio de parada, se calculan nuevos plazos virtuales. Para ello se hace uso de la misma formulación planteada por HOPA y utilizada posteriormente por HOSDA, usando los tiempos de respuesta de peor caso calculados, y los mismos parámetros k_a *y* k_r para controlar el proceso. La nueva asignación de plazos virtuales es también conforme a los plazos de principio a fin, por lo que el algoritmo continúa por el paso 2, repitiendo el proceso hasta que algún criterio de parada se cumpla.

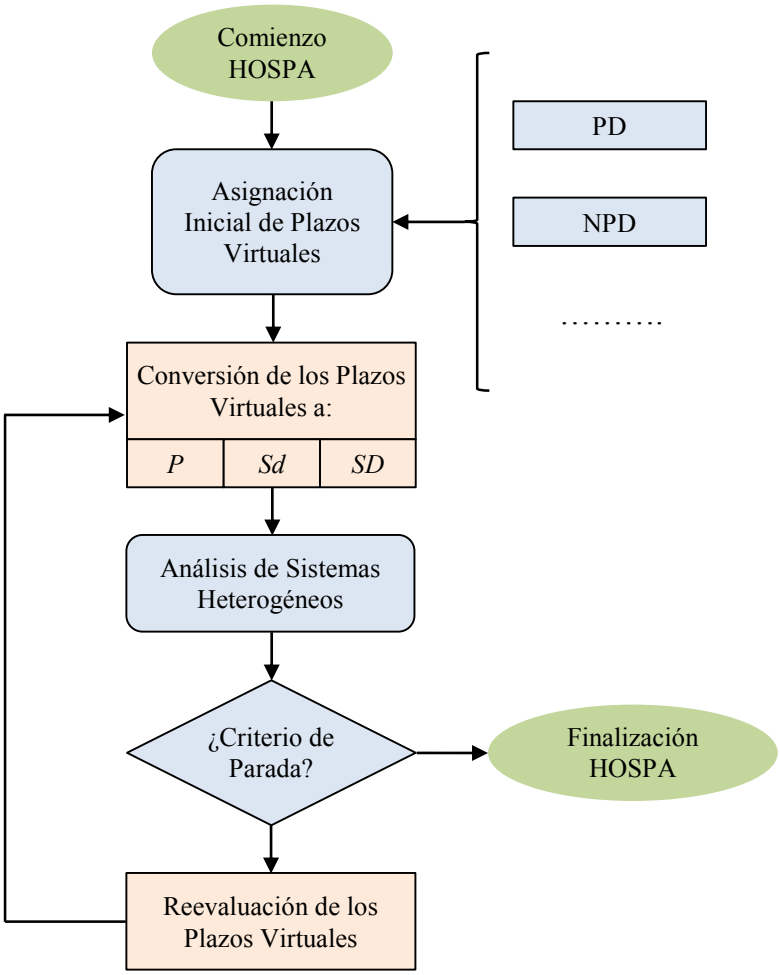


Figura 3-7 Esquema de funcionamiento del algoritmo de asignación HOSPA

3.2.3 Integración en MAST

Para completar el soporte de sistemas heterogéneos de MAST, implementamos el algoritmo HOSPA y el resto de algoritmos de reparto del plazo de principio a fin.

El usuario puede especificar como parámetro de ejecución de MAST la técnica de asignación de parámetros de planificación a aplicar en todo el sistema, siendo los valores admitidos los siguientes: UD, ED, PD, NPD, EQS, EQF, HOSPA

Si el usuario elige HOSPA, tiene la opción de modificar su funcionamiento mediante el mismo fichero de configuración que se utilizaba en HOPA y HOSDA, en el que puede establecer los siguientes valores:

- Tipo de asignación inicial. En HOSPA sólo se soportan asignaciones locales conformes. De momento implementa PD y NPD, siendo PD la utilizada por defecto. Adicionalmente se puede utilizar como asignación inicial la asignación que ya pueda venir establecida en el fichero descriptor del sistema.
- Pareja de valores [*ka, kr*]. Por defecto se utilizan 3 parejas de valores para estos parámetros: [1.5, 1.5], [2.0, 2.0], [3.0, 3.0].
- Número de iteraciones a llevar a cabo con cada pareja de valores *ka, kr*. Por defecto es 60.
- Iteraciones a realizar sobre una solución ya planificable. Por defecto el valor es 0.

La implementación en MAST de la asignación de parámetros de planificación en sistemas heterogéneos se integra en un único procedimiento llamado *HOSPA*. Este procedimiento está compuesto de dos partes:

1. Asignación inicial: En este primer paso se lleva a cabo una asignación inicial de parámetros de planificación en función de la técnica especificada por el usuario.
    a. Si el usuario ha elegido aplicar UD, ED, PD, NPD, EQS o EQF, se lleva a cabo dicha asignación, haciendo uso de las transformaciones descritas en la Tabla 3-1. A continuación el procedimiento *HOSPA* finaliza su ejecución.
    b. Si el usuario ha elegido aplicar HOSPA, esta primera asignación es PD o NPD (según venga especificado en el fichero de configuración de HOSPA), o la ya existente en el modelo del sistema.
2. Parte iterativa: En esta parte se implementa la funcionalidad iterativa del algoritmo HOSPA, con el que es capaz de calcular una asignación de parámetros de planificación nueva a partir de una asignación previa y los tiempos de respuesta de peor caso. La ejecución de esta parte es opcional, y sólo ocurre si el usuario ha seleccionado HOSPA como técnica de asignación de parámetros de planificación.

El funcionamiento del procedimiento *HOSPA* se describe de manera simplificada en el pseudocódigo de la Figura 3-8. Sus parámetros de entrada y salida son los siguientes:

- The_System: Parámetro de entrada y salida en el que se encuentra la descripción del sistema.
- Analysis_Tool: Puntero al procedimiento de análisis. Por requerimiento de las definiciones de los procedimientos de asignación en MAST, se debe aportar el puntero a la técnica de análisis. En el caso del procedimiento *HOSPA*, ese puntero apunta siempre a la técnica de análisis de sistemas heterogéneos implementada en MAST y descrita en la Sección 3.1.1.

```
procedure HOSPA (The_System      : in out Mast.Systems.System;
                 Analysis_Tool   : in Mast.Tools.Worst_Case_Analysis_Tool)
is
   Initial_Assignment      : String;
   HOSPA_will_iterate      : Boolean;
begin

   # 1.- Se obtiene la técnica de asignación requerida por el usuario (UD, ED, PD, HOSPA, etc.)
   Initial_Assignment := Get_User_Analysis(The_System)

   # 2.- Si se desea aplicar HOSPA, se obtiene su asignación inicial
   if Initial_Assignment = "HOSPA" then
      Initial_Assignment := Get_HOSPA_Initial_Assignment() #Puede ser PD, NPD o USER.
      HOSPA_will_iterate := True
   else
      HOSPA_will_iterate := False
   end if;

   # 3.- Se calculan los plazos virtuales iniciales, y se transforman en parámetros de
planificación
   Perform_Initial_Virtual_Assignment(Initial_Assignment, The_System);
   Virtual_Deadlines_to_Scheduling_Parameters(The_system)

   # 4.- Si se aplica algoritmo HOSPA, se itera sobre esta asignación inicial, según diagrama
Figura 3-7
   if HOSPA_will_iterate then
      loop
         Analysis_Tool(The_system,…)
         exit when Stopping_Criterion()
         New_Virtual_Deadlines(The_system)
         Virtual_Deadlines_to_Scheduling_Parameters(The_system)
      end loop;
   end if;

end HOSPA;
```

Figura 3-8 Pseudocódigo del procedimiento de asignación de parámetros de planificación en sistemas heterogéneo.

Desde la versión 1.4 de la herramienta MAST, la labor de asignación de parámetros de planificación en sistemas distribuidos (heterogéneos o no), la realiza por defecto el procedimiento *HOSPA*, sustituyendo a las técnicas precedentes (a excepción del templado simulado, que se mantiene como técnica independiente).

3.3 Caso de estudio: sistema de control de vuelo

En esta sección se mostrará a modo de ejemplo una situación en la que, desde el punto de vista del diseñador del sistema, puede resultar interesante tener la capacidad de analizar y asignar parámetros de planificación en un sistema heterogéneo FP+EDF.

Este ejemplo consiste en un sistema de control de vuelo simplificado descrito en [JAY08]. El proceso de diseño de este tipo de sistemas suele regirse por unos requisitos muy estrictos para su certificación, por lo que es habitual que utilicen esquemas de planificación extensamente probados y verificados en la industria, como el ejecutivo cíclico o algún otro equivalente, y no así planificadores basados en prioridades como EDF o FP. En el presente caso de estudio mostraremos a modo teórico la flexibilidad que aportan los sistemas distribuidos heterogéneos EDF+FP.

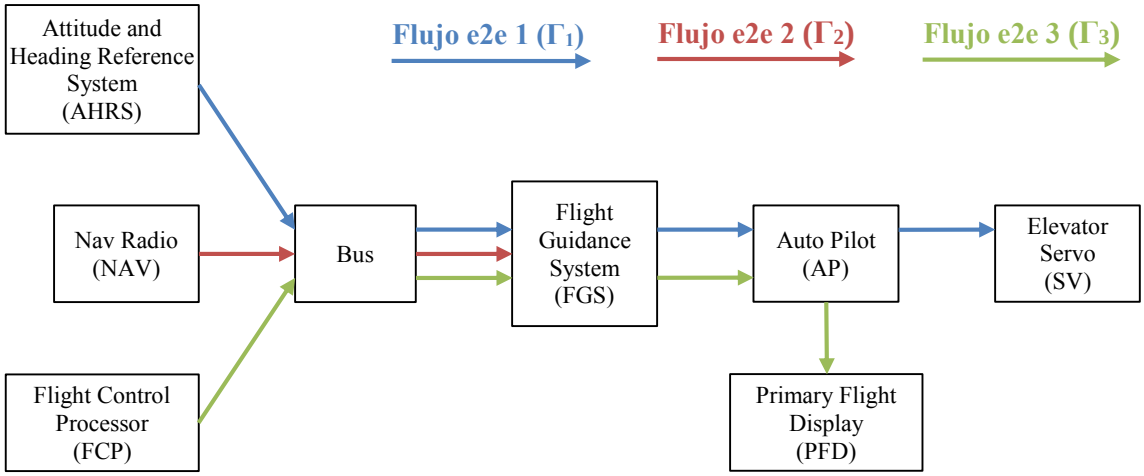


Figura 3-9 Recursos procesadores y flujos e2e del sistema de control de vuelo simplificado

En la Figura 3-9 se muestra el esquema del sistema de control de vuelo simplificado, presentado en [JAY08], del que mantenemos la nomenclatura de los diferentes recursos procesadores, y los valores temporales del sistema (tiempos de ejecución de peor caso, periodos y plazos de principio a fin). El sistema está formado por tres flujos e2e periódicos ejecutados en ocho recursos procesadores. El flujo e2e 1 (Γ_1) recibe lecturas periódicas de los sensores en el *Attitude and Heading Reference System* (AHRS), procesa la información en el *Flight Guidance System* (FGS) y el *Auto Pilot* (AP) de forma secuencial, y envía las señales de control al *Elevator Servo* (SV). El flujo e2e 2 (Γ_2) recibe periódicamente información relativa a la navegación en el *Navigation Radio* (NAV) y procesa la información en FGS. El flujo e2e 3 (Γ_3) lee entradas del *Flight Control Processor* (FCP), las procesa en el FGS y el AP, y muestra los resultados en el *Primary Flight Display* (PFD). Todas las lecturas e información recibida en el FGS se reciben a través de un bus. Por simplificación, tratamos a este bus como un procesador más en el sistema, y a los mensajes transmitidos en él como actividades ejecutando en un procesador.

En la Tabla 3-2 se muestran los tiempos de ejecución de peor caso de cada actividad junto con el recurso procesador en el que se ejecutan, y los periodos y plazos de principio a fin de cada flujo e2e.

	AHRS (C_{ij})	**NAV (C_{ij})**	**FCP (C_{ij})**	**BUS (C_{ij})**	**FGS (C_{ij})**	**AP (C_{ij})**	**SV (C_{ij})**	**PFD (C_{ij})**	**Periodo (T_i)**	**Plazo e2e (D_i)**
Γ_1	10			4	15	20	10		100	100
Γ_2		10		6	20				250	200
Γ_3			15	15	10	15		10	500	450

Tabla 3-2 Tiempos de ejecución de peor caso de las actividades, y periodos y plazos de principio a fin de los flujos e2e, del sistema de control de vuelo simplificado (en ms.).

Sobre este ejemplo sencillo, en un primer lugar aplicaremos una política FP en todos los recursos procesadores. Posteriormente modificaremos la especificación del sistema, y mostraremos cómo la combinación de diferentes políticas de planificación, en combinación con las técnicas para sistemas heterogéneos presentadas en este capítulo, permiten afrontar este cambio en el sistema.

Como primer paso, modelamos el sistema con MAST, utilizando en la primera aproximación una política FP en todos los recursos procesadores. A continuación utilizamos la herramienta de análisis y optimización de MAST para calcular una asignación de prioridades fijas con el algoritmo HOSPA, y calculamos los tiempos de respuesta de peor caso con el análisis de sistemas heterogéneos. En todos los casos utilizamos el análisis basado en *offsets* (*offset based analysis*).

Las prioridades fijas calculadas con HOSPA, y los correspondientes tiempos de respuesta de peor caso se muestran en la Tabla 3-3. Las prioridades fijas se seleccionan en el rango entre 1 y 256, siendo 256 la prioridad máxima. En la tabla se puede comprobar que el sistema es ampliamente planificable.

	AHRS (P_{ij})	**NAV (P_{ij})**	**FCP (P_{ij})**	**BUS (P_{ij})**	**FGS (P_{ij})**	**AP (P_{ij})**	**SV (P_{ij})**	**PFD (P_{ij})**	***D_i***	***R_i***
Γ_1	1			171	171	129	1		100	**59**
Γ_2		1		86	1				200	**65**
Γ_3			1	1	86	1		1	450	**110**

Tabla 3-3 Asignación de prioridades fijas y tiempos de respuesta de peor caso para el sistema de control de vuelo simplificado, planificado con FP (tiempos en ms.).

Posteriormente, debido a un cambio en la especificación del sistema, las actividades del recurso procesador FGS ahora requieren realizar cálculos computacionalmente más intensivos, resultando en un aumento de sus tiempos de ejecución de peor caso por un factor de 3,5. Sin embargo, los periodos y los plazos de principio a fin no se modifican. Las características temporales del nuevo sistema se muestran en la Tabla 3-4. El aumento en los tiempos de ejecución de peor caso provoca que la utilización en el recurso procesador FGS aumente del 25% al 87,5%.

	AHRS (C_{ij})	**NAV (C_{ij})**	**FCP (C_{ij})**	**BUS (C_{ij})**	**FGS (C_{ij})**	**AP (C_{ij})**	**SV (C_{ij})**	**PFD (C_{ij})**	**Periodo (T_i)**	**Plazo e2e (D_i)**
Γ_1	10			4	**52,5**	20	10		100	100
Γ_2		10		6	**70**				250	200
Γ_3			15	15	**35**	15		10	500	450

Tabla 3-4 Tiempos de ejecución de peor caso de las actividades, y periodos y plazos de principio a fin de los flujos e2e, del sistema de control de vuelo simplificado, con mayor carga en el recurso procesador FGS (en ms.).

Modelamos la nueva situación del sistema con MAST, y aplicamos HOSPA y el análisis basado en *offsets* (*offset based analysis*). En la Tabla 3-5 se muestra la asignación de prioridades fijas (que no varía con respecto al caso anterior), y los tiempos de respuesta de peor caso asociados. En la tabla se comprueba que en este caso el sistema no es planificable, ya que el flujo e2e 2 no cumple su plazo.

	AHRS (P_{ij})	**NAV (P_{ij})**	**FCP (P_{ij})**	**BUS (P_{ij})**	**FGS (P_{ij})**	**AP (P_{ij})**	**SV (P_{ij})**	**PFD (P_{ij})**	***Di***	***R_i***
Γ_1	1			171	171	129	1		100	**96,5**
Γ_2		1		86	1				200	**282,5**
Γ_3			1	1	86	1		1	450	**245**

Tabla 3-5 Asignación de prioridades fijas y tiempos de respuesta de peor caso para el sistema de control de vuelo simplificado, planificado con FP, y con mayor carga en el recurso procesador FGS. (tiempos en ms.)

Para conseguir la planificabilidad del sistema en su nueva situación, proponemos cambiar la política de planificación de FGS por otra que típicamente funcione mejor con cargas elevadas, como EDF. Asumimos la disponibilidad de sincronización en los relojes, y aplicamos la política GC-EDF. Para comprobar si esta solución es válida en este caso, actuamos de la misma manera que en los casos anteriores: modelamos el nuevo sistema con MAST, y aplicamos el algoritmo HOSPA y el análisis de planificabilidad de sistemas heterogéneos. En este caso aplicamos igualmente el análisis basado en *offsets* en los diferentes recursos procesadores (*offset based analysis*). En la Tabla 3-6 se muestra la asignación de parámetros de planificación obtenida por HOSPA, y los tiempos de respuesta de peor caso de esta nueva configuración

La nueva asignación y sus tiempos de respuesta de peor caso se muestran en la Tabla 3-6. En los resultados puede observarse cómo el uso de GC-EDF en el procesador FGS produce una variación en los tiempos de respuesta de peor caso que consigue que el sistema sea planificable. Así, el tiempo de respuesta de peor caso de Γ_3 aumenta con respecto al sistema todo FP, pero a cambio el tiempo de respuesta de Γ_2 disminuye lo suficiente para cumplir su plazo de principio a fin.

	AHRS (P_{ij})	**NAV (P_{ij})**	**FCP (P_{ij})**	**BUS (P_{ij})**	**FGS (SD_{ij})**	**AP (P_{ij})**	**SV (P_{ij})**	**PFD (P_{ij})**	***Di***	***R_i***
Γ_1	1			171	**68,9**	129	1		100	**96,5**
Γ_2		1		86	**200**				200	**195**
Γ_3			1	1	**325**	1		1	450	**367,5**

Tabla 3-6 Asignación de prioridades fijas y plazos globales de planificación (sólo en FGS), y tiempos de respuesta de peor caso, para el sistema de control de vuelo simplificado, con mayor carga en el recurso procesador FGS y planificado por GC-EDF (resto de procesadores planificados por FP) (tiempos en ms.).

3.4 Conclusiones

En el presente capítulo abordamos el primero de los objetivos planteados en esta tesis, desarrollar técnicas para el soporte de sistemas heterogéneos en los que convivan recursos procesadores con diferentes políticas de planificación, en este caso FP y EDF.

En primer lugar abordamos el análisis de planificabilidad de sistemas heterogéneos. Con este objetivo, propusimos un método composicional que adapta las técnicas de análisis que se rigen por una visión holística del sistema (como el análisis holístico de Tindell y Clark), para el soporte de sistemas heterogéneos. Este método composicional se basa en la extracción en cada técnica de análisis con visión holística, del componente que realiza el análisis de cada actividad, y su aislamiento en un nuevo componente al que llamamos Análisis de Actividad. Se definen además los mecanismos por los cuales los diferentes Análisis de Actividad de cada técnica propagan sus resultados. Este método composicional se integró dentro del entorno de herramientas de software libre MAST. Desde la versión 1.4 de esta herramienta, el análisis de sistemas distribuidos en MAST se realiza utilizando el método composicional aquí propuesto.

En la segunda parte del capítulo se afrontó el problema de la asignación de parámetros de planificación en sistemas heterogéneos. Para ello propusimos una metodología que se basa en el reparto de los plazos de principio a fin entre las actividades, y su posterior transformación en el parámetro de planificación adecuado (prioridad fija para FP, plazo

global de planificación para GC-EDF, o plazo local de planificación para FP). Además, haciendo uso de estos principios, integramos los algoritmos HOPA y HOSDA en el nuevo algoritmo heurístico HOSPA para la asignación de parámetros de planificación en sistemas heterogéneos. Estas nuevas técnicas de asignación fueron integradas también dentro del entorno MAST.

Para finalizar el capítulo, se presentó un caso de estudio en el que se mostró la ventaja de poder utilizar la política más adecuada en cada recurso procesador con el objetivo de poder alcanzar la planificabilidad en todo el sistema. En este ejemplo se han utilizado las técnicas de análisis de planificabilidad y de asignación de parámetros de planificación para sistemas heterogéneos que se han propuesto.

4 Generador Automático de Estudios para MAST: GEN4MAST

El campo de los sistemas de tiempo real es un área de investigación activa en el que constantemente se proponen nuevas ideas o aproximaciones. Estas nuevas propuestas pueden tener objetivos muy variados, desde una política de planificación que intente mejorar una nueva métrica como la eficiencia energética, a una técnica de análisis de planificabilidad menos pesimista que las anteriores, o un método para asignar parámetros de planificación más avanzado que mejore la utilización de los recursos.

Normalmente con cada nueva propuesta surge la necesidad de llevar a cabo un estudio de rendimiento en el que se realice una comparación con las técnicas existentes previamente. Típicamente estos estudios se llevan a cabo probando estas técnicas sobre un conjunto sintético de ejemplos. Este método de evaluación aporta una serie de ventajas sobre utilizar implementaciones en sistemas reales:

- Con un conjunto sintético de ejemplos se puede abarcar un abanico más amplio de sistemas y circunstancias, lo que facilita la caracterización y validación de las técnicas.
- En ocasiones las implementaciones reales aún no están disponibles, o son muy costosas. Con estudios sintéticos se permite un desarrollo de las técnicas más rápido y barato: nuevas modificaciones pueden ser probadas inmediatamente, y no se hace necesario tener que esperar a la disponibilidad de las implementaciones reales.
- Estudiar el comportamiento en determinados escenarios de funcionamiento puede ser un proceso complicado para una implementación real, mientras que la configuración de sistemas sintéticos suele ser un proceso comparativamente más sencillo.

Uno de los objetivos que nos planteamos en esta tesis es el de llevar a cabo uno de estos estudios sintéticos, en el que compararemos el rendimiento de algunas de las técnicas de análisis y asignación de parámetros de planificación presentes en MAST, en particular las desarrolladas en esta tesis, sobre un conjunto de ejemplos lo más amplio posible. Teniendo en cuenta la cantidad de variables que definen un sistema de tiempo real (número de procesadores, número de flujos *e2e*, tipo de planificadores, etc.), la variedad de técnicas disponibles en MAST (*holistic*, *offset based*, asignación HOSPA, etc.) y también de parámetros con los que se configuran estas técnicas (parámetros *k* en HOSPA por ejemplo), el número de combinaciones a ejecutar en un estudio de este tipo es potencialmente muy elevado, con crecimiento exponencial a medida que se añadan nuevas variables de estudio. Por lo tanto, para llevar a cabo el estudio aquí propuesto, nos planteamos dos problemas que es necesario resolver:

- Tener la capacidad de generar de forma masiva y sistematizada los sistemas que formen parte del estudio.
- Utilizar algún método de ejecución que permita llevarlo a cabo en un tiempo razonable.

En la actualidad el método para generar modelos MAST consiste en la creación de un fichero de descripción (texto o XML) para cada sistema individualmente. Este fichero se puede crear manualmente con un editor, o mediante el uso de alguna herramienta gráfica como el editor integrado en MAST *gmasteditor*, o de transformación de modelos como UML-MAST [MED01]. El uso de cualquiera de estos mecanismos de generación individual de modelos resulta inadecuado en el estudio que pretendemos realizar por el carácter masivo que posee. Necesitamos una herramienta que automatice tanto la generación de modelos como la aplicación de las técnicas que se quieren estudiar.

En los trabajos que precedieron a esta tesis [RIV08][RIV09] el problema de la creación de ejemplos de test se afrontó realizando generadores *ad-hoc* que automatizaban el proceso completo de creación y ejecución, pero estaban enfocados sólo a un estudio muy concreto, no permitiendo su ampliación o parametrización para abarcar nuevos objetos de estudio.

Existen trabajos en la literatura que han afrontado el problema de la generación sistemática de ejemplos. En [COU11] se presenta la herramienta FORTAS que permite la generación masiva y el análisis de sistemas multiprocesadores con actividades independientes. STORM [URU10] es otra herramienta similar, que permite la definición de varios tipos de planificadores, pero no es capaz de generar sistemas de forma masiva. RTMultiSim [HAN12] es una herramienta que genera masivamente sistemas formados por flujos e2e distribuidos, pero implementa un número reducido de métodos de generación. Otra herramienta que cabe destacar es YARTISS [CHA12], que posee una arquitectura modular para la generación de sistemas multiprocesadores formados por flujos de actividades. Ninguna de estas herramientas se puede usar directamente para nuestro propósito porque, por un lado no se pueden aplicar a sus modelos las técnicas que se quieren evaluar en esta tesis, y por otro, no disponen de la capacidad de variación de parámetros que se quieren verificar en el estudio.

Así, tomando como inspiración estas herramientas, y con la mira puesta en los objetivos de esta tesis, en este capítulo presentamos la herramienta GEN4MAST (*GENerator for MAST*), una herramienta de código abierto que automatiza el proceso completo de generación de sistemas distribuidos sintéticos, la aplicación de las técnicas seleccionadas, y el procesado de los resultados. Aunque esta herramienta está diseñada

para el entorno MAST, tanto la metodología como algunos de sus componentes pueden ser aplicados a otros modelos.

La principal característica que diferencia a GEN4MAST de las herramientas anteriormente citadas es que posee la capacidad de ejecutar los test en un supercomputador. En estudios anteriores [RIV08][RIV09][RIV12] se observó que existen sistemas sobre los cuales llevar a cabo el análisis de planificabilidad puede conllevar horas de ejecución en un PC convencional. Teniendo en cuenta que la evaluación de un solo test puede estar compuesta por miles de este tipo de ejecuciones, poder distribuir esta ejecución en un supercomputador abre la puerta a poder realizar evaluaciones extensivas en tiempos razonables. Aunque un supercomputador pueda ser un recurso con coste elevado, durante los últimos tiempos se está observando cómo cada vez son más habituales en las instituciones de investigación debido a los claros beneficios que aportan.

Al estar basado en MAST, GEN4MAST hereda todas sus prestaciones, siendo las más destacables las siguientes:

- MAST proporciona un modelo rico con el que describir sistemas de tiempo real, y está alineado con el estándar MARTE [MARTE] de la OMG.
- MAST está constituido por un conjunto de herramientas de código abierto, y proporciona una metodología con la que añadir nuevas técnicas de análisis y asignación de parámetros de planificación. (ver Capítulo 3 y [RIV11]).

El presente capítulo ahondará sobre las características y funcionamiento de GEN4MAST, comenzando en la Sección 4.1 con una descripción general de su arquitectura, para a continuación detallar cada una de sus fases de ejecución en las Secciones 4.2, 4.3 y 4.4. Para finalizar, en la Sección 4.5 se llevará a cabo un estudio de los métodos de generación integrados en GEN4MAST, que además servirá como muestra de un caso de uso típico.

4.1 Arquitectura de GEN4MAST

GEN4MAST se construye alrededor de la herramienta de análisis y optimización MAST a modo de lo que en inglés se denomina *wrapper*. La herramienta de análisis y optimización de MAST está diseñada para trabajar con un sistema cada vez: acepta como entrada un fichero con la descripción de un sistema según el modelo MAST, a continuación se ejecutan sobre dicho sistema las técnicas que el usuario haya escogido, y devuelve un fichero con los resultados. GEN4MAST automatiza este proceso completo, coordinando la ejecución de múltiples llamadas a las herramientas de análisis y optimización de MAST (potencialmente millones de ejecuciones).

La arquitectura de GEN4MAST se muestra en la Figura 4-1 y su funcionamiento se divide en tres fases, que se introducen brevemente a continuación:

1. Fase de generación: En esta fase el Módulo Generador genera todos los ficheros necesarios para ejecutar el estudio; esto incluye los ficheros descriptores de los sistemas (utilizando el modelo MAST), y los *scripts* de ejecución de la herramienta de análisis y optimización de MAST. Las características de estos ficheros se describen en el fichero de configuración de GEN4MAST.

2. Fase de ejecución: Durante esta fase se llevan a cabo las ejecuciones de la herramienta de análisis y optimización de MAST, que pueden realizarse en un PC local o en un supercomputador (en caso de estar disponible). Esta labor la realiza el Módulo Despachador de Trabajos. La ejecución en el supercomputador puede ser configurada, cuyos parámetros se especifican en el fichero de configuración de GEN4MAST. Cada ejecución de la herramienta de análisis y optimización de MAST genera un fichero de resultados, con un formato propio de MAST.
3. Fase de resultados: Una vez que las ejecuciones del estudio han terminado, el Módulo Procesador de Resultados de GEN4MAST recopila todos los resultados, y los combina en una base de datos indexada de fácil acceso. Esta base de datos utiliza el formato HDF5 a través de la librería Pytables [PYTAB], que permite manejar grandes cantidades de datos de manera eficiente, y posee gran aceptación en el mundo científico. GEN4MAST además suministra funciones para acceder a estos resultados de manera sencilla, mediante la creación de tablas y gráficas.

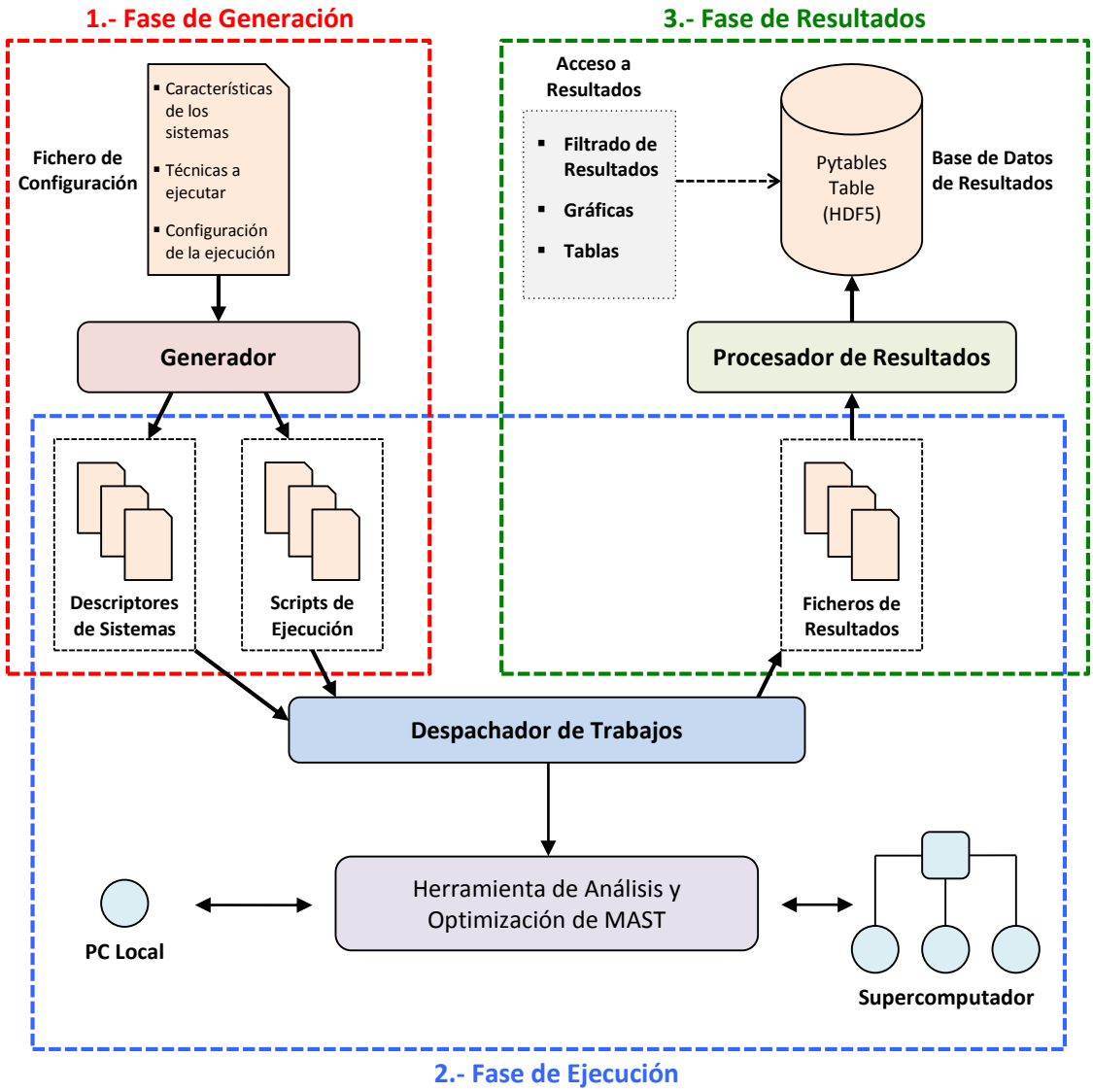


Figura 4-1 Esquema de funcionamiento de GEN4MAST

En el fichero de configuración se especifican las características de los estudios que se quieren llevar a cabo. Para este fichero se ha definido un lenguaje de configuración especifico. El fichero de configuración de GEN4MAST está formado por parámetros de configuración agrupados en cuatro secciones (ver Figura 4-2) en función de la labor que desempeñan. Las cuatro secciones en las que se divide el fichero son:

- GENERATOR y ARCHITECTURE: En estas dos secciones se agrupan los parámetros de configuración que definen las características de los sistemas que se van a generar.
- EXECUTION: En esta sección se especifican qué técnicas se van a aplicar sobre los sistemas generados (técnicas de análisis, optimización, y parámetros que las configuran).
- CLUSTER: En esta sección se configuran las particularidades de la ejecución dentro de un supercomputador.

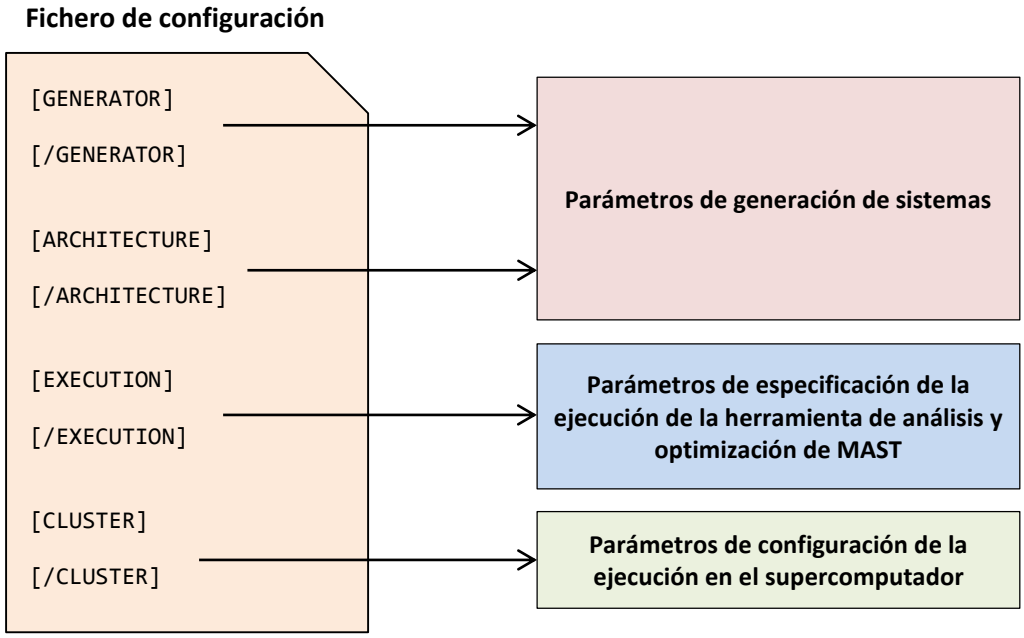


Figura 4-2 Estructura del fichero de configuración de GEN4MAST.

La sintaxis con la que se especifican los parámetros en el fichero de configuración de GEN4MAST consiste en el nombre del parámetro de configuración, seguido de uno o varios valores separados por espacios. GEN4MAST generará un estudio que abarca todas las combinaciones de estos valores asignados a todos los parámetros de configuración.

Los parámetros de configuración definidos, junto con su significado y funcionamiento, se irán tratando a continuación en las descripciones de las diferentes fases en las que se divide el proceso de generación de GEN4MAST. En estas descripciones utilizaremos una nomenclatura específica para describir la sintaxis de cada parámetro de configuración. Si un parámetro de configuración admite una lista de valores, la lista se denotará con puntos suspensivos. Por ejemplo, un parámetro concreto llamado PARÁMETRO_X que admita una lista de valores, lo describiremos como:

PARÁMETRO_X valor1 ...

donde “valor1” es el primer valor de la lista.

Algunos parámetros de configuración admiten únicamente un conjunto limitado de valores posibles. Tal situación la describimos especificando los valores admitidos separados por el carácter "/". Por ejemplo, un parámetro concreto llamado PARÁMETRO_Y que admita una lista de valores con opciones [OPCION1, OPCION2, OPCION3], lo describiremos como:

PARÁMETRO_Y OPCION1/OPCION2/OPCION3 ...

GEN4MAST está programado en el lenguaje Python. La alta aceptación de este lenguaje contribuye a la disponibilidad de multitud de librerías como Pytables que aceleran el desarrollo de herramientas como GEN4MAST. Adicionalmente, Python es un lenguaje interpretado multiplataforma, lo que facilita el desarrollo de programas que como GEN4MAST pueden ser utilizados en diferentes entornos (PC convencional o supercomputador) con potencialmente distintas arquitecturas.

4.2 Fase de generación

En la fase de generación se establece el entorno de ejecución del estudio que se va a realizar. Este entorno está compuesto por los ficheros descriptores de los sistemas a estudiar y los *scripts* de ejecución de MAST, todos organizados en una estructura de directorios concreta. En las siguientes secciones se describirá la generación de los sistemas (Sección 4.2.1), la generación de los *scripts* de ejecución (Sección 4.2.2), y la estructura del entorno de ejecución del estudio (Sección 4.2.3).

4.2.1 Generación de sistemas

El conjunto de sistemas correspondientes a un caso de estudio concreto se crea mediante una serie de reglas de generación, siguiendo los valores establecidos por los parámetros de configuración especificados en el fichero de configuración de GEN4MAST. Estos parámetros de configuración se especifican en la sección GENERATOR dentro del fichero de configuración de GEN4MAST, cuyo comienzo y finalización se especifican con los delimitadores "[GENERATOR]" y "[/GENERATOR]" respectivamente. En la Figura 4-3 se muestran los parámetros disponibles en esta sección.

Los parámetros de generación de sistemas incluidos en GEN4MAST cubren la mayoría de las características que afectan al comportamiento temporal de los sistemas de tiempo real, y que se pueden modelar con MAST. Estos parámetros hacen referencia a características tales como el número de procesadores en el sistema, número de flujos e2e, rango de los periodos, etc. En las próximas secciones se describe el proceso de generación de los sistemas, y el significado de estos parámetros.

En la versión inicial de GEN4MAST se incluyen los elementos más representativos del modelo MAST que permiten cubrir los aspectos necesarios para realizar el conjunto de estudios propuestos en esta tesis. Así por ejemplo, no se generarán redes de comunicación ni los menajes que se puedan transmitir a través de las mismas, ya que sin pérdida de generalidad, los mensajes y las redes se pueden considerar como segmentos de

código y procesadores desde el punto de vista del análisis de planificabilidad (ver Sección 1.7.1).

Las descripciones de los sistemas generados por GEN4MAST utilizan la versión 1 del modelo MAST, ya que ésta es la utilizada en la herramienta de análisis y optimización de MAST de forma nativa.

```
[GENERATOR]
POPULATION
N_FLOWS
N_STEPS
SINGLE_FLOWS
FIXED_LENGTH
N_PROCESSORS
REPETITION
SCHEDULING_POLICY
PERIOD_BASE
PERIOD_RATIOS
PERIOD_DISTRIBUTION
DEADLINES
UTILIZATION_START
UTILIZATION_STEP
UTILIZATION_STOP
UNIFORM_UTILIZATION
WORKLOAD
BEST_CASE
[/GENERATOR]
```

Figura 4-3 Parámetros de generación de sistemas en GEN4MAST

4.2.1.1 Número de flujos e2e, y número de actividades en cada flujo

En el modelo de descripción de sistemas de MAST, la carga computacional de los sistemas viene dada por las respuestas a los eventos externos del sistema que se representan por medio de los flujos e2e, cuyas actividades pueden estar distribuidas en un número de recursos procesadores. Por lo tanto el establecimiento del número de flujos e2e, y el número de actividades de cada flujo (longitud del flujo), son dos parámetros fundamentales a la hora de definir un sistema en MAST.

Los parámetros que definen el número de flujos e2e y sus longitudes, junto con su sintaxis, se muestran en la Figura 4-4. El parámetro N_FLOWS establece el número de flujos e2e en el sistema, y N_STEPS el número máximo de actividades en cualquier flujo.

```
N_FLOWS a …
N_STEPS b …
SINGLE_FLOWS c …
FIXED_LENGTH True/False …
```

Figura 4-4 Parámetros de generación y localización de las actividades

Habitualmente los flujos e2e de un sistema no poseen la misma longitud. En GEN4MAST se permite modelar esta variabilidad en las longitudes mediante el parámetro booleano FIXED_LENGTH. Para un valor concreto *b* de N_STEPS, si FIXED_LENGTH es True, todos los flujos tendrán *b* actividades. Si FIXED_LENGTH es False, la longitud se calcula para cada flujo e2e de forma aleatoria, siguiendo una distribución uniforme en el rango [2,*b*].

Adicionalmente, también habrá sistemas en los que existan flujos e2e con una sola actividad (flujos e2e simples). Esta característica se ha querido configurar aparte con el parámetro SINGLE_FLOWS para tener un control más fino sobre la influencia de esta circunstancia en los sistemas. Así, independientemente de los valores especificados en N_STEPS y FIXED_LENGTH, con SINGLE_FLOWS se especifica el porcentaje de flujos e2e que tendrán longitud 1. Para un valor *c* concreto de SINGLE_FLOWS, el *c*% de flujos e2e (redondeado al entero más próximo) tendrá una única actividad.

4.2.1.2 Localización de las actividades

La localización de las actividades es el proceso que asigna cada actividad a un recurso procesador. La actividad residirá en dicho recurso de manera estática. Este proceso se controla mediante los parámetros mostrados en la Figura 4-5.

```
SCHEDULING_POLICY FP/EDF …
N_PROCESSORS np …
REPETITION YES/NO/CONS …
```

Figura 4-5 Parámetros de localización de las actividades

El parámetro N_PROCESSORS establece el número de recursos procesadores disponibles en el sistema. Las actividades se localizan en los recursos procesadores de forma aleatoria, siguiendo unas directivas que se establecen con el parámetro REPETITION. Los valores válidos para este parámetro son:

- YES: Para cada actividad, el recurso en el que se localiza se selecciona aleatoria y uniformemente en el rango [1,N_PROCESSORS]. Por lo tanto, un mismo flujo e2e puede recorrer el mismo recurso procesador más de una vez. A este tipo de localización la denominamos localización aleatoria.
- NO: Se intenta evitar que los flujos e2e recorran el mismo recurso procesador en más de una ocasión. A este tipo de localización la denominamos pseudo-aleatoria. Este comportamiento sólo es posible si la longitud del flujo e2e es menor o igual que el número de recursos procesadores, de lo contrario se lleva a cabo una localización aleatoria uniforme (valor YES).
- CONS: Las actividades se localizan de manera aleatoria uniforme, pero se evita que dos actividades consecutivas residan en el mismo recurso procesador. A este tipo de localización la denominamos aleatoria sin repeticiones consecutivas. Cuando el número de recursos es sólo 1, esta restricción es imposible de cumplir, y se revierte a un valor YES.

El parámetro SCHEDULING_POLICY establece la política de planificación de todos los recursos procesadores del sistema. En esta primera versión de GEN4MAST no se soporta la generación de sistemas heterogéneos FP+EDF. El parámetro SCHEDULING_POLICY admite dos tipos de políticas de planificación, FP o EDF. El modelo MAST en su versión 1, que es el utilizado en GEN4MAST, no soporta la especificación del subtipo de planificación EDF para sistemas distribuidos (LC-EDF o GC-EDF) en la descripción del sistema. Esta especificación se realiza como parámetro de ejecución de la herramienta de análisis y optimización de MAST. En GEN4MAST la especificación del subtipo de planificación EDF se realiza en la definición de los *scripts* de ejecución la herramienta de análisis y optimización de MAST (ver Sección 4.2.2).

4.2.1.3 Generación de los periodos

Todos los eventos que activan los flujos e2e de los sistemas generados por GEN4MAST son periódicos. Es habitual encontrarnos con sistemas de tiempo real en los que co-existan actividades que requieran periodos muy bajos con otras actividades ordenes de magnitud más frecuentes [LIJ00][BUR06]. En la literatura se ha observado [DAV08][PAL03] cómo la presencia de diferentes órdenes de magnitud en los valores de los periodos de las actividades de un mismo sistema tiene grandes implicaciones en su planificabilidad. Por lo tanto, desde el punto de vista de una herramienta para la generación de estudios de planificabilidad como GEN4MAST, resulta interesante tener la capacidad de establecer diferentes relaciones entre el periodo máximo y mínimo como otro parámetro más de la generación. La especificación de los periodos en la generación se establecen con los parámetros de configuración mostrados en la Figura 4-6

```
PERIOD_BASE Tbase
PERIOD_RATIOS pr …
PERIOD DISTRIBUTION CUSTOM-UNIFORM/LOG-UNIFORM …
```

Figura 4-6 Parámetros para la generación de los periodos

Para unos valores *pr* y *Tbase* determinados para los parámetros PERIOD_RATIO y PERIOD_BASE respectivamente, el periodo para cada flujo e2e se selecciona de forma aleatoria en el rango [*Tbase*, *Tbase*pr*]. GEN4MAST suministra dos tipos distintos de función de distribución de probabilidad con la que elegir el valor del periodo dentro del rango establecido. El tipo de distribución que se utilizará se establece con el parámetro PERIOD_DISTRIBUTION.

La función de distribución más sencilla que podemos utilizar es una distribución uniforme, esto es, cualquier valor en el rango [*Tbase*, *Tbase*pr*] tiene la misma probabilidad de ser seleccionado. A priori puede parecer que esta distribución es la apropiada para todos los casos, pero cuando el rango de selección abarca varios órdenes de magnitud, una distribución uniforme premia la selección de periodos cercanos al valor máximo. Este hecho se puede ilustrar con un ejemplo sencillo, en el que se seleccionan 100000 periodos de forma uniforme en el rango [1, 10^6]. En la Figura 4-7 se representa el número de periodos seleccionados en cada uno de los 6 órdenes de magnitud que componen el rango total.

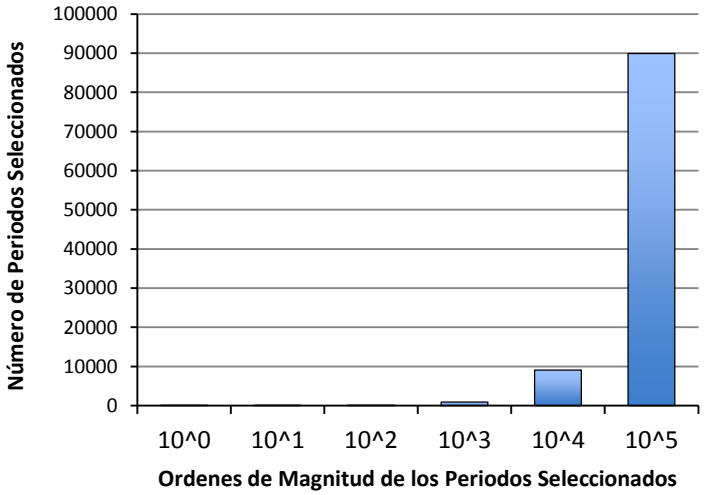


Figura 4-7 Histograma de los periodos seleccionados de forma uniforme en el rango [1, 10^6]

A la vista de la Figura 4-7, una distribución uniforme no es capaz de modelar eficientemente sistemas en los que aparezcan los diferentes órdenes de magnitud especificados en el rango de estudio. Para solventarlo, en GEN4MAST se utiliza una distribución uniforme personalizada que funciona en dos etapas. En la primera, se le asigna a cada flujo un periodo uniforme en el rango [*Tbase*, *Tbase*pr*]. A continuación, se eligen aleatoriamente dos flujos e2e en el sistema. Al primero de ellos se le asigna un periodo *Tbase*, y al segundo un periodo *Tbase*pr*. A esta asignación la llamamos *Custom-Uniform*, y en GEN4MAST se utiliza asignando el valor CUSTOM-UNIFORM al parámetro PERIOD_DISTRIBUTION. Si el sistema tuviera un único flujo, su periodo se mantiene al valor obtenido en la distribución uniforme inicial.

Una asignación *Custom-Uniform* de periodos no resuelve por completo el problema de aparición de todos los órdenes de magnitud especificados en el rango de estudio, ya que todos los flujos, salvo dos, tendrán tendencia a poseer periodos en la parte superior del rango. Para solventarlo, en GEN4MAST se incluye un segundo tipo de distribución, esta vez logarítmica [EMB10], a la que llamamos *Log-Uniform*. Esta distribución asigna periodos de forma que todos los órdenes de magnitud del rango de estudio aparecen uniformemente representados. El método *Log-Uniform* se utiliza cuando el parámetro PERIOD_DISTRIBUTION es LOG-UNIFORM.

Para mostrar su funcionamiento, realizamos un ejemplo en el que seleccionamos 100000 periodos con la distribución *Log-Uniform* en el rango [1, 10^6]. En la Figura 4-8 mostramos el histograma de los periodos obtenidos, y como se puede observar los periodos se reparten de forma equitativa entre todos los órdenes de magnitud.

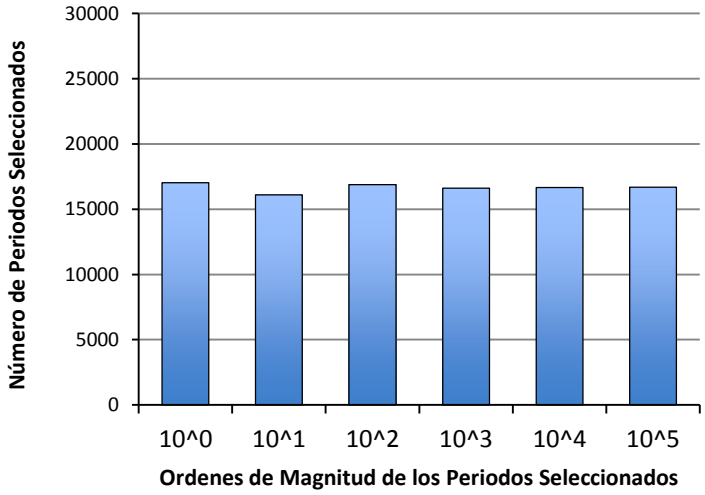


Figura 4-8 Histograma de los periodos seleccionados con la distribución *Log-Uniform* en el rango [1, 10^6]

4.2.1.4 Selección de los plazos de principio a fin

En el modelo MAST el plazo de principio a fin (*D*) es el requisito temporal que debe cumplir cada flujo e2e en el sistema, y por lo tanto, es un parámetro de vital importancia.

En general, los sistemas distribuidos de tiempo real poseen, en función de la aplicación, requisitos temporales de diferentes órdenes relativos a los periodos de activación. Por ello, resulta interesante tener la capacidad de estudiar el comportamiento

de las técnicas de análisis y optimización ante distintos valores de plazos y observar cómo se adaptan a ellos. Para realizar este tipo de estudio, en GEN4MAST se pueden especificar diferentes plazos de principio a fin con el parámetro DEADLINES, cuya sintaxis y valores permitidos se muestran en la Figura 4-9.

```
DEADLINES K/T/NT/Q1/Q2/Q3/T1/T2/2NT/RANDOM …
```

Figura 4-9 Parámetros de generación de los plazos de principio a fin.

En los sistemas con flujos e2e simples (de longitud 1), es común que los plazos sean igual a los periodos. En cambio, en los sistemas con flujos e2e de mayor longitud que 1, es habitual que los plazos de principio a fin sean mayores que los periodos. En concreto, suelen ser un valor que escala el periodo por la longitud del flujo e2e ($D_i=N_i*T_i$). Teniendo en cuenta esta particularidad, en GEN4MAST los plazos de principio a fin de cada flujo e2e se establecen en relación a su periodo. Se definen dos modos de generación. En el primer modo, el valor registrado en el fichero de configuración para el parámetro DEADLINES es un entero K que establece que los plazos de principio a fin de todos los flujos e2e del sistema sean $D_i=K*T_i$.

En el segundo modo, los plazos de principio a fin se asignan en puntos preestablecidos del segmento $D_i=[T_i, N_i*T_i]$. Estos puntos pueden ser el primer, segundo y tercer cuarto del segmento (denotados como Q1, Q2 y Q3 respectivamente), el primero y segundo tercio del segmento (denotados como T1 y T2 respectivamente), los extremos del segmento (nombrados T y NT), o un punto aleatorio del segmento seleccionado de forma uniforme (denotado RANDOM). El segmento junto con estos valores preestablecidos se representan en la Figura 4-10. Adicionalmente, se añade otro punto preestablecido fuera de dicho segmento, situado en $D_i=2*N_i*T_i$, denotado como 2NT.

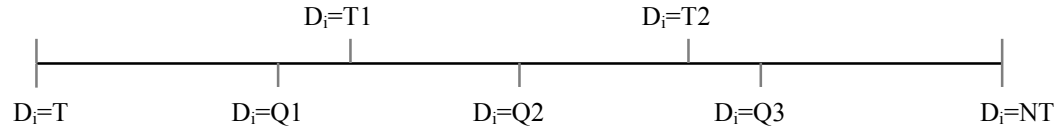


Figura 4-10 Puntos preestablecidos de Plazos de principio a fin en el segmento $[T_i, N_i*T_i]$

A continuación, se muestra a modo de resumen el conjunto de los posibles valores para el parámetro DEADLINES, y el plazo de principio a fin que se asigna a cada uno de ellos:

Valor del parámetro DEADLINES	Valor de los plazos e2e de todos los flujos e2e
K	$D_i = K \cdot T_i$
T	$D_i = T_i$
NT	$D_i = N_i \cdot T_i$
Q1	$D_i = \frac{T_i}{4}(N_i + 3)$
Q2	$D_i = \frac{T_i}{2}(N_i + 1)$
Q3	$D_i = \frac{T_i}{4}(3 \cdot N_i + 1)$
T1	$D_i = \frac{T_i}{3}(N_i + 2)$
T2	$D_i = \frac{T_i}{3}(2 \cdot N_i + 1)$
2NT	$D_i = 2 \cdot N_i \cdot T_i$
RANDOM	$D_i = random(T_i,\ N_i \cdot T_i)$

Tabla 4-1 Valores admitidos por el parámetro de generación DEADLINES, y las fórmulas asociadas.

4.2.1.5 Generación de Series de Utilizaciones

En los estudios sobre sistemas de tiempo real es habitual observar el comportamiento de determinados parámetros (por ejemplo el tiempo de respuesta) a medida que se aumenta la utilización del sistema. Así, Lehoczky *et al* [LEH89] establecieron una métrica con la que determinar la capacidad de planificación de un determinado algoritmo. Este método consiste en aumentar progresivamente la utilización de un sistema dado, mediante el escalado de los tiempos de ejecución, hasta llegar a un punto en el que el sistema deje de ser planificable. Al último punto de utilización planificable se le conoce como utilización de ruptura, o utilización máxima planificable.

Siguiendo el criterio de [LEH89], en GEN4MAST, para cada una de las combinaciones de parámetros de generación de sistemas especificados previamente, se establece un Sistema Raíz en el que sólo faltan por determinar los tiempos de ejecución de peor caso de cada actividad (C_{ij}). Estos tiempos de ejecución se calculan de manera progresiva de forma que se obtiene una serie de sistemas con las características del Sistema Raíz, que abarcan el rango de utilizaciones totales preestablecidas por el usuario. A esta serie la llamamos Serie de Utilizaciones, gracias a la cual podemos estudiar la evolución de los sistemas en función de la utilización, o determinar la utilización máxima planificable.

Antes de describir el proceso de generación de las Series de Utilizaciones, repasamos las definiciones de la utilización de la actividad, del recurso procesador, y del sistema. Definimos la utilización de una actividad τ_{ij} como la división entre su tiempo de ejecución de peor caso, y el periodo del flujo e2e en el que se sitúa (Ecuación (4-1)). Definimos la utilización de un recurso procesador PR_k como la suma de las utilizaciones de las actividades contenidas en él, tal y como se especifica en la Ecuación (4-2). La utilización

total de un sistema distribuido la definimos en el rango [0-100] (en %) como la media entre las utilizaciones de los recursos procesadores que lo componen (multiplicado por 100 para obtener términos porcentuales), como se muestra en la Ecuación (4-3), donde N_{PR} es el número de recursos procesadores en el sistema.

$$U_{ij} = \frac{C_{ij}}{T_i} \tag{4-1}$$

$$U_{PR_k} = \sum_{\forall i,\, j \in PR_k} U_{ij} \tag{4-2}$$

$$U = \frac{\sum_{\forall k} U_{PR_k}}{N_{PR}} \cdot 100 \tag{4-3}$$

Los parámetros de GEN4MAST que controlan la generación de la utilización, y por extensión también los tiempos de ejecución de peor caso de las actividades, se muestran en la Figura 4-11, junto con su sintaxis.

```
UTILIZATION_START UStart
UTILIZATION_STEP UStep
UTILIZATION_STOP UStop
UNIFORM_UTILIZATION True/False …
WORKLOAD UUNIFAST/SCALE-WCET …
```

Figura 4-11 Parámetros para la generación de las Series de Utilizaciones

El rango de utilizaciones totales a generar en una serie se especifica con tres valores: la utilización mínima, la utilización máxima, y el paso (incremento de utilización con el que se van obteniendo los valores intermedios). Estos se establecen con los parámetros del fichero de configuración llamados UTILIZATION_START, UTILIZATION_STOP y UTILIZATION_STEP respectivamente.

Una vez conocido el rango de utilizaciones totales a abarcar, se hace necesario algún método con el que calcular los tiempos de ejecución de peor caso de las actividades de forma que los recursos procesadores tengan la utilización requerida. En GEN4MAST se incluyen dos métodos: *SCALE-WCET* y *UUnifast* [BIN05]. Se especifican con el parámetro de configuración WORKLOAD con los valores SCALE-WCET y UUNIFAST respectivamente. Para poder aplicar estos métodos asumimos que todos los flujos e2e ya poseen un periodo asignado, y las actividades ya han sido asignadas a un recurso procesador.

El método *SCALE-WCET* es una primera aproximación intuitiva al problema del cálculo de los tiempos de ejecución de peor caso. Si todas las actividades del recurso PR_k tuvieran la misma utilización U_{ij}, la utilización total de dicho recurso procesador se puede calcular con la Ecuación (4-4), donde n es el número de actividades en el recurso PR_k.

$$U_{PR_k} = n \cdot U_{ij} \tag{4-4}$$

A partir de la Ecuación (4-4) se obtiene un método con el que calcular los tiempos de ejecución de las actividades en un recurso procesador PR_k concreto, de forma que alcance una utilización U_{PR_k}, estableciendo que todas las actividades posean la misma utilización.

Este es el método de cálculo de la utilización que llamamos *SCALE-WCET*, y se formaliza en la Ecuación (4-5).

$$C_{ij}^{SCALE-WCET} = \frac{U_{PR_k} \cdot T_i}{n} \tag{4-5}$$

Con el algoritmo *SCALE-WCET* podemos calcular de una manera sencilla los tiempos de ejecución de peor caso para que un recurso procesador resulte cargado con la utilización que le especifiquemos. Una característica de esta aproximación es que todas las actividades poseen la misma utilización, hecho que en sistemas monoprocesadores se ha comprobado que puede provocar que se generen sistemas con cierto sesgo que penaliza la utilización máxima planificable [BIN05][BIN03].

Con el objetivo de generar sistemas con utilizaciones menos sesgadas, Bini y Butazzo diseñaron el algoritmo *UUnifast* [BIN05], con gran aceptación entre las herramientas de generación de sistemas. *UUnifast* calcula los tiempos de ejecución de las actividades de un recurso procesador de forma que sus utilizaciones sumen un valor especificado U_{PR_k}, pero a diferencia de *SCALE-WCET*, las utilizaciones de las actividades siguen una distribución uniforme. Al aumentar la dispersión de las utilizaciones de las actividades se minimiza el sesgo en los cálculos de la utilización máxima planificable [BIN05]. Aunque *UUnifast* fue diseñado para actuar en sistemas monoprocesadores, como GEN4MAST lo aplica individualmente en cada procesador, es compatible con nuestro modelo de sistema distribuido.

SCALE-WCET y *UUnifast* nos proveen de dos técnicas con las que cargar cada recurso procesador de forma que alcance una utilización deseada. Utilizando estas técnicas, GEN4MAST permite aplicar dos métodos distintos con los que distribuir la utilización entre los diferentes recursos procesadores de forma que se alcance una utilización total específica en el sistema. Estos métodos controlan la evolución de la utilización total en el rango de utilizaciones totales a generar. La selección de estos dos métodos de evolución se controlan mediante el parámetro booleano UNIFORM_UTILIZATION.

```
U = UTILIZATION_START
loop
        for P in Todos los recursos del sistema
        loop
                # Asigna utilización U al recurso P
                Metodo_Asignación_Cij(P,U)
        end loop
        # Almacena sistema en la Serie de Utilizaciones
        Almacena_Sistema()
        U = U + UTILIZATION_STEP
        exit when U > UTILIZATION_TOP
end loop
```

Figura 4-12 Pseudocódigo de la regla de evolución del reparto uniforme de utilización entre los recursos

En el primer método de evolución, todos los recursos procesadores del sistema poseen la misma utilización en cada uno de los pasos de la Serie de Utilizaciones totales. De esta forma, el valor de la utilización total del sistema coincide con la de cada uno de sus recursos procesadores. Para crear una utilización total *U*, se utiliza alguno de los métodos de generación de tiempos de ejecución de peor caso de forma que cada recurso procesador tenga utilización *U*. A este método de evolución lo llamamos de reparto

uniforme de la utilización, y se describe con el pseudocódigo de la Figura 4-12. En GEN4MAST se utiliza este método asignando *True* al parámetro UNIFORM_UTILIZATION.

El segundo método de evolución consiste en aumentar la utilización de un único recurso procesador, elegido aleatoriamente, para cada una de las utilizaciones totales del rango de estudio. Así, para que la utilización total del sistema aumente en el valor dado por UTILIZATION_STEP (en %), el aumento de la utilización del recurso seleccionado debe ser UTILIZATION_STEP*NP, donde NP es el número de recursos procesadores del sistema. En esta evolución, llamada no uniforme, se debe verificar que ningún recurso procesador resulte cargado por encima del 100%. El funcionamiento se esquematiza en el pseudocódigo de la Figura 4-13, y se corresponde con el valor *False* del parámetro UNIFORM_UTILIZATION.

```
U = UTILIZATION_START
for P in Todos los recursos del sistema
loop
       Metodo_Asignación_Cij(P, U)
end loop
# Almacena sistema en la Serie de Utilizaciones
Almacena_Sistema()

loop
       U = U + UTILIZATION_STEP
       exit when U > UTILIZATION_TOP
       loop
              P = random(1..NP) # Número aleatorio en el rango 1,NP
              UP = Utilización(P) # Utilización del recurso P
              if (UP + UTILIZATION_STEP*NP) < 100
                     exit
              end if
       end loop
       # Asigna utilización (UP + UTILIZATION_STEP*NP) al recurso P
       Metodo_Asignación_Cij(P, UP + UTILIZATION_STEP*NP)

       # Almacena sistema en la Serie de Utilizaciones
       Almacena_Sistema()
end loop
```

Figura 4-13 Pseudocódigo de la regla de evolución del reparto no uniforme de utilización

4.2.1.6 Tiempos de ejecución de mejor caso

Adicionalmente al tiempo de ejecución de peor caso (C_{ij}), las actividades también pueden tener un tiempo mínimo de ejecución no nulo, que llamamos tiempo de ejecución de mejor caso (Cb_{ij}). Acotar el tiempo de ejecución de una actividad, estableciendo sus tiempos mínimo y máximo de ejecución, reduce el *jitter* calculado por las técnicas de análisis de planificabilidad (ver Sección 2.1), y por lo tanto, produce una mejora en la estimación de los tiempos de respuesta y en la planificabilidad (observado para sistemas planificados por FP en [GUT96]). Es por ello que estudiar el efecto de distintas cotas mínimas para el tiempo de ejecución puede resultar también interesante.

```
BEST_CASE bc …
```

Figura 4-14 Parámetro para el control del número de repeticiones

En GEN4MAST se incluye el parámetro de generación BEST_CASE, con el que se controlan los tiempos de ejecución de mejor caso en la generación de las Series de Utilizaciones. El formato de este parámetro se muestra en la Figura 4-14. Cada valor asignado a BEST_CASE es un entero en el rango [0,100] que establece el valor de todos los tiempos de ejecución de mejor caso como un porcentaje del tiempo de ejecución de peor caso. Su funcionamiento se formaliza en la siguiente ecuación, para un valor de BEST_CASE igual a *bc*:

$$Cb_{ij} = \frac{bc \cdot C_{ij}}{100} \tag{4-6}$$

4.2.1.7 Control de relevancia estadística

Como se ha comprobado en las secciones anteriores, la generación de los sistemas posee cierto componente aleatorio. Para establecer algunos parámetros concretos se utilizan funciones de probabilidad, como por ejemplo para la selección de los periodos. En GEN4MAST se utiliza el método estadístico Monte Carlo, en el que se repite el proceso de generación un número determinado de veces, con el fin de obtener varias muestras dentro de estas funciones de distribución, y poder así obtener resultados estadísticamente relevantes. El número de repeticiones, o lo que es lo mismo, el número de sistemas con los mismos parámetros de generación, se especifica mediante el valor que sigue al parámetro POPULATION, cuyo formato se destaca en la Figura 4-15.

```
POPULATION N
```

Figura 4-15 Parámetro para el control del número de repeticiones

A todos los sistemas generados en cada una de las repeticiones se le asigna un número de identificación diferente de manera secuencial, que se corresponde con la iteración en la que se generó. Este identificador se utiliza para discriminar los sistemas de distintas repeticiones dentro de la base de datos de resultados.

4.2.1.8 Proceso de generación de los sistemas

Cuando en algún parámetro de generación se especifica una lista de posibles valores, GEN4MAST generará sistemas con todas las combinaciones de valores suministrada. Todos estos sistemas generados no son totalmente independientes entre sí. Dentro del conjunto de parámetros de generación distinguimos dos grupos: los parámetros de arquitectura primarios, y los secundarios.

Los parámetros de arquitectura primarios especifican la arquitectura básica del sistema, y son el número de flujos e2e, el número de actividades en cada flujo, y la localización de las actividades en un recurso procesador. Estos parámetros son N_FLOWS, SINGLE_FLOWS, N_STEPS, FIXED_LENGTH, N_PROCESSORS y REPETITION. Con cada combinación de valores de los parámetros primarios, se especifica un sistema primario. El proceso de generación del sistema primario, junto con el control de las repeticiones y la asignación de identificadores, se describe en el

pseudocódigo de la Figura 4-16, en el que se puede apreciar cómo en cada iteración se genera un sistema primario *S*.

```
for ID in 1..POPULATION
  for N in N_FLOWS
    for M in SINGLE_FLOWS
      for NT in N_STEPS
        for F in FIXED_LENGTH
          for P in N_PROCESSORS
            for R in REPETITION
              S = Sistema_Primario (ID,N,M,NT,F,P,R)
```

Figura 4-16 Pseudocódigo del Proceso de Generación del Sistema Primario en GEN4MAST

Los parámetros de arquitectura secundarios son el resto de parámetros de generación. Estos parámetros secundarios actúan sobre un sistema primario ya establecido. El resultado es que los sistemas finales se generan a partir de las combinaciones de los parámetros secundarios sobre una base común determinada por cada sistema primario generado.

Este proceso de generación se detalla en el pseudocódigo de la Figura 4-17, que comienza a partir de la creación de un sistema primario *S*, para iterar a continuación sobre los valores suministrados para la asignación de los periodos, plazos y políticas de planificación. Estos valores se van utilizando sucesivamente para completar la definición de las características del sistema. El último paso es el de iterar los valores de los parámetros de generación de utilizaciones (WORKLOAD y UNIFORM_UTILIZATION), en conjunción con el parámetro de tiempos de ejecución de mejor caso (BEST_CASE), y calcular los tiempos de ejecución de las actividades, creando Series de Utilizaciones según lo descrito en la Sección 4.2.1.5.

```
…
S = Sistema_Primario (ID,N,M,NT,F,P)

for PR in PERIOD_RATIOS
  for PD in PERIOD_DISTRIBUTION
    for D in DEADLINES
      for SP in SCHEDULING_POLICY
        for W in WORKLOAD
          for UU in UNIFORM_UTILIZATION
            for BC in BEST_CASE

              #Crea Sistema Raíz
              SB = Sistema_Raíz(S,PR,PD,D,SP)

              Umin = UTILIZATION_START
              Ustep = UTILIZATION_STEP
              Utop = UTILIZATION_TOP

              #Crea Serie de Utilizaciones
              Crear_Serie_de_Utilizaciones(SB,Umin,Ustep,Utop,W,UU,BC)
```

Figura 4-17 Pseudocódigo del Proceso de Generación de Series de Utilizaciones en GEN4MAST

4.2.1.9 Generación de sistemas con arquitectura fija

Con los parámetros que hemos descrito previamente se pueden especificar sistemas de tiempo real modelados en MAST, pero gran parte de los valores concretos de los

parámetros eran calculados por GEN4MAST de forma aleatoria, siguiendo alguna distribución de probabilidad. Por ejemplo, la especificación de los periodos se llevaba a cabo definiendo un rango, y la función de distribución a utilizar. Los valores concretos de los periodos eran calculados por GEN4MAST en tiempo de ejecución.

Existen situaciones en las que el usuario puede desear generar sistemas que posean unas características concretas, y generar sistemas masivos a partir de esa base. Por ejemplo, se puede desear analizar cómo afectan distintos tipos de distribución de carga en una arquitectura de tipo cliente servidor, en la que el cliente hace una petición al servidor, este la procesa, responde, y el cliente procesa la respuesta. Una situación de este estilo se puede modelar con un flujo e2e de 5 actividades que recorren 3 recursos procesadores en el siguiente orden:

Cliente – Red – Servidor – Red – Cliente

Si especificamos a GEN4MAST generar sistemas con 1 flujo e2e de 5 actividades localizadas en 3 recursos, creará sistemas que localizaran las actividades de forma aleatoria, que en la mayoría de los casos no seguirán la arquitectura que deseamos estudiar.

Para este tipo de situaciones, GEN4MAST proporciona otro método adicional y opcional de generación, en el que se pueden especificar de forma concreta algunas características de los sistemas. Este método se define dentro de la sección ARCHITECTURE del fichero de configuración, cuya sintaxis se muestra en la Figura 4-18, y permite determinar las siguientes características de los sistemas a generar:

- LENGTH: Lista con los valores de las longitudes (número de actividades) de cada flujo e2e del sistema. El número total de flujos e2e del sistema (*N* en la figura), se extrae de manera implícita como la longitud de esta lista.
- PROCESSORS: Lista con *N* sub-listas, una por cada flujo e2e. Cada sub-lista especifica el índice del recurso procesador en el que se localizan las actividades de cada flujo e2e.
- PERIODS: Lista con los periodos de cada uno de los flujo e2e.
- DEADLINES: Lista con los plazos de principio a fin de cada uno de los flujo e2e.

```
[ARCHITECTURE]
LENGTH [L1,L2,…,LN]
PROCESSORS [[P11,…,P1L1],…,[PN1,…,PNLN]]
PERIODS [T1,T2,…,TN]
DEADLINES [D1,D2,…,DN]
[/ARCHITECTURE]
```

Figura 4-18 Sección de especificación de una arquitectura de generación fija.

Los valores que se establecen en los parámetros de la sección ARCHITECTURE son *strings* que representan listas en formato Python. En la Figura 4-19 se muestra la especificación del caso de arquitectura cliente-servidor, en la que el flujo e2e posee un periodo de 100, un plazo de principio a fin de 500, y las actividades se ejecutan en los recursos procesadores en el orden: Cliente(1) – Red(2) – Servidor(3) – Red(2) – Cliente(1)

```
[ARCHITECTURE]
LENGTH [5]
PROCESSORS [[1,2,3,2,1]]
PERIODS [100]
DEADLINES [500]
[/ARCHITECTURE]
```

Figura 4-19 Ejemplo de definición de una arquitectura de generación fija.

Todos estos parámetros de la sección ARCHITECTURE son opcionales, y funcionan en combinación con los definidos en la sección GENERATOR, teniendo prioridad los especificados en ARCHITECTURE. Esto significa que si se especificara alguno en ARCHITECTURE, sus equivalentes en la sección GENERATOR serían ignorados. Los parámetros que no se especifiquen en ARCHITECTURE serán generados con los criterios de generación de la sección GENERATOR.

4.2.2 Generación de *scripts* de ejecución

Una vez completado el proceso de generación de los modelos de MAST de todos los sistemas, el siguiente paso es el de crear los *scripts* de ejecución de la herramienta de análisis y optimización de MAST. Una ejecución de esta herramienta consiste en la aplicación de una o varias de las técnicas disponibles en MAST sobre un sistema concreto, con el objetivo de obtener resultados correspondientes sobre su planificabilidad.

Los dos tipos de técnicas que se pueden ejecutar son los algoritmos de asignación de parámetros de planificación, y las técnicas de análisis de planificabilidad con las que calcular los tiempos de respuesta de peor caso de todas las actividades del sistema. Adicionalmente, se pueden especificar parámetros que modifican el comportamiento de estas técnicas, como por ejemplo límites en el número de iteraciones de análisis y/o asignación de parámetros de planificación, o los valores de los parámetros *k* del algoritmo de asignación HOSPA.

La configuración de los *scripts* de ejecución se declara en la sección EXECUTION del fichero de configuración de GEN4MAST. La sintaxis de esta sección, junto con sus parámetros de configuración se resumen en la Figura 4-20. En los parámetros que lo

```
[EXECUTION]
MAST_PATH path?description …
ANALYSIS_TOOL HOLISTIC/OFFSET/OFFSET_OPT/SLANTED/BRUTE_FORCE …
DEADLINES_TYPE LOCAL/GLOBAL …
ASSIGNMENT_TOOL UD/ED/PD/NPD/EQS/EQF/HOSPA …
STOP_FACTOR sf …
SLACK True/False
NO_JITTER True/False …
K ka,kr …
ITERATIONS_PER_PAIR iterations …
OPTIMIZATION_ITERATIONS optiter …
INITIALIZATION_TYPE PD/NPD …
[/EXECUTION]
```

Figura 4-20 Formato de la sección de configuración de la ejecución

soportan, si se especificaran varios valores separados por espacios, se generarán *scripts* de ejecución con todas las combinaciones de valores suministrados.

A continuación se describe el cometido de cada uno de estos parámetros:

- MAST_PATH: Localización del ejecutable de la herramienta de análisis y optimización de MAST (*mast_analysis*). Se podría querer suministrar diferentes ejecutables, por ejemplo para probar versiones personalizadas de MAST en las que se están experimentando nuevas técnicas. Opcionalmente, a cada localización se le puede asignar un descriptor con el que identificar la versión de MAST asociada dentro de la base de datos de resultados. Este descriptor se separa de la localización mediante el carácter "?".

- ANALYSIS_TOOL: Declara la técnica de análisis de planificabilidad que se aplicará. Si la técnica especificada no está disponible para la política de planificación especificada, MAST utilizaría la técnica holística, ya que es la única disponible tanto para FP como LC-EDF y GC-EDF. De acuerdo con las técnicas disponibles en MAST (ver Sección 2.5), los valores permitidos en este parámetro son:

  - ✓ "HOLISTIC": Técnica holística de análisis (*holistic analysis*).
  - ✓ "OFFSET": Técnica de análisis basada en o*ffsets* (*offset based analysis*).
  - ✓ "OFFSET_OPT": Técnica de análisis basada en o*ffsets*, con relaciones de precedencia (*offset based analysis with precedence relationships*).
  - ✓ "SLANTED": Técnica de análisis basada en o*ffsets* s*lanted* (*offset based slanted analysis*).
  - ✓ "BRUTE_FORCE": Técnica de análisis basada en *offsets* de fuerza bruta (*offset based brute force analysis*).

- STOP_FACTOR: Entero positivo que establece el factor de interrupción del análisis. Este es un mecanismo incorporado en MAST para prevenir los posibles problemas de convergencia en sistemas no planificables que se puedan presentar durante el análisis. Si los tiempos de respuesta de peor caso provisionales obtenidos en algún momento del proceso de análisis para alguna actividad τ_{ij} superan el valor STOP_FACTOR*D_{ij}, el análisis se detiene automáticamente, declarando el sistema como no planificable. (Parámetro Stop_Factor_When_Not_Schedulable en la Sección 3.1.1).

- DEADLINES_TYPE: Especifica cómo se deben interpretar los plazos de planificación en EDF en función del tipo de sincronización de relojes disponible. En las versiones del modelo MAST anteriores a la 2.0, la diferenciación del tipo de planificador EDF en sistemas distribuidos no está soportada en el modelo, y debe ser especificada en la ejecución de la herramienta MAST. Los valores soportados son:

  - ✓ "LOCAL": Plazos de planificación locales para planificación LC-EDF.
  - ✓ "GLOBAL": Plazos de planificación globales para planificación GC-EDF.

- ASSIGNMENT_TOOL: Establece el algoritmo de asignación de parámetros de planificación a aplicar. De acuerdo con las herramientas disponibles en MAST (ver Sección 2.5), los valores permitidos en este parámetro son:

- ✓ "UD": Asignación *Ultimate Deadline* (UD).
- ✓ "ED": Asignación *Effective Deadline* (ED).
- ✓ "PD": Asignación *Proportional Deadline* (PD).
- ✓ "NPD": Asignación *Normalized Proportional Deadline* (NPD).
- ✓ "EQS": Asignación *Equal Slack* (EQS).
- ✓ "EQF": Asignación *Equal Flexibility* (EQF).
- ✓ "HOSPA": Asignación *Heuristic Optimized Scheduling Parameter Assignment* (HOSPA).

- SLACK: Es un booleano que con valor *True* indica que se calculará el *slack* del sistema, de acuerdo con la definición dada en la Sección 1.4.

Adicionalmente a estos parámetros de aplicación general, GEN4MAST también soporta la configuración de la ejecución del algoritmo de asignación HOSPA mediante los siguientes parámetros:

- K: Parejas de valores separados por coma, que especifican los parámetros k_a y k_r del algoritmo HOSPA.

- ITERATIONS_PER_PAIR: Entero que establece el número de iteraciones a llevar a cabo con cada pareja k_a,k_r.

- OPTIMIZATION_ITERATIONS: Entero que especifica el número de sobre-iteraciones de optimización a llevar a cabo sobre una solución ya planificable, si se encontrara.

- INITIALIZATION_TYPE: *String* que especifica el tipo de asignación inicial de parámetros de planificación que utilizará HOSPA. Los tipos de inicialización disponibles para este parámetro son "PD" y "NPD".

La ejecución de cada una de las instancias de la herramienta MAST se llevará a cabo a través de su interfaz de línea de comandos. Una invocación de MAST por línea de comandos define la aplicación de una o varias técnicas sobre un sistema concreto, y como respuesta se obtiene un fichero con los resultados sobre la planificabilidad. En la Figura 4-21 se muestra el formato de una de estas invocaciones, para la versión 1.4 de MAST. Se puede apreciar que sobre el sistema definido en el fichero "*sistema_mast.txt*" se aplica la técnica de asignación de parámetros de planificación HOSPA, utilizando los parámetros de HOSPA (por ejemplo, los parámetros *k*) definidos en el fichero "*param.txt*", y utilizando además el análisis holístico como técnica de análisis de planificabilidad. En este ejemplo los resultados se almacenan en el fichero "*resultados.txt*". La sintaxis completa de la herramienta de análisis y optimización de MAST (*mast_analysis*) se puede encontrar en su manual de usuario accesible en [MASTh].

```
mast_analysis.exe holistic -p -t hospa sistema_mast.txt resultados.txt -a param.txt
```

Figura 4-21 Comando de ejecución de la herramienta de análisis de MAST

Cada uno de los *scripts* de ejecución que genera GEN4MAST agrupa las ejecuciones de una combinación concreta de parámetros de ejecución sobre una Serie de Utilizaciones generadas sobre un Sistema Raíz concreto. Al unificar todas las ejecuciones de una Serie de Utilizaciones dentro de un mismo *script* nos aseguramos de que en el caso de que se

produzca algún fallo en alguna ejecución de MAST, no aparezcan puntos intermedios en las Series de Utilizaciones sin resultados. Este proceso se visualiza en la Figura 4-22, en la que se aplica el análisis holístico y la asignación HOSPA sobre una serie de utilizaciones que comienza en el 1% para un sistema denominado "Sis".

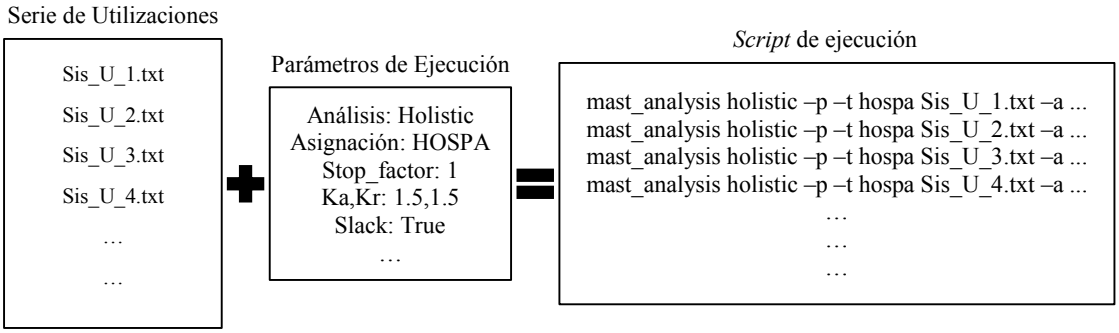


Figura 4-22 Agrupación de invocaciones de MAST en *scripts* de ejecución

4.2.3 Estructura del entorno de ejecución

Como se ha ido describiendo, durante la fase de generación se producen dos tipos de ficheros: ficheros con las descripciones de los sistemas de acuerdo al modelo MAST, y ficheros *script* en los que se especifican las ejecuciones concretas de la herramienta MAST. Adicionalmente, se crean otros ficheros auxiliares y la estructura de directorios que servirán para facilitar la posterior ejecución de la herramienta de procesado de resultados de GEN4MAST. Todo el conjunto de ficheros y la estructura de directorios propia de una generación de la herramienta GEN4MAST se esquematiza en la Figura 4-23, al que denominamos el entorno de ejecución de GEN4MAST.

```
raíz_estudio
    |--config.txt
    |--ID
        |--Nombre_Serie_Sistemas
            |--log_series.txt
            |--info_gen.csv
            |--example_names.txt
            |--Sistema_Utilización_Inicial.txt
            |--(. . .)
            |--Sistema_Utilización_Final.txt
            |--Tecnicas_Aplicadas
                |--commands.txt
                |--info_bench.csv
                |--scripts
                |   |--
                |--results
                |   |--
                |--out
                |   |--
                |--done
                    |--
```

Figura 4-23 Estructura de ficheros y directorios que forma el entorno de ejecución de GEN4MAST

El directorio raíz del entorno de ejecución puede ser cualquier directorio que contenga el fichero de configuración de GEN4MAST, llamado "config.txt" en el ejemplo de la Figura 4-23. Dentro del directorio raíz se crea un directorio para cada repetición del proceso de generación establecido con el parámetro POPULATION. Este directorio se identifica mediante el identificador (ID en la figura) de cada repetición (ver Sección 4.2.1.7).

Dentro de cada directorio ID se almacenan una serie de directorios, uno por cada serie de utilizaciones generadas a partir de un Sistema Raíz. El nombre de cada uno de estos directorios de almacenamiento de las series ("Nombre_Serie_Sistemas" en la figura) identifica a cada serie de forma única, y se genera a partir de los valores de los parámetros del Sistema Raíz sobre el que se construyen.

En cada uno de estos directorios se almacenan los ficheros descriptores de los sistemas que componen la Serie de Utilizaciones. Adicionalmente esta carpeta contiene ficheros auxiliares como "log_series.txt" en el que se registran los eventos sucedidos durante el proceso de generación, "info_gen.csv" en el que se almacenan los valores de los parámetros de generación de la serie, y "example_names.txt" que contiene los nombres de los ficheros descriptores de los sistemas generados.

Dentro del directorio de cada serie se crea un directorio por cada combinación de parámetros de ejecución especificada, en el que se almacenan los ficheros relativos a la ejecución de la herramienta MAST. Estos ficheros son un *script* de ejecución de la herramienta MAST ("commands.txt" en el ejemplo), e "info_bench.csv", en el que se almacenan los parámetros de ejecución del directorio en el que reside. El nombre de este directorio ("Tecnicas Aplicadas" en el ejemplo), se genera a partir de los parámetros de generación de los *scripts*. En los directorios *scripts*, *results*, *out* y *done* se almacenarán los ficheros que se generarán durante la fase de ejecución de GEN4MAST.

4.2.4 Ejemplo de generación

Una vez descrito el proceso completo de generación de GEN4MAST, ahora mostramos un ejemplo de uso en el que se aprecie cómo se pueden utilizar los parámetros de generación que hemos descrito para conseguir llevar a cabo un estudio concreto.

Suponemos que queremos llevar a cabo un estudio que compare los resultados obtenidos por las técnicas de análisis holística y basada en *offsets*, observando las diferencias entre ellas ante diferentes circunstancias:

- ✓ Sistemas en los que todos sus flujos e2e tengan 3, 5, 7 y 9 actividades.
- ✓ Parámetros de planificación asignados por PD y HOSPA.
- ✓ Planificación FP y GC-EDF.
- ✓ Actividades con la misma utilización ("SCALE-WCET"), o con distribución uniforme ("UUNIFAST").

Queremos estudiar el comportamiento de ambas técnicas de análisis en el rango de utilizaciones del 10% al 80% en pasos del 1%. Todos los sistemas tendrán 4 flujos e2e, con periodos fijos con valores 100, 500, 1000 y 2000 para cada flujo e2e. Se repetirá el proceso en 10 ocasiones para intentar obtener resultados estadísticamente relevantes. La ejecución de HOSPA se llevará a cabo con sus parámetros por defecto. En la Figura 4-24 se muestra cómo debería especificarse el fichero de configuración de GEN4MAST para

generar el entorno de estudio necesario para llevar a cabo la comparación que nos proponemos.

```
[GENERATOR]
POPULATION 10
N_FLOWS 4
N_STEPS 3 5 7 9
SINGLE_FLOWS 0
FIXED_LENGTH True
N_PROCESSORS 3
REPETITION YES
SCHEDULING_POLICY FP EDF
DEADLINES NT
UTILIZATION_START 1
UTILIZATION_STEP 1
UTILIZATION_STOP 80
UNIFORM_UTILIZATION True
WORKLOAD UUNIFAST SCALE
BEST_CASE 0
[/GENERATOR]

[ARCHITECTURE]
PERIODS [100,500,1000,2000]
[/ARCHITECTURE]

[EXECUTION]
MAST_PATH /mast/mast_analysis
ANALYSIS_TOOL HOLISTIC OFFSET_BASED
ASSIGNMENT_TOOL PD HOSPA
[/EXECUTION]
```

Figura 4-24 Fichero de configuración para el ejemplo de utilización de GEN4MAST

Durante la fase de generación se crean todos los ficheros necesarios para ejecutar el estudio. El siguiente paso es llevar a cabo todas las ejecuciones de la herramienta MAST especificadas. Este paso constituye la Fase de Ejecución, y se describe en la siguiente sección.

4.3 Fase de ejecución

En la Fase de Ejecución, el módulo Despachador de Trabajos de GEN4MAST se encarga de ejecutar todos los *scripts* de la herramienta MAST almacenados en el entorno de ejecución, de forma que todos los resultados relevantes queden registrados.

GEN4MAST soporta dos modos para ejecutar el estudio: de forma local sobre un PC convencional, o de forma distribuida en un supercomputador. A continuación describimos ambos métodos.

4.3.1 Ejecución de forma local

El mecanismo de ejecución local es una manera trivial de ejecución en la que todos los *scripts* del estudio se ejecutan de forma secuencial en un PC local. Las principales ventajas son la sencillez y la accesibilidad, ya que sólo se requiere un PC que disponga de una instalación de Python, y un ejecutable de la herramienta de análisis y optimización de MAST (*mast_analysis*). En la página de MAST [MASTh] se suministran ejecutables para Windows y Linux de esta herramienta. Además, se suministra el código fuente para ser compilado en otros sistemas operativos.

Sin embargo, la ejecución local sólo está indicada para estudios relativamente sencillos en los que el tiempo total de CPU requerido no sea elevado. Es habitual encontrarnos con sistemas cuyo análisis requiera varias horas, o incluso días, o también podemos tener estudios en los que cada análisis dura poco tiempo, pero se requiere un número muy elevado de ellos. En cualquiera de los dos casos, podemos estar ante requisitos de uso de CPU del orden de las semanas o los meses, por lo que el uso de un PC convencional en la ejecución de estos estudios es inadecuado. El uso de un PC con un procesador de múltiples núcleos puede acelerar la ejecución al permitir que varios *scripts* se puedan ejecutar en paralelo, pero en cualquier caso para los estudios que se pretenden realizar en esta tesis se hace necesaria una solución con mayor capacidad de cómputo.

4.3.2 Ejecución en un supercomputador

A medida que ampliamos la complejidad de los estudios que llevamos a cabo con GEN4MAST, el número de ejecuciones y el tiempo requerido para llevarlas a cabo en una única CPU crece exponencialmente. Las ejecuciones de MAST que realizamos en el estudio son completamente independientes entre sí, esto es, los resultados de una ejecución no dependen de los resultados de otra. Esta propiedad permite que este proceso sea fácilmente adaptable a un sistema de cómputo paralelo, en el que se pueden obtener grandes reducciones en los tiempos totales de ejecución. Por todo ello, añadimos a GEN4MAST la capacidad de distribuir su trabajo en un supercomputador, abriéndonos así la puerta a poder llevar a cabo estudios más complejos o extensos.

Resulta cada vez más habitual encontrarnos en universidades o centros de investigación con supercomputadores que están a disposición de sus investigadores para el desarrollo de sus tareas. Existen incluso iniciativas como la Red Española de Supercomputación [RES] en la que uno de sus objetivos es el de dar acceso a estos recursos al mayor número de investigadores en España.

En el Departamento de Ingeniería Informática y Electrónica de la Universidad de Cantabria tenemos disponible el supercomputador Tres Mares [TRESM], compuesto en la actualidad por 570 procesadores del tipo AMD Opteron e Intel Xeon. Cada supercomputador puede poseer distintas particularidades que condicionan su forma de uso. El soporte para supercomputadores de GEN4MAST está pensado para ser utilizado en Tres Mares.

Tres Mares utiliza el sistema de encolado de trabajos TORQUE [TORQ], que es un sistema de gestión de trabajos ampliamente utilizado en los sistemas de supercomputación. Para ejecutar GEN4MAST en otro supercomputador que también utilice TORQUE sólo se requerirían mínimas modificaciones, o posiblemente ninguna.

El funcionamiento básico de un supercomputador como Tres Mares se basa en un sistema de colas. Si se desea ejecutar un trabajo en el supercomputador, éste se envía al sistema de colas, que a continuación decidirá cuándo será ejecutado, y en qué nodo procesador. Esta decisión se toma basándose en factores tales como la disponibilidad de algún nodo libre, la prioridad que se le ha asignado al trabajo, o los privilegios del propio usuario. Un resumen del proceso se representa en la Figura 4-25.

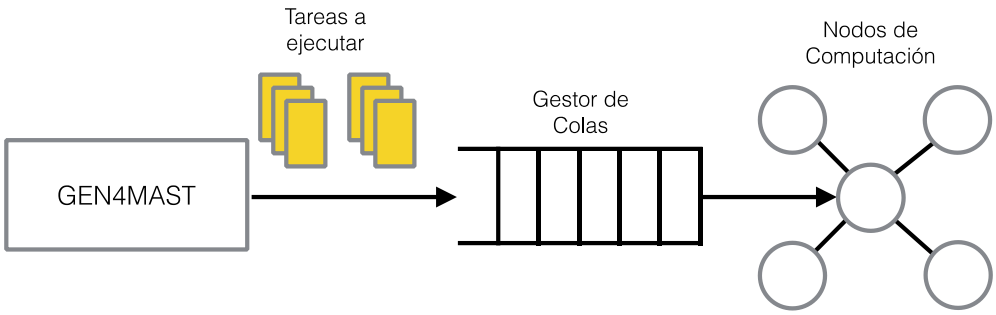


Figura 4-25 Esquema de ejecución en el supercomputador

Los sistemas de supercomputación son un medio que típicamente es compartido por varios usuarios al mismo tiempo. Por esta razón, el módulo Despachador de Trabajos de GEN4MAST, y en general cualquier usuario del supercomputador, debe tomar ciertas medidas para evitar sobrecargar las colas y nodos, evitando la degradación del rendimiento al resto de usuarios. Algunas de estas medidas son configurables por el usuario mediante la sección CLUSTER del fichero de configuración de GEN4MAST. Esta sección queda delimitada por los *strings* [CLUSTER] y [/CLUSTER], cuyos parámetros de configuración se resumen en la Figura 4-26.

```
[CLUSTER]
USER_NAME user
PACK_SIZE number
JOB_NAME name
TIMEOUT time
GROUP groupname
BATCH_SIZE b_size
BATCH_SLEEP b_sleep
[/CLUSTER]
```

Figura 4-26 Formato de la sección de configuración del envío de tareas al supercomputador

Las medidas que toma el Despachador de Trabajos de GEN4MAST son:

- Los trabajos que se envían al supercomputador tienen asignada una prioridad. El usuario de GEN4MAST puede seleccionar la prioridad de sus trabajos mediante el parámetro GROUP. Los valores que admite este parámetro son propios de Tres Mares, y son "*short*", "*medium*" y "*long*" para prioridad alta, media y baja respectivamente. Es responsabilidad del usuario elegir la prioridad adecuada, observando el estado del supercomputador antes de proceder al envío de los trabajos. Por defecto se elige una prioridad baja.

- Los trabajos que se envían al sistema de colas del supercomputador pueden estar formados por uno o varios de los *scripts* de ejecución que se crean en la fase de generación. De esta forma el número total de trabajos a enviar se puede reducir. El

parámetro PACK_SIZE indica cuantos *scripts* de ejecución de la herramienta de análisis y optimización de MAST se empaquetan en cada trabajo enviado al supercomputador.

- Se puede especificar un tiempo límite de ejecución para los trabajos. Cuando dicho tiempo se supera, el trabajo se cancela, liberando así recursos para otros trabajos o usuarios. Este tiempo se puede especificar con el parámetro TIMEOUT. Limitar el tiempo de ejecución de los trabajos es útil para cancelar las ejecuciones que presenten problemas de convergencia.

- Aunque las colas en el supercomputador tienen gran capacidad, cuando una cola tiene un número alto de trabajos se pueden producir problemas de rendimiento que afectarían a todos los usuarios del sistema. Para evitar esta circunstancia, en GEN4MAST se puede establecer un número máximo de trabajos que puede tener el usuario en el sistema en todo momento, independientemente de si están ejecutando o esperando en la cola. Si se alcanzara dicho límite, se esperaría una cantidad de tiempo predeterminada para que fueran finalizando algunos trabajos. El número total de trabajos, y el tiempo de espera, se configuran con los parámetros BATCH_SIZE y BATCH_SLEEP respectivamente. El algoritmo con el que se envían los trabajos se describe con el pseudocódigo de la Figura 4-27. Para que GEN4MAST pueda obtener correctamente el número de trabajos en el supercomputador, debe conocer el nombre del usuario (*login name*). Este nombre se suministra a través del parámetro USER_NAME.

```
loop
   if TRABAJOS_DEL_USUARIO_EN_CLUSTER < BATCH_SIZE
      enviar siguiente trabajo
      exit when no hay más trabajos
   else
      wait (BATCH_SLEEP)
end loop
```

Figura 4-27 Algoritmo de envío de tareas al *cluster*

Adicionalmente, a los trabajos enviados se les puede asignar un nombre dentro del supercomputador, de forma que se les pueda distinguir de otros trabajos pertenecientes a otros estudios del mismo usuario. El nombre de los trabajos se establece con el parámetro JOB_NAME. Todos los trabajos del mismo estudio tendrán como nombre el establecido por JOB_NAME, seguido de un número que aumentará secuencialmente a medida que se envíen trabajos.

La ejecución en el supercomputador requiere la creación de ficheros temporales en los que, siguiendo un formato establecido por Tres Mares, se define el envío de los trabajos a las colas. Tales ficheros son creados automáticamente por GEN4MAST, de forma que el envío de trabajos se produce de manera transparente para el usuario. Estos ficheros auxiliares residen en el directorio "*scripts*" de cada directorio de ejecución (véase Figura 4-23), y se borran automáticamente una vez que el envío de trabajos ha finalizado.

4.4 Fase de resultados

Independientemente del método de ejecución utilizado, local o en un supercomputador, cada ejecución de MAST genera un fichero con los resultados obtenidos. Estos ficheros son de texto plano y se puede acceder a ellos utilizando cualquier editor de texto. Los resultados que se almacenan para cada ejemplo contienen información con:

- Tiempos de respuesta de peor caso de todos los flujos e2e. Si se ha llevado a cabo la asignación de parámetros de planificación iterativo HOSPA, también se guardan los tiempos de respuesta de peor caso de todos los flujos e2e obtenidos en cada una de las iteraciones del algoritmo.
- Plazos de principio a fin de todos los flujos e2e. Se incluyen para tener un fácil acceso a estos valores, y poder así determinar la planificabilidad del sistema sin necesidad de acceder al fichero de descripción original.
- *Slack* del sistema, si éste fue calculado.
- Tiempo de ejecución requerido para el cálculo de los tiempos de respuesta de peor caso.
- Tiempo de ejecución requerido para la asignación de parámetros de planificación (si es el caso).
- Tiempo de ejecución requerido para el cálculo del *slack* del sistema (si es el caso).

En cualquier estudio que realicemos se generarán potencialmente miles de ficheros de resultados. Acceder a ellos de forma individual, aunque se automatizara el proceso, resultaría una tarea ardua e ineficiente. Para facilitar el acceso a los resultados, especialmente en los estudios más extensos, GEN4MAST proporciona un módulo (Procesador de Resultados) que compila todos los resultados en una base de datos indexada de fácil acceso.

La base de datos construida por el módulo Procesador de Resultados de GEN4MAST utiliza el formato PyTables [PYTAB], que es un paquete Python para manejar y almacenar grandes cantidades de datos numéricos de una manera eficiente. GEN4MAST automatiza la creación y acceso a esta base de datos, de forma que el usuario no necesita conocer las particularidades del uso de PyTables. Las bases de datos Pytables siguen el formato tradicional de tabla, en el que cada nueva entrada es una fila, y los datos forman parte de sus columnas.

A la hora de realizar esta tesis y estudiando otros trabajos en la literatura, identificamos dos tipos distintos de acceso a los resultados que se llevan a cabo típicamente: estudio de la utilización máxima planificable, y estudio de la evolución de alguna métrica de rendimiento a medida que se aumenta la utilización de los sistemas. La estructura de la base de datos que genera GEN4MAST se crea teniendo en cuenta estos casos de uso típicos.

La base de datos que genera GEN4MAST posee una tabla principal (*main table*). Cada entrada (fila) en esta tabla reúne la información relativa a los resultados obtenidos para una combinación de parámetros de ejecución (en la sección EXECUTION del fichero de configuración) sobre una Serie de Utilizaciones. Las columnas de cada entrada de la tabla principal se dividen en dos tipos: columnas descriptivas, y columnas de resultados. En la Figura 4-28(a) se puede observar una lista con los nombres de las columnas de la tabla principal y sus tipos asociados. Los tipos UInt8Col y Uint16Col se

corresponden con un entero sin signo de 8 y 16 bits respectivamente, Float32Col es un número en coma flotante de 32 bits, BoolCol es un booleano, y StringCol es un *string*.

Las columnas descriptivas incluyen los parámetros que se utilizaron para generar la serie de resultados, como por ejemplo el número de flujos e2e de los sistemas que lo componen, o el análisis de planificabilidad aplicado. Estas columnas son un mapeo directo de los parámetros de generación y ejecución suministrados al módulo Generador de GEN4MAST, y se utilizan para distinguir las series y así poder filtrar los resultados.

Las dos últimas columnas en la tabla principal (dos últimos elementos de la lista en la Figura 4-28(a)), se corresponden con las columnas de resultados: MAX_SCHEDULABLE y DETAILS. En MAX_SCHEDULABLE se almacena la utilización máxima planificable obtenida en la serie de utilizaciones asociada, mientras que la columna DETAILS almacena un *string* que actúa como puntero a otra tabla en la que en cada una de sus entradas (filas) se almacenan los resultados de cada uno de las utilizaciones dentro de la Serie de Utilizaciones. A estas nuevas tablas las llamamos tablas de detalles (*details tables*). En la Figura 4-28(b) se muestran las columnas de las tablas de detalles, cuyos contenidos son:

Tabla Principal
(*Main Table*)

ID	UInt8Col
N_FLOWS	UInt8Col
N_STEPS	UInt8Col
SINGLE_FLOWS	UInt8Col
FIXED_LENGTH	BoolCol
N_PROCESSORS	UInt8Col
SCHEDULING_POLICY	StringCol
PERIOD_BASE	UInt16Col
PERIOD_RATIOS	UInt16Col
UTILIZATION_START	UInt8Col
UTILIZATION_STEP	UInt8Col
UTILIZATION_STOP	UInt8Col
DEADLINES	StringCol
UNIFORM_UTILIZATION	BoolCol
WORKLOAD	StringCol
PERIOD_DISTRIBUTION	StringCol
BEST_CASE	UInt8Col
OBSERVATIONS	StringCol
ANALYSIS_TOOL	StringCol
DEADLINES_TYPE	StringCol
ASSIGNMENT_TOOL	StringCol
STOP_FACTOR	Float32Col
SLACK	BoolCol
Ka	Float32Col
Kr	Float32Col
ITERATIONS_PER_PAIR	UInt8Col
OPTIMIZATION_ITERATIONS	UInt8Col
INITIALIZATION_TYPE	StringCol
MAX_SCHEDULABLE	UInt8Col
DETAILS	StringCol

(a)

Tabla Detalles
(*Details Table*)

u	UInt8Col
min_r	Float32Col
max_r	Float32Col
avg_r	Float32Col
as_t	Float32Col
an_t	Float32Col
s_t	Float32Col
slack	Float32Col
sched	BoolCol
n_iter	UInt8Col
index	Float32Col

(b)

Figura 4-28 Estructura de la base de datos de GEN4MAST, para (a) la tabla principal, y (b) la tabla de detalles.

- u: Utilización del sistema.
- min_r: Tiempo de respuesta de peor caso mínimo entre todos los flujos e2e.
- max_r: Tiempo de respuesta de peor caso máximo entre todos los flujos e2e.
- avg_r: Tiempo de respuesta de peor caso medio entre todos los flujos e2e.
- as_t: Tiempo de ejecución requerido para el cálculo de los parámetros de planificación.
- an_t: Tiempo de ejecución requerido para el cálculo final de los tiempos de respuesta de peor caso.
- s_t: Tiempo de ejecución requerido para el cálculo del *slack* del sistema.
- *slack*: *Slack* del sistema (en %).
- sched: Booleano que indica si el sistema es planificable.
- n_iter: Número de iteraciones que necesitó el algoritmo HOSPA para encontrar una asignación que consiguiera hacer al sistema planificable. Si no se consiguió la planificabilidad, este asigna el valor especial 0.

Se puede observar que en la tabla de detalles, en vez de almacenar los tiempos de respuesta de peor caso de cada flujo e2e, se almacenan tres valores estadísticos de éstos: los valores mínimo, medio y máximo. Almacenar los tiempos de respuesta de peor caso de todos los flujos conlleva la definición de bases de datos con estructuras innecesariamente complejas, ya que el número de flujos e2e es variable.

4.4.1 Acceso a los resultados

El formato Pytables de la base de datos de GEN4MAST posee una API bien documentada que puede ser utilizada para acceder a los resultados. Con el objetivo de facilitar este acceso a los resultados, GEN4MAST suministra una librería Python con la que, con conocimientos básicos de Pytables, se puede filtrar la información contenida en la base de datos, y presentar los resultados con gráficas o tablas.

El proceso de creación de gráficas y tablas con GEN4MAST se basa en el uso de filtros que seleccionan las entradas de la base de datos sobre las que se quieren observar resultados. Los filtros de GEN4MAST son diccionarios Python en los que las llaves (*keys*) son las columnas de descripción de la tabla principal, y sus valores asociados son las características que poseen los sistemas sobre los que se desean obtener los resultados. Por ejemplo, si queremos seleccionar los resultados de los sistemas con 5 procesadores, y analizados con la técnica holística, el filtro que deberíamos utilizar seria el siguiente:

```
dict(N_PROCESSORS=5, ANALYSIS_TOOL=”HOLISTIC”)
```

Una descripción detallada de los métodos de generación de gráficas y tablas, junto con un tutorial, puede encontrarse en el manual de usuario de GEN4MAST [MASTh].

4.5 Comportamiento de los métodos de generación

Uno de los objetivos de GEN4MAST es el de generar sistemas sintéticos que representen situaciones reales lo más fehacientemente posible. Para conseguir este objetivo, hemos mostrado que GEN4MAST incorpora diferentes métodos de generación de las cargas.

Los periodos de los flujos e2e (T_i) y los tiempos de ejecución de peor caso (C_{ij}) de las actividades son dos parámetros que tienen una gran influencia en la planificabilidad de los sistemas [BIN05]. En esta sección vamos a llevar a cabo una evaluación de la influencia en sistemas distribuidos de los diferentes métodos de generación de periodos y tiempos de ejecución de peor caso incluidos en GEN4MAST. Esta evaluación servirá también para mostrar un caso de uso típico de la herramienta.

Utilizamos GEN4MAST para generar un conjunto de sistemas con las características de la Tabla 4-2. Para generar los periodos utilizamos tanto *Custom-Uniform* como *Log-Uniform*. Para los tiempos de ejecución de peor caso utilizamos *SCALE-WCET* y *UUnifast*. Para cada combinación de características generamos 300 Series de Utilizaciones (POPULATION 300) en el rango [10,99] (en %). En total se generaron 324000 sistemas.

Nº de flujos e2e	**Nº actividades por flujo e2e**	**Nº procesadores**	**Ratio periodos**	**Plazos D_i**	**Política de planificación**
10	10	5	100	$10*T_i$ ($2*T_i$ para GC-EDF)	FP LC-EDF GC-EDF

Tabla 4-2 Características de los sistemas del estudio comparativo de métodos de generación

Analizamos la planificabilidad utilizando la técnica holística en todos los casos, puesto que es la única disponible para las tres políticas de planificación bajo estudio. Observamos la evolución de la utilización máxima planificable (UMP) media en dos direcciones distintas: en función del número de series de utilizaciones promediadas, y en función de los métodos de generación utilizados. La ejecución del estudio se llevó a cabo en 300 procesadores del supercomputador Tres Mares, y requirió 30 minutos. Si se hubiera ejecutado en un PC convencional se hubiera necesitado casi una semana.

Antes de mostrar los resultados es importante hacer notar que en los casos planificados por GC-EDF, los plazos de principio a fin se establecieron como $D_i=2*T_i$ en vez de $D_i=10*T_i$. Esto es así debido a que en estudios anteriores [RIV09][RIV12], se observó cómo la planificación GC-EDF alcanza prácticamente el 100% de utilización máxima planificable cuando los plazos de principio a fin tienen valores $D_i=N_i*T_i$, prácticamente con independencia de cualquier otro parámetro del sistema. Con utilizaciones planificables tan elevadas resultaría difícil poder observar las diferencias producidas por los diferentes métodos de generación, o cualquier otro parámetro de generación. Para ayudar a la observación de este estudio, se eligen plazos de principio a fin menores, en este caso $D_i=2*T_i$, que disminuyen las utilizaciones máximas planificables, alejándolas del 100%, que sería un valor de saturación en este caso.

Mostramos los resultados obtenidos para cada política de planificación por separado.

4.5.1 Sistemas distribuidos con prioridades fijas (FP)

Un factor importante a estudiar de los métodos de generación es el de obtener su comportamiento estadístico. El objetivo es determinar el número de sistemas con las mismas características que se deben generar para poder obtener resultados relevantes. Este es un factor importante, ya que permite establecer un límite en el número de sistemas a generar, y por lo tanto limitar el tiempo de ejecución de los estudios, con la confianza de estar obteniendo resultados que estén lo suficientemente próximos a los valores finales a los que convergen.

En la Figura 4-29 se representan las utilizaciones máximas planificables (UMP) medias para cada combinación de métodos de generación, en función del número de series promediadas del conjunto total de las 300 series generadas. Consideramos que con 300 series, el valor obtenido para la media es el final, y se muestra en la etiqueta asociada a cada línea. En la figura puede observarse que para cualquier combinación de métodos, la media comienza a estabilizarse a partir de aproximadamente 20 series promediadas.

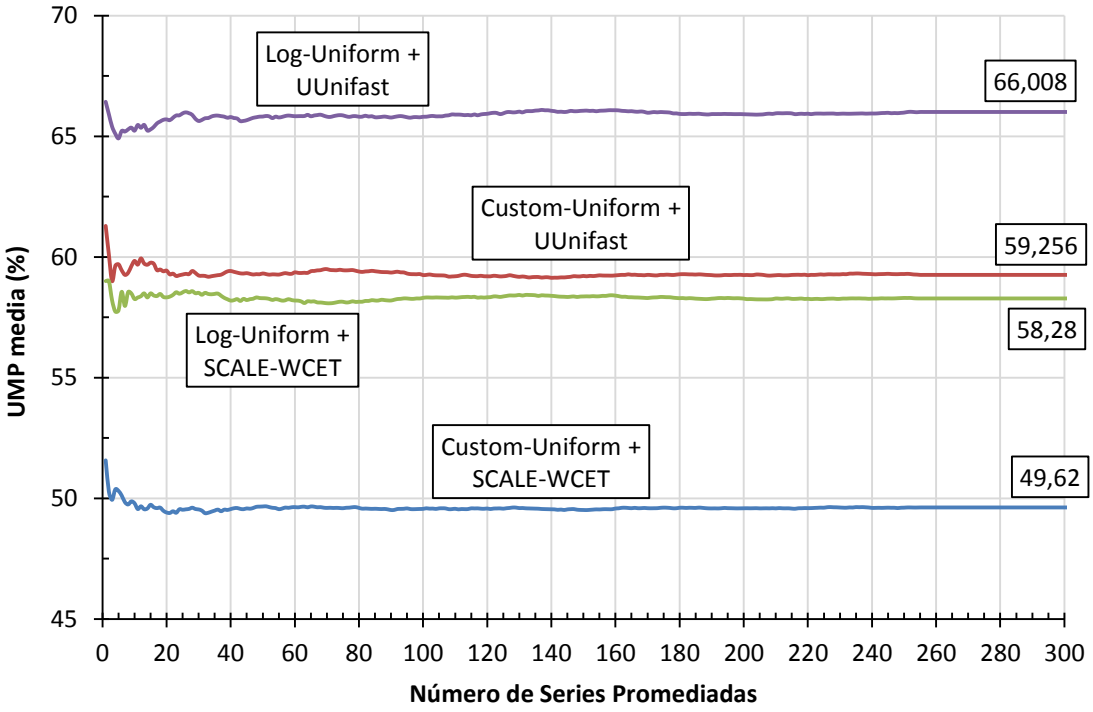


Figura 4-29 Convergencia de las medias de las utilizaciones máximas planificables en función del número de sistemas promediados, para las cuatro combinaciones de métodos de generación, y sistemas FP

Para observar con mayor detalle la evolución de la convergencia de la utilización máxima planificable, en la Figura 4-30 se representan las evoluciones de las desviaciones obtenidas con cada una de las cuatro combinaciones de métodos de generación, en función del número de series utilizadas en el promedio. Definimos la desviación como el valor absoluto de la diferencia entre el valor de la media obtenida utilizando un número determinado de series promediadas, y el valor final de convergencia, que se asume es el obtenido con la totalidad de 300 series. Añadimos una línea horizontal de referencia en 0,5%, valor de desviación que consideramos suficientemente bajo para llevar a cabo estudios con GEN4MAST con resultados relevantes. En la figura puede observarse cómo, a partir de 17 series, la desviación en el promedio de la utilización máxima planificable

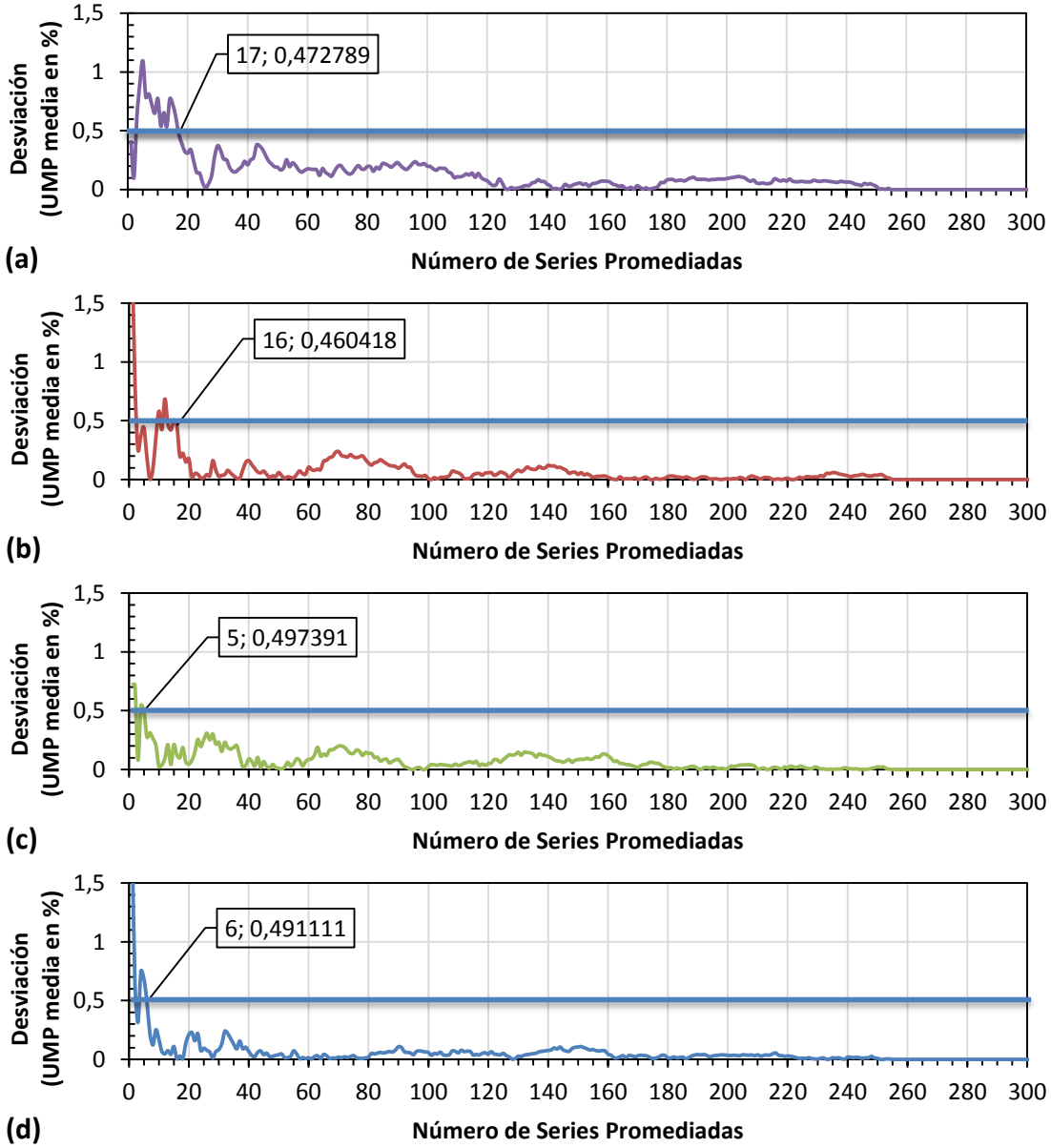


Figura 4-30 Desviación (distancia al valor de convergencia) en función del número de series promediadas, para sistemas FP y los métodos **(a)** ***Log-Uniform + UUnifast*****, (b)** ***Custom-Uniform + Uunifast*****, (c)** ***Log-Uniform + SCALE-WCET*****, (d)** ***Custom-Uniform + SCALE-WCET***

queda acotada a un valor de 0,5% para cualquier método utilizado. Esto nos indica que a la hora de llevar a cabo un estudio masivo con sistemas FP, podemos limitar a 17 el número de series a promediar, con cierta confianza de que los resultados ya son estadísticamente relevantes. En la práctica totalidad de aplicaciones de GEN4MAST, tal desviación resulta ya suficiente, por lo que resultaría innecesario aumentar por encima de 17 el número de Series de Utilizaciones a promediar.

Además de cómo se alcanza la convergencia a un valor medio, en la Figura 4-29 también pueden observarse las diferencias en cuanto a la utilización máxima planificable que se obtienen para cada uno de los métodos de generación probados. En la Figura 4-31 se reúnen estos valores de convergencia en una única figura, representando las utilizaciones máximas planificables (UMP) medias de las cuatro combinaciones de

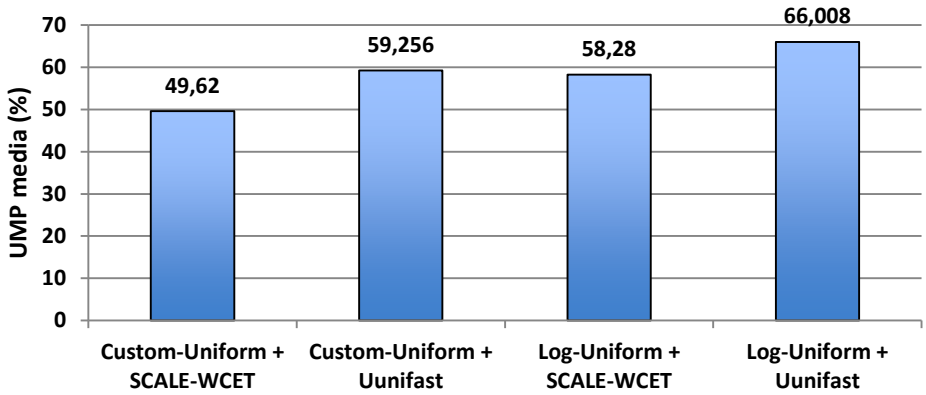


Figura 4-31 Media de las utilizaciones máximas planificables (UMP) para las cuatro combinaciones de métodos de generación, y sistemas FP

parámetros de generación. En esta figura se comprueba en primer lugar cómo, para el mismo método de generación de periodos (*Custom-Uniform* o *Log-Uniform*), el sesgo del algoritmo *SCALE-WCET*, en el que todas las actividades poseen la misma utilización, se manifiesta penalizando la planificación FP en mayor medida que utilizando *UUnifast*, que asigna utilizaciones a las actividades de forma uniforme. Esta conclusión es equivalente a la que se obtiene en [BIN05] para sistemas monoprocesadores.

Adicionalmente, en la Figura 4-31 se puede comprobar cómo la mayor dispersión de los periodos que produce el algoritmo *Log-Uniform* sobre *Custom-Uniform* da lugar a un aumento en la planificabilidad de los sistemas, comportamiento similar a lo que Lehoczky obtuvo en [LEH89] para sistemas monoprocesadores y planificación FP.

Por último, se puede concluir que para llevar a cabo estudios con GEN4MAST en los que se muestren con claridad los efectos de la dispersión de los valores de los periodos, es recomendable hacer uso del algoritmo de generación *Log-Uniform*. El algoritmo *Custom-Uniform* resulta más apropiado para modelar sistemas en los que los periodos se acumulen en torno al valor máximo.

4.5.2 Sistemas distribuidos EDF con relojes locales (LC-EDF)

Repetimos el mismo proceso realizado anteriormente para FP, pero esta vez con sistemas distribuidos EDF en los que no existe la sincronización entre los relojes (LC-EDF). Partimos de las mismas 300 series de utilizaciones con sistemas con las características de la Tabla 4-2, y aplicamos las cuatro combinaciones disponibles de métodos de generación. En la Figura 4-32 se representan, para las cuatro combinaciones de métodos de generación, los valores de las utilizaciones máximas planificables medias, en función del número de series promediadas. En la figura puede observarse cómo, al igual que ocurría con sistemas FP, con 300 series promediadas ya podemos asumir que el valor obtenido es el final. Estos valores finales se muestran en las etiquetas asociadas a cada línea.

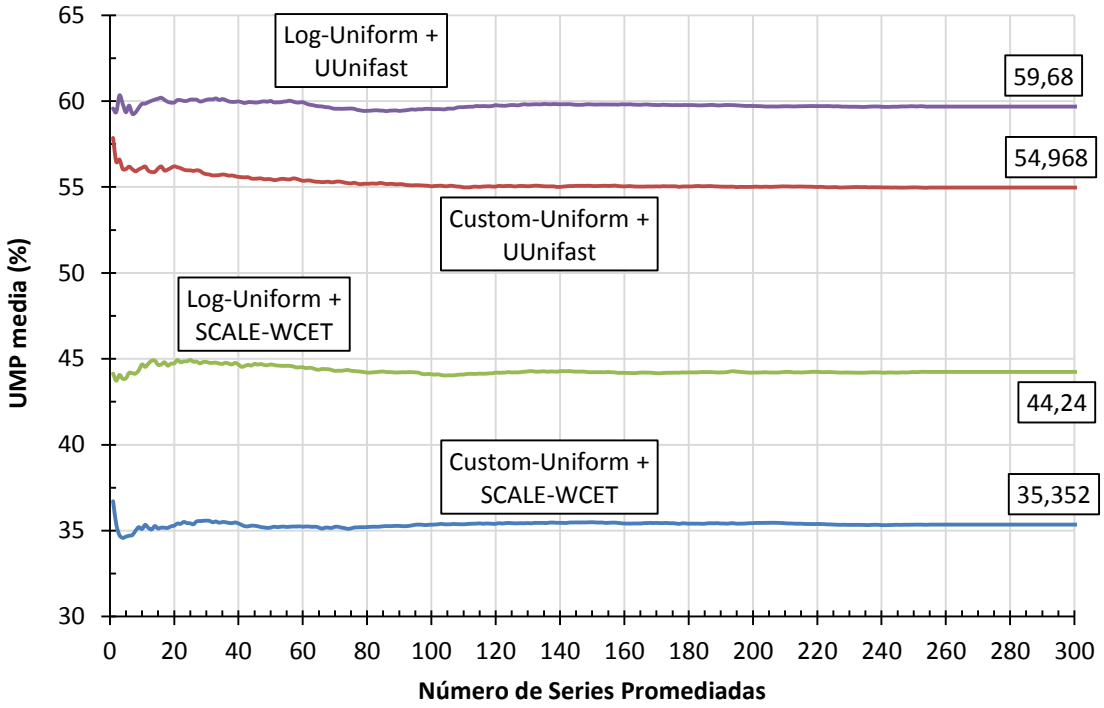


Figura 4-32 Convergencia de las medias de las utilizaciones máximas planificables en función del número de sistemas promediados, para las cuatro combinaciones de métodos de generación, y sistemas LC-EDF

En la Figura 4-32 puede además intuirse que, en comparación con los sistemas FP, la media necesita más series para aproximarse al valor de convergencia. Para poder determinar con mayor precisión la evolución de la convergencia, en la Figura 4-33 se representan las evoluciones de la desviación absoluta, definida de la misma forma que en el punto anterior para sistemas FP, para las cuatro combinaciones de métodos de generación. En dicha figura se puede observar que la combinación *Custom-Uniform+UUnifast* es la que más series necesitó promediar para alcanzar la desviación absoluta objetivo de 0,5%, con 59 series promediadas, mientras que el resto de combinaciones de métodos de generación necesitaron 40 series de utilizaciones, o menos.

Para observar la influencia de los métodos de generación en la planificabilidad de sistemas LC-EDF, en la Figura 4-34 se recopilan los valores de convergencia de las medias de las utilizaciones máximas planificables para las cuatro combinaciones de métodos de generación. La primera conclusión que se puede obtener de la figura es que, para el mismo método de generación de periodos, el sesgo inducido por el método *SCALE-WCET* penaliza en mayor medida a la planificación LC-EDF que a FP. Esta mayor penalización se observa comprobando cómo la diferencia entre *SCALE-WCET* y *UUnifast* se sitúa entre el 15% y el 20%, frente al aproximadamente 10% observado en sistemas FP. En esta figura puede comprobarse también cómo el efecto de la dispersión de los periodos resulta más apreciable si se hace uso del algoritmo *Log-Uniform*, conclusión similar a la observada para sistemas FP, y monoprocesadores en [LEH89].

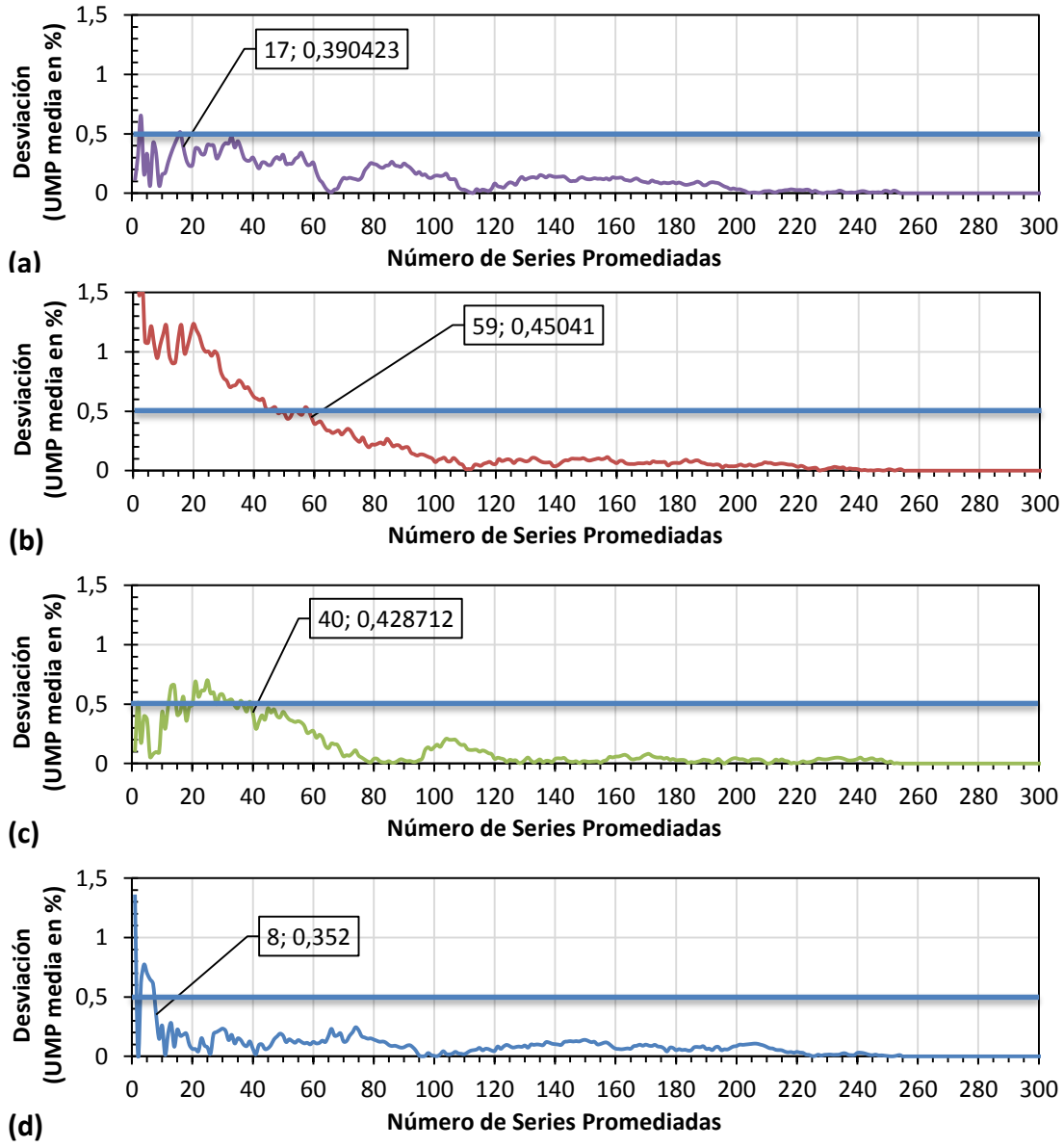


Figura 4-33 Desviación (distancia al valor de convergencia) en función del número de series promediadas, para sistemas LC-EDF y los métodos **(a)** ***Log-Uniform*** **+** ***UUnifast*****, (b)** ***Custom-Uniform*** **+** ***Uunifast*****, (c)** ***Log-Uniform*** **+** ***SCALE-WCET*****, (d)** ***Custom-Uniform*** **+** ***SCALE-WCET***

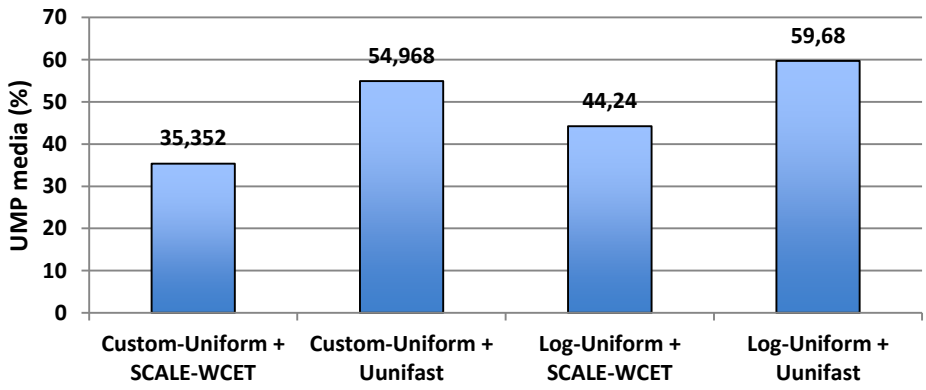


Figura 4-34 Media de las utilizaciones máximas planificables (UMP) para las cuatro combinaciones de métodos de generación, y sistemas LC-EDF

4.5.3 Sistemas distribuidos EDF con relojes globales (GC-EDF)

Para sistemas GC-EDF repetimos el estudio realizado anteriormente en sistemas FP y LC-EDF, y generamos 300 series de utilizaciones con sistemas con las características de la Tabla 4-2, aplicando las cuatro combinaciones disponibles de métodos de generación, con la diferencia de que ahora realizaremos el estudio con plazos de principio a fin $D_i=2*T_i$. Una consecuencia a tener en cuenta sobre esta decisión es que los valores absolutos de las utilizaciones máximas planificables que se presentarán en esta sección para GC-EDF no podrán compararse con los obtenidos previamente para FP y LC-EDF.

En la Figura 4-35 se muestra la evolución de la convergencia de las utilizaciones máximas planificables medias para cada una de las cuatro combinaciones de métodos de generación. Al igual que ocurría con FP y LC-EDF, con 300 series promediadas, el valor de la media ya se puede considerar como su valor final, representado con las etiquetas asociadas a cada línea.

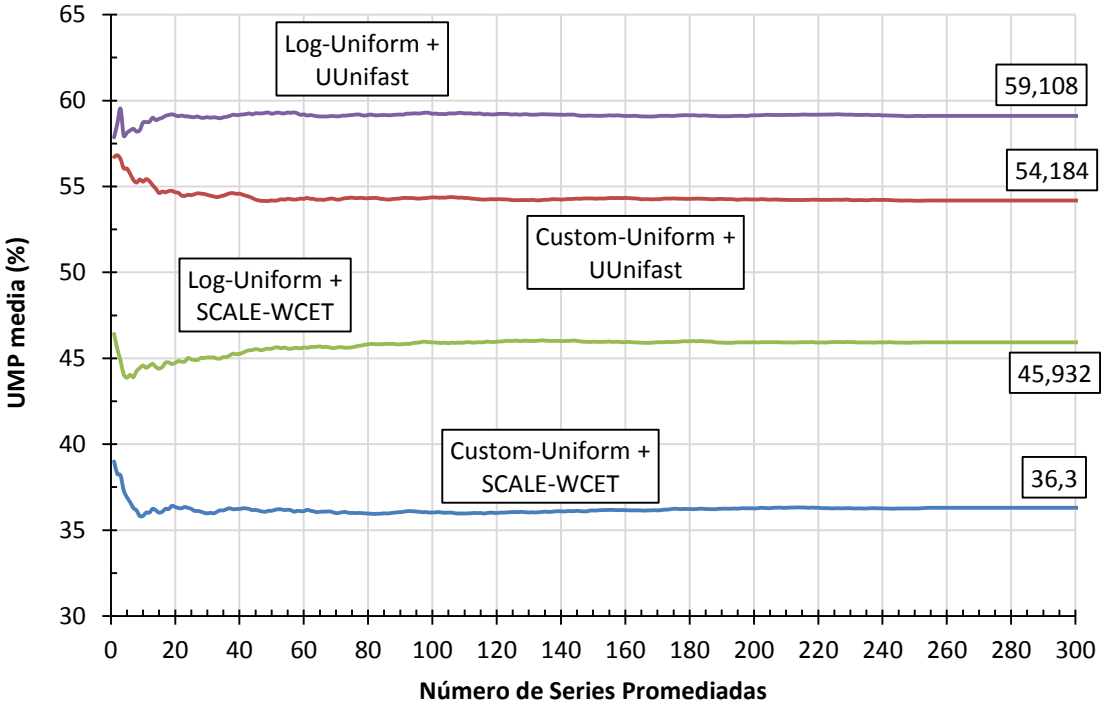


Figura 4-35 Convergencia de las medias de las utilizaciones máximas planificables en función del número de sistemas promediados, para las cuatro combinaciones de métodos de generación, y sistemas GC-EDF

En la Figura 4-36 se muestra la progresión de la desviación absoluta en función del número de series promediadas, definiendo la desviación absoluta de la misma manera que en los puntos anteriores para FP y LC-EDF. En la figura se puede observar que la combinación de métodos de generación que necesitó promediar el mayor número de series de utilizaciones para alcanzar el objetivo del 0,5% de desviación absoluta fue *Log-Uniform+SCALE-WCET*, con 43 series de utilizaciones.

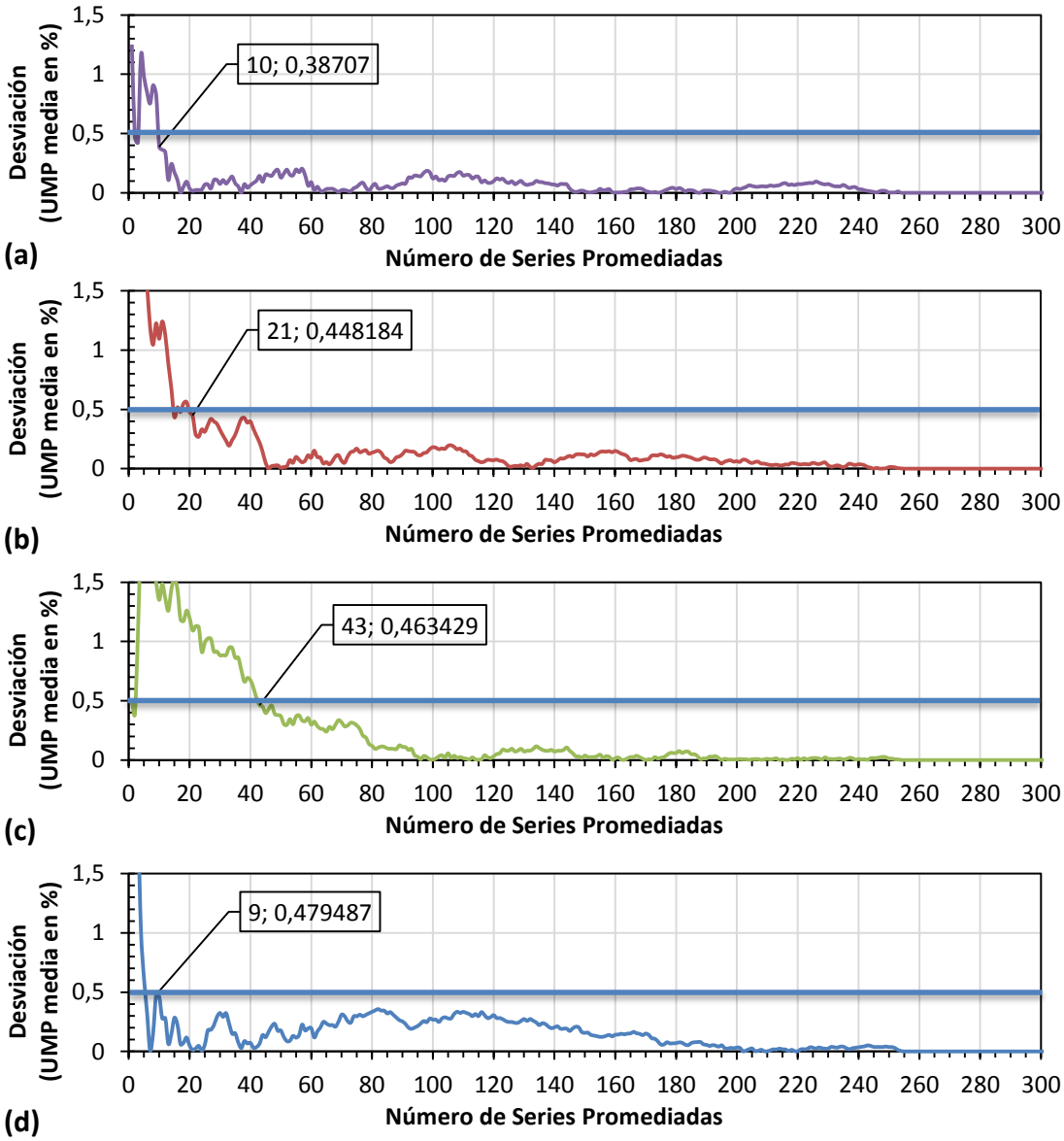


Figura 4-36 Desviación (distancia al valor de convergencia) en función del número de series promediadas, para sistemas GC-EDF y los métodos **(a)** ***Log-Uniform* + *UUnifast***, **(b)** ***Custom-Uniform* + *Uunifast***, **(c)** ***Log-Uniform* + *SCALE-WCET***, **(d)** ***Custom-Uniform* + *SCALE-WCET***

En la Figura 4-37 se recopilan los valores de convergencia de las medias de las utilizaciones máximas planificables obtenidas con las cuatro combinaciones de métodos de generación. Aunque estos resultados no se pueden comparar con los obtenidos con FP y LC-EDF por estar haciendo uso de plazos de principio a fin menores, se pueden extraer conclusiones similares. El sesgo en la utilización de las actividades provocado por el algoritmo *SCALE-WCET* se manifiesta con un empeoramiento entre el 15% y el 20% de utilización máxima planificable con respecto a las cargas calculadas con *UUnifast*. En la figura también puede observarse como la mayor dispersión de los periodos generados por *Log-Uniform* en comparación con *Custom-Uniform* provoca utilizaciones máximas planificables mayores

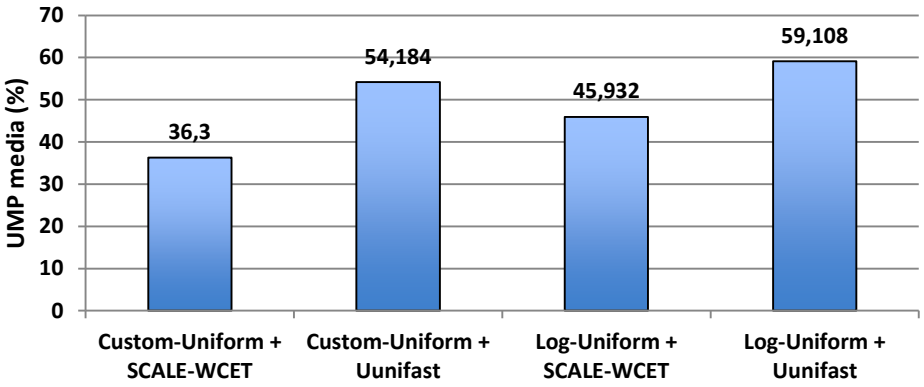


Figura 4-37 Media de las utilizaciones máximas planificables (UMP) para las cuatro combinaciones de métodos de generación, y sistemas GC-EDF

5 Estudio Comparativo de Técnicas Análisis de Planificabilidad en Sistemas Distribuidos

En el Capítulo 2 se introdujeron las principales técnicas de análisis de planificabilidad para sistemas distribuidos: 3 técnicas holísticas y 5 basadas en *offsets*. Las bondades y ventajas de cada una de estas técnicas quedan establecidas mediante los estudios y comparaciones que se incluyen en los trabajos en los que se presentan cada una de ellas.

Estos estudios ayudan a visualizar el rendimiento de las diferentes técnicas, pero poseen un carácter limitado, ya que abarcan por lo general una variedad reducida de sistemas, que además suelen representar aquellas situaciones en las que el algoritmo presentado posee mayor ventaja. Adicionalmente, en esas comparaciones es habitual que se incluya un número reducido de técnicas como contraste, por lo que se carece de una visión global, y bajo las mismas circunstancias, del rendimiento del conjunto total de técnicas.

En la actualidad no existe ningún trabajo que se plantee como objetivo la comparación global del el rendimiento de las diferentes técnicas de análisis y planificación de sistemas distribuidos para un amplio espectro de este tipo de sistemas. En el presente capítulo se quiere suplir este vacío, llevando a cabo un estudio comparativo masivo en el que se completen los estudios particulares realizados en los trabajos previos. Las principales características del estudio son:

- Se realiza utilizando el conjunto de herramientas MAST, y la herramienta GEN4MAST para la generación de los sistemas. Todas las herramientas se

ofrecen como código abierto, por lo que los resultados del estudio son fácilmente replicables.
- El estudio abarca una gran cantidad de tipos de sistemas, con variedad de tamaños y complejidades.
- Se busca obtener resultados estadísticamente relevantes.
- Las diferentes técnicas de análisis se comparan utilizando el mismo conjunto de sistemas, por lo que se pueden realizar comparaciones directas de rendimiento.
- El conjunto de sistemas se diseña con la intención de abarcar el mayor espectro posible, y con el objetivo de obtener la mayor cantidad de información posible sobre el funcionamiento de cada técnica en diferentes circunstancias. *A priori* el diseño de los sistemas del estudio no favorece a ninguna técnica en particular.

Una vez establecidos los objetivos, el primer paso necesario para realizar el estudio es el de definir el conjunto de sistemas sobre el que se va a realizar. Esta definición no resulta un proceso trivial, puesto que en él se deben considerar varios factores contrapuestos, y conseguir encontrar una solución balanceada. El proceso de definición del conjunto de sistemas se explica en la siguiente sección.

5.1 Dominio del estudio

El estudio que aquí queremos llevar a cabo tiene como objetivo comprobar cómo se comportan las diferentes técnicas de análisis con sistemas en los que se varían sus características básicas, como por ejemplo el número de flujos e2e, actividades, procesadores, etc. Llamamos dominio del estudio al espectro de características de los sistemas que van a formar parte de él. En el campo de los sistemas empotrados distribuidos, se pueden encontrar sistemas reales con características muy dispares. Por ejemplo, dos sistemas pueden tener el mismo número de procesadores e incluso una carga muy parecida, pero debido a las diferencias en el tipo de aplicación, no parecerse en nada en el número de flujos e2e y su longitud, en las frecuencias de activación de los flujos e2e, o incluso en los plazos que se deben cumplir. Para definir el dominio del estudio de las técnicas de análisis de planificabilidad, vamos a tener en cuenta dos factores fundamentales: la representatividad de los sistemas que lo componen, y el tiempo de ejecución total del estudio.

En general, los estudios sobre técnicas de sistemas de tiempo real, como el análisis de planificabilidad o la asignación de parámetros de planificación, se llevan a cabo sobre sistemas sintéticos. Por ejemplo, en [GUT95] se trabaja con un sistema compuesto por 93 actividades localizadas en 11 recursos procesadores para estudiar el algoritmo HOPA. En [TIN94] se estudia el análisis holístico para FP en un sistema con 46 actividades en 4 recursos. En [PAL03] se analizan sistemas formados por 100 actividades localizadas en 4 recursos procesadores para estudiar el rendimiento del análisis basado en *offsets* para GC-EDF. Mientras que en la literatura resulta sencillo encontrar casos de estudio sintéticos, es menos común encontrarnos con especificaciones de sistemas utilizados en aplicaciones reales, debido principalmente a la confidencialidad de esta información en sistemas comerciales.

Como sustitutos de los sistemas reales, se suelen aportar sistemas simplificados que incluyen con mayor o menor precisión las características de los sistemas reales. Por ejemplo, en [JAY08] se utiliza un sistema de control de vuelo simplificado compuesto

por 13 actividades en tres flujos e2e que recorren 8 recursos procesadores. En [MUB12] se describe un sistema de control de crucero activo para automóviles, compuesto por 4 procesadores y una red CAN. También aplicado en sistemas de automoción, en [DEU11] se modelan cargas reales utilizando sistemas con entre 20 y 30 actividades, periodos entre 2,5 y 50 ms., utilizaciones de las actividades (C_{ij}/T_i) entre 0,05 y 0,5, y plazos iguales a los periodos.

En ocasiones, las propias empresas que diseñan los sistemas reales suministran casos de estudio que emulan situaciones reales con las que deben tratar, con el objetivo de incitar un debate en la comunidad para desarrollar soluciones a problemas reales. Por ejemplo, en [CHA06] se estudian tres métodos para calcular tiempos de respuesta de peor caso de flujos e2e usando redes AFDX, como la utilizada en el Airbus A380. Para ello, hacen uso de un caso de estudio suministrado por Airbus, compuesto por 123 procesadores (*end systems* en terminología de AFDX*)*, 8 conmutadores AFDX, y 964 flujos e2e (*virtual links* en terminología de AFDX). Otro ejemplo lo encontramos en [EAD08][PER09b], en el que la empresa aeroespacial EADS propone un caso de estudio compuesto por 192 procesadores conectados a un servidor central mediante una red de tipo Ethernet para tiempo real (como AFDX), con el objetivo de encontrar soluciones que minimicen el *jitter* y los tiempos de respuesta de peor caso en flujos e2e. Por último, en [HEN15], la división de I+D de Thales (*Thales Research and Technology*) propone un reto consistente en calcular tiempos de respuesta de mejor y peor caso, y asignaciones de prioridades fijas, en un sistema de seguimiento de video aéreo, compuesto por 4 recursos procesadores, 7 actividades agrupadas en 2 flujos e2e, y diversos escenarios con diferentes *jitters* en los estímulos de entrada. Los ejemplos aquí citados, tanto sintéticos como basados en aplicaciones reales, nos dan una cierta intuición sobre el tamaño de los sistemas a los que debemos apuntar en el dominio del estudio que queremos definir en este capítulo.

Adicionalmente a la representatividad de los ejemplos a incluir en el dominio, el segundo factor que tendremos en cuenta a la hora de establecer el dominio del estudio está relacionado con el tiempo necesario para llevarlo a cabo. A medida que aumenta la complejidad o el tamaño de un sistema, el tiempo necesario para ejecutar el análisis de planificabilidad aumenta de manera exponencial. Por ejemplo, en estudios preliminares previos se comprobó cómo para analizar un sistema compuesto por 200 actividades situadas en dos procesadores, el análisis basado en *offsets* para GC-EDF requirió hasta 8 horas para completar su ejecución. Teniendo en cuenta que el presente estudio requerirá de cientos de miles de ejecuciones de este tipo, hay que buscar un equilibrio entre el número de ejemplos a generar, y la capacidad de cómputo que tenemos disponible para llevarlo a cabo.

El *cluster* de computación Tres Mares disponible en el Departamento de Ingeniería Informática y Electrónica de la Universidad de Cantabria nos permitirá ejecutar GEN4MAST sobre un extenso dominio de estudio, que no podría alcanzarse en un PC convencional. Sin embargo, tener acceso a un *cluster* de computación no es sinónimo de potencia de cálculo ilimitada. Estos sistemas son un medio de cómputo compartido por un conjunto de usuarios que pueden hacer uso de él de forma concurrente. Por lo tanto, un uso responsable del *cluster* nos obliga a limitar su tiempo de uso, para poder acomodar los requisitos de cálculo de su conjunto de usuarios.

Teniendo en cuenta los factores descritos, diseñamos un dominio de estudio que estará formado por sistemas cuyas características pivotarán alrededor de un sistema central, al que llamamos *sistema base*, que tiene las siguientes características:

- Planificación FP, GC-EDF y LC-EDF.
- Número de flujos: 10
- Número de actividades por flujo e2e (longitud del flujo): 10
- Número de recursos procesadores: 5.
- Localización de las actividades pseudo-aleatoria: si es posible, los flujos e2e no repetirán recurso procesador (podrá suceder cuando se varíen el número de actividades por flujo e2e).
- Plazos de principio a fin: $D_i=N_i*T_i$
- Periodos: se seleccionan aleatoriamente en el rango [100,1000] (Ratio=10), mediante una distribución de probabilidad logarítmico-uniforme (*Log-Uniform*)
- Utilización de las actividades calculadas mediante el algoritmo *UUnifast*.
- Tiempos de ejecución de mejor caso de las actividades igual a 0.
- Todos los procesadores poseen la misma utilización en todo momento.

Este sistema base está compuesto por 100 actividades situadas en 5 procesadores, lo que conlleva que en cada procesador residan en promedio 20 actividades.

Tomando como referencia este sistema base, se crean nuevos sistemas variando los valores de algunas de sus características, con el fin de observar cómo esa variación influye en la planificabilidad global del sistema. Las características cuya variación va a ser estudiada son las siguientes:

- Número de recursos procesadores.
- Longitud de los flujos e2e.
- Número de flujos e2e.
- Rango de los periodos.

El estudio que vamos a llevar a cabo por lo tanto se dividirá en 4 sub-estudios que abarcarán estas variaciones, y que serán generados por GEN4MAST. Adicionalmente, y de forma puntual, se podrá estudiar el efecto de modificar alguna otra característica asociada.

Para cada combinación de características de los sistemas, se crearán series de utilizaciones en el rango [0,96] (en %), según lo descrito en la Sección 4.2.1.5. Dado que el tiempo de ejecución crece exponencialmente con la utilización, decidimos limitar el valor máximo de utilización generado al 96%, lo que reduce drásticamente el tiempo de ejecución total del estudio, manteniendo la capacidad de obtener conclusiones válidas.

Para obtener resultados estadísticamente relevantes, se repetirá el proceso de generación 30 veces. El resultado es que se crean 30 series de utilizaciones para cada combinación de parámetros de generación. En la Sección 4.5 se comprobó que promediando 30 series de utilizaciones se consigue acotar la desviación absoluta a un 0.5% en la mayoría de los casos. En las peores situaciones, con promedios con 30 series de utilizaciones la desviación absoluta era de menos del 1%.

A continuación se describen con más detalle cada uno de estos 4 sub-estudios.

5.1.1 Variación del número de recursos procesadores

Los flujos e2e del sistema base recorren de manera aleatoria, y al menos en una ocasión, cada uno de los 5 procesadores del sistema. El primer sub-estudio consistirá en

variar el número de procesadores, y comprobar qué efectos induce este cambio en los resultados de planificabilidad de las diferentes técnicas de análisis. Para completar las conclusiones, estas variaciones se llevaran a cabo en tres situaciones distintas: con flujos e2e de longitud 5, 10 y 20 actividades, de forma que se pueda observar el efecto de la repetición de un mismo recurso dentro de los flujos.

Modificar el número de recursos procesadores implica re-localizar las actividades en los recursos disponibles. La forma de localizar las actividades también será estudiada. Para ello, se utilizarán dos métodos distintos:

- Localización pseudo-aleatoria: Las actividades se sitúan de forma aleatoria en los procesadores del sistema, pero forzando, en la medida de lo posible, que no se repita un recurso procesador dentro del flujo e2e. Este es el comportamiento del sistema base, y en GEN4MAST se corresponde con el parámetro "REPETITION NO".
- Localización aleatoria: Las actividades se localizan en un procesador de manera totalmente aleatoria. Esta localización puede dar lugar a que algún sistema tenga algún recurso procesador libre de actividades. Se corresponde con el parámetro "REPETITION YES".

En la Tabla 5-1 se resumen las características que se variarán, junto con sus valores. En verde se destacan los parámetros del sistema base.

Número de procesadores	2	3	**5**	8	10	15	20
Longitud de los flujos e2e	5		**10**		20		
Localización de las actividades	Aleatoria			**Semi Aleatoria**			

Tabla 5-1 Variaciones para el estudio de la influencia del número de recursos procesadores.

Las variaciones de este sub-estudio se generan mediante el fichero de configuración de GEN4MAST mostrado en la Figura 5-1.

```
[GENERATOR]
POPULATION 30
N_FLOWS 10
N_STEPS 5 10 20
SINGLE_FLOWS 0
FIXED_LENGTH True
N_PROCESSORS 2 3 5 8 10 15 20
REPETITION YES NO
SCHEDULING_POLICY FP EDF
PERIOD_BASE 100
PERIOD_RATIOS 10
PERIOD_DISTRIBUTION LOG-UNIFORM
DEADLINES NT
UTILIZATION_START 10
UTILIZATION_STEP 1
UTILIZATION_STOP 96
UNIFORM_UTILIZATION
WORKLOAD UUNIFAST
BEST_CASE 0
[/GENERATOR]
```

Figura 5-1 Parámetros de generación para GEN4MAST del estudio de la influencia del número de recursos procesadores.

5.1.2 Variación de la longitud de los flujos e2e

Es evidente que añadir nuevas actividades a un flujo e2e, y por tanto aumentar la utilización en el sistema, manteniendo los plazos de principio a fin, tiene un efecto negativo en la planificabilidad del sistema. Para comprobar el efecto real producido por la longitud de los flujos, lo que vamos a comprobar es el efecto de repartir una misma carga computacional (dada en forma de la utilización total del sistema) en sistemas con flujos e2e de distintas longitudes. Para ello, a partir del sistema base, se van a generar sistemas cuyos flujos e2e tienen longitudes de 3, 5, 10, 15 y 20 actividades, manteniendo el número de procesadores constante en 5. Los plazos de principio a fin se escalan automáticamente a la longitud de los flujos e2e, ya que les asignamos un valor de $D_i=N_i*T_i$, donde N_i es la longitud del flujo Γ_i.

Tal y como se definió en la Sección 2.1, el *jitter* de una actividad se calcula como la variación entre los tiempos de respuesta de peor y mejor caso de la actividad precedente en el flujo e2e. Por lo tanto, aumentar la longitud de los flujos e2e tiene como consecuencia el aumento de este efecto de propagación. Adicionalmente, cuando las actividades tienen un tiempo de repuesta de mejor caso mayor que 0, los tiempos de respuesta de mejor caso obtenidos son también mayores que cero, por lo que se reducirá el *jitter* propagado en el flujo. Por ello, el estudio de la influencia de la longitud se va a completar planteando dos situaciones distintas:

- Todas las actividades tienen un tiempo de ejecución de mejor caso igual a 0 (Cb_{ij} = 0).
- Todas las actividades tienen un tiempo de ejecución de mejor caso igual al de peor caso ($Cb_{ij} = C_{ij}$).

▪ Longitud de los Flujos	3	5	**10**	15	20
Tiempos de ejecución de mejor caso	**0**			C_{ij}	

Tabla 5-2 Variaciones para el estudio del de la influencia de la longitud de los flujos.

```
[GENERATOR]
POPULATION 30
N_FLOWS 10
N_STEPS 3 5 10 15 20
SINGLE_FLOWS 0
FIXED_LENGTH True
N_PROCESSORS 5
REPETITION NO
SCHEDULING_POLICY FP EDF
PERIOD_BASE 100
PERIOD_RATIOS 10
PERIOD_DISTRIBUTION LOG-UNIFORM
DEADLINES NT
UTILIZATION_START 10
UTILIZATION_STEP 1
UTILIZATION_STOP 96
UNIFORM_UTILIZATION
WORKLOAD UUNIFAST
BEST_CASE 0 100
[/GENERATOR]
```

Figura 5-2 Parámetros de generación para GEN4MAST del estudio de la longitud de los flujos.

En la Tabla 5-2 se muestran a modo de resumen los valores de todas estas variaciones (en verde las características del sistema base), que se mapean en los parámetros de generación de GEN4MAST mostrados en la Figura 5-2.

5.1.3 Variación del número de flujos e2e

Para los estudios de la variación del número de procesadores y de la longitud de los flujos se han generado sistemas en los que el número de flujos se mantiene constante. El siguiente sub-estudio consiste en comprobar los efectos de variar el número de flujos e2e.

El objetivo de variar el número de flujos es el de observar si, para una carga computacional concreta, su distribución en un número mayor o menos de flujos e2e tiene influencia en la planificabilidad del sistema. Para ello se van a generar sistemas, que partiendo de las características del sistema base, van a tener 1, 5, 10, 15 y 20 flujos e2e. Esta variación se llevará a cabo con tres situaciones distintas: flujos e2e con longitudes de 5, 10 y 20 actividades.

Entre todos los valores de las variaciones observamos que hay tres casos en los que se tienen 100 actividades pero distribuidas de manera diferente: 10 flujos e2e de longitud 10, 20 flujos e2e de longitud 5, y 5 flujos e2e de longitud 20. Con los resultados obtenidos con estos tres sub-conjuntos de sistemas se podrá comprobar el efecto de tener menos flujos largos, o más flujos cortos, para repartir el mismo número de actividades.

En la Tabla 5-3 se resumen las características que se variarán junto con sus valores (sombreados en verde aparecen las características del sistema base), que se generan con los parámetros de GEN4MAST mostrados en la Figura 5-3.

Número de flujos e2e	1	5	**10**	15	20
Longitud	5		**10**	20	

Tabla 5-3 Variaciones para el estudio de la influencia del número de flujos.

```
[GENERATOR]
POPULATION 30
N_FLOWS 1 5 10 15 20
N_STEPS 5 10 20
SINGLE_FLOWS 0
FIXED_LENGTH True
N_PROCESSORS 5
REPETITION NO
SCHEDULING_POLICY FP EDF
PERIOD_BASE 100
PERIOD_RATIOS 10
PERIOD_DISTRIBUTION LOG-UNIFORM
DEADLINES NT
UTILIZATION_START 10
UTILIZATION_STEP 1
UTILIZATION_STOP 96
UNIFORM_UTILIZATION
WORKLOAD UUNIFAST
BEST_CASE 0
[/GENERATOR]
```

Figura 5-3 Parámetros de generación para GEN4MAST del estudio de la influencia del número de flujos.

5.1.4 Variación del rango de los periodos

En GEN4MAST los periodos se eligen aleatoriamente dentro del rango especificado. En aplicaciones de tiempo real es habitual eventos con periodos que difieren en varios órdenes de magnitud [BUR06]. Para estudiar cómo afecta esta dispersión de periodos, vamos a generar sistemas con distintas ratios de periodos. Los valores de las ratios que estudiaremos serán 1, 10, 100 y 1000, o lo que es lo mismo, se estudiarán respectivamente sistemas con todos los periodos iguales, con todos los periodos en el mismo orden de magnitud, en dos órdenes de magnitud, y en tres órdenes de magnitud. Se utilizarán las dos funciones de distribución disponibles para la selección de los periodos (ver Sección 4.2.1.3): *Custom-Uniform* y *Log-Uniform*.

Según el modelo que utilizamos en este trabajo, la única entidad del sistema a la que se le asigna un periodo es al evento externo que activa el flujo e2e. Por lo tanto, la influencia que los periodos puedan tener en la planificabilidad está relacionada con el número flujos e2e que haya. Por esta razón este sub-estudio se llevará a cabo con sistemas en los que además se variará el número de flujos e2e, con valores de 1, 10 y 20.

En la Tabla 5-4 se resumen las características que se variarán junto con sus valores (sombreados en verde aparecen las características del sistema base), que se mapean en los parámetros de GEN4MAST mostrados en la Figura 5-4.

Ratio Periodos	1	**10**	100	1000
Función de Distribución	Custom-Uniform		**Log-Uniform**	
Número de Flujos	1	**10**	20	

Tabla 5-4 Variaciones para el estudio de la influencia del rango de selección de los periodos.

```
[GENERATOR]
POPULATION 30
N_FLOWS 1 10 20
N_STEPS 10
SINGLE_FLOWS 0
FIXED_LENGTH True
N_PROCESSORS 5
REPETITION NO
SCHEDULING_POLICY FP EDF
PERIOD_BASE 100
PERIOD_RATIOS 1 10 100 1000
PERIOD_DISTRIBUTION CUSTOM-UNIFORM LOG-UNIFORM
DEADLINES NT
UTILIZATION_START 10
UTILIZATION_STEP 1
UTILIZATION_STOP 96
UNIFORM_UTILIZATION
WORKLOAD UUNIFAST
BEST_CASE 0
[/GENERATOR]
```

Figura 5-4 Parámetros de generación para GEN4MAST del estudio de la influencia del rango de selección de los periodos.

5.2 Ejecución del estudio

Una vez que se han generado todos los ejemplos que forman parte del dominio del estudio, el siguiente paso es el de aplicarles las técnicas de análisis para sistemas distribuidos disponibles en MAST que son nuestro objeto de estudio. Estas son, para cada algoritmo de planificación:

- FP: Técnica holística, técnica basada en *offsets*, técnica basada en *offsets slanted*, y técnica basada en *offsets* con relaciones de precedencia. La técnica de análisis basada en *offsets* de fuerza bruta, debido a la intratabilidad de sus tiempos de ejecución, se aplicará sobre un subconjunto reducido de sistemas con complejidad reducida.
- GC-EDF: Técnica holística, y la técnica basada en *offsets.*
- LC-EDF: Técnica holística.

En la Tabla 5-5 se resumen las técnicas de análisis disponibles que se comparan en el estudio. A lo largo del estudio se identificará a cada técnica de análisis con el nombre inglés (mostrado en la tabla) al resultar una nomenclatura más compacta.

	Holistic	***Offset based***	***Offset based slanted***	***Offset based w/pr***	***Offsets based brute force***
FP	✓	✓	✓	✓	✓*
GC-EDF	✓	✓			
LC-EDF	✓				

*Sistemas simplificados

Tabla 5-5 Conjunto de técnicas de análisis a estudiar, y algoritmos de planificación de los sistemas sobre los que se aplica.

El principal resultado que se obtiene al aplicar una de las técnicas de análisis de planificabilidad son los tiempos de respuesta de peor caso de las actividades del sistema. Para determinar la planificabilidad del sistema se comparan estos tiempos de respuesta con los requisitos temporales dados por los plazos. En el estudio que vamos a llevar a cabo, los requisitos se imponen siempre a la última actividad de los flujos e2e. Todas las técnicas que vamos a comparar son pesimistas en mayor o menor medida, en el sentido de que obtienen cotas superiores de estos tiempos de respuesta de peor caso.

Una métrica que utilizaremos para comparar las bondades de las diferentes técnicas de análisis de planificabilidad es la utilización máxima planificable (UMP). Éste es un resultado que guarda GEN4MAST (ver Sección 4.4) para cada ejecución de cada una de las técnicas de análisis bajo estudio sobre cada serie de utilizaciones generada. Mostraremos el valor medio y la desviación típica de las UMP en las 30 series de utilizaciones generadas sobre la misma base para obtener resultados estadísticamente relevantes.

Otra figura de mérito que observaremos es la evolución de los tiempos de respuesta de peor caso obtenidos por las diferentes técnicas de análisis en función de la utilización total del sistema, comparando esta evolución ante diferentes variaciones de las características de los sistemas. Los tiempos de respuesta que resultan relevantes son los de la totalidad del flujo e2e (tiempo de respuesta de la última actividad), puesto que son los que se comparan con los plazos de principio a fin. Debido a que los sistemas están formados por varios flujos e2e, para la simplificación de la presentación de los resultados

se mostrará el promedio de los tiempos de respuesta de peor caso de todos los flujos e2e del sistema.

Para un sistema concreto, y una técnica de análisis de planificabilidad *X*, definimos *R(X)* como el promedio de los tiempos de respuesta de peor caso de los flujos e2e de dicho sistema, calculados con la técnica de análisis de planificabilidad *X*. El concepto se formaliza en la siguiente ecuación:

$$R(X) = \frac{\sum_{\forall i} R_{iN_i}}{N_\Gamma} \tag{5-1}$$

donde N_i es el número de actividades del flujo e2e Γ_i, y N_Γ es el número de flujos e2e del sistema.

Con objeto de poder observar las diferencias con mayor claridad, *R(X)* se presentará normalizado a los valores obtenidos por una técnica determinada. Si son normalizados a los valores obtenidos por *holistic*, a estos valores normalizados los denominaremos *R_norm(holistic*), concepto formalizado en la siguiente ecuación:

$$R\_\text{norm}(\text{holistic}) = \frac{R(X)}{R(holistic)} \tag{5-2}$$

Para poder analizar un sistema, todas sus actividades deben tener asignado un parámetro de planificación. Para este estudio comparativo de técnicas de análisis utilizamos la asignación PD en todos los casos, ya que nos proporciona un método sencillo (es una técnica no iterativa), suficientemente robusto, y cuya asignación es independiente de la técnica de análisis, de forma que no interfiera en la comparativa que se quiere realizar.

Para limitar el tiempo de análisis en sistemas no planificables, utilizamos el parámetro STOP_FACTOR de GEN4MAST (Ver Sección 4.2.2) con un valor de 5. Además, indicamos a GEN4MAST que una vez que un análisis es detenido por el STOP_FACTOR, no se continúe analizando los siguientes sistemas de la serie de utilizaciones.

En la Figura 5-5 se muestran los parámetros usados en GEN4MAST para la especificación de la ejecución de este estudio. El estudio para el análisis basado en *offsets* de fuerza bruta ("ANALYSIS_TOOL BRUTE_FORCE") se llevará a cabo sobre un dominio de estudio simplificado (ver Sección 5.3.5).

```
[EXECUTION]
MAST_PATH mast_analysis
ANALYSIS_TOOL HOLISTIC OFFSET SLANTED OFFSET_OPT
ASSIGNMENT_TOOL PD
STOP_FACTOR 5
[/EXECUTION]
```

Figura 5-5 Parámetros de GEN4MAST para la definición de la ejecución del estudio comparativo de técnicas de análisis

La presentación de los resultados de cada uno de los sub-estudios se realiza para cada política de planificación por separado (FP, GC-EDF y LC-EDF).

5.3 Planificación FP

Presentamos los resultados de los cuatro sub-estudios para sistemas planificados por FP.

5.3.1 Influencia del número de recursos procesadores

Para estudiar la influencia del número de procesadores se generaron sistemas que partiendo del sistema base, variaban su número de procesadores entre los valores 2, 3, 5, 8, 10, 15 y 20, con flujos de longitud 5, 10 y 20, y localización de las actividades pseudo-aleatoria y aleatoria.

En la Figura 5-6 se muestran las UMP medias obtenidas por *holistic* en función del número de procesadores del sistema y la longitud de los flujos e2e, cuando las actividades se localizan de forma pseudo-aleatoria. Puede observarse cómo, para una longitud constante, la variación del número de procesadores tiene una influencia limitada, observándose una disminución de la UMP media alcanzada cuando los flujos e2e son de longitud 5 y se dispone de más procesadores. Este efecto se explica en el hecho de que los flujos e2e cortos sólo recorren un número reducido de procesadores, lo que provoca que estadísticamente pueda existir algún procesador que quede libre de actividades, o lo que es lo mismo, con utilización del 0%.

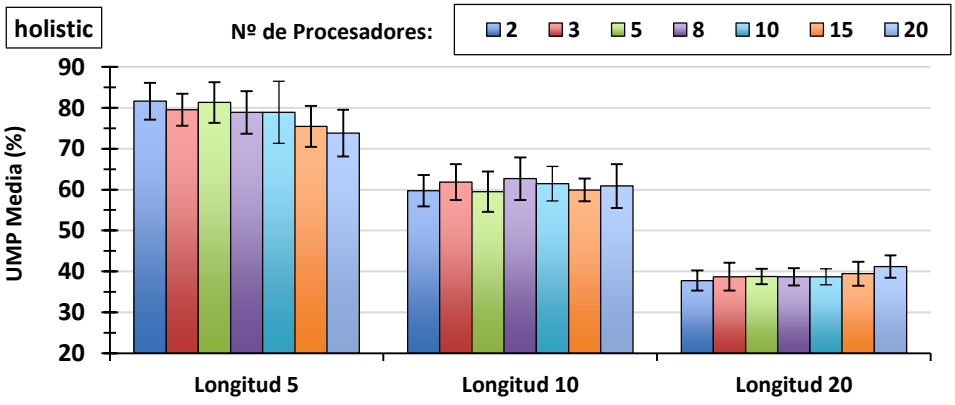


Figura 5-6 Influencia del número de procesadores en la UMP media de sistemas FP analizados con *holistic*, para flujos de longitud 5, 10 y 20, y **localización de actividades pseudo-aleatoria**.

Las técnicas basadas en *offsets* se crean con la idea de reducir el pesimismo de la técnica holística, es decir, obtener cotas más pequeñas de los tiempos de respuesta de peor caso. Para observar este nivel de mejora, en la Figura 5-7 se muestran las ratios de las UMP medias obtenidas por las técnicas basadas en *offsets* sobre el análisis holístico, cuando la localización es pseudo-aleatoria. En la figura se observa cómo las técnicas (a) *offset based* y (b) *offset based slanted* obtienen mejoras sobre *holistic* en las situaciones en las que existen más actividades por procesador. Estas mejoras son reducidas (<=1%), y ligeramente superiores en el análisis *offset based slanted*. Sin embargo, en la Figura 5-7(c) puede observarse cómo el análisis *offset based w/pr* es claramente superior al resto en las situaciones en las que en un mismo flujo se recorren los mismos procesadores en más de una ocasión, con mejoras del hasta el 16% cuando los procesadores se repiten hasta 10 veces en un mismo flujo (longitud 20 en 2 procesadores). Cuando no ocurren estas repeticiones (la longitud del flujo e2e es menor o igual que el número de procesadores), el análisis *offset based w/pr* no produce ninguna mejora sobre *holistic*.

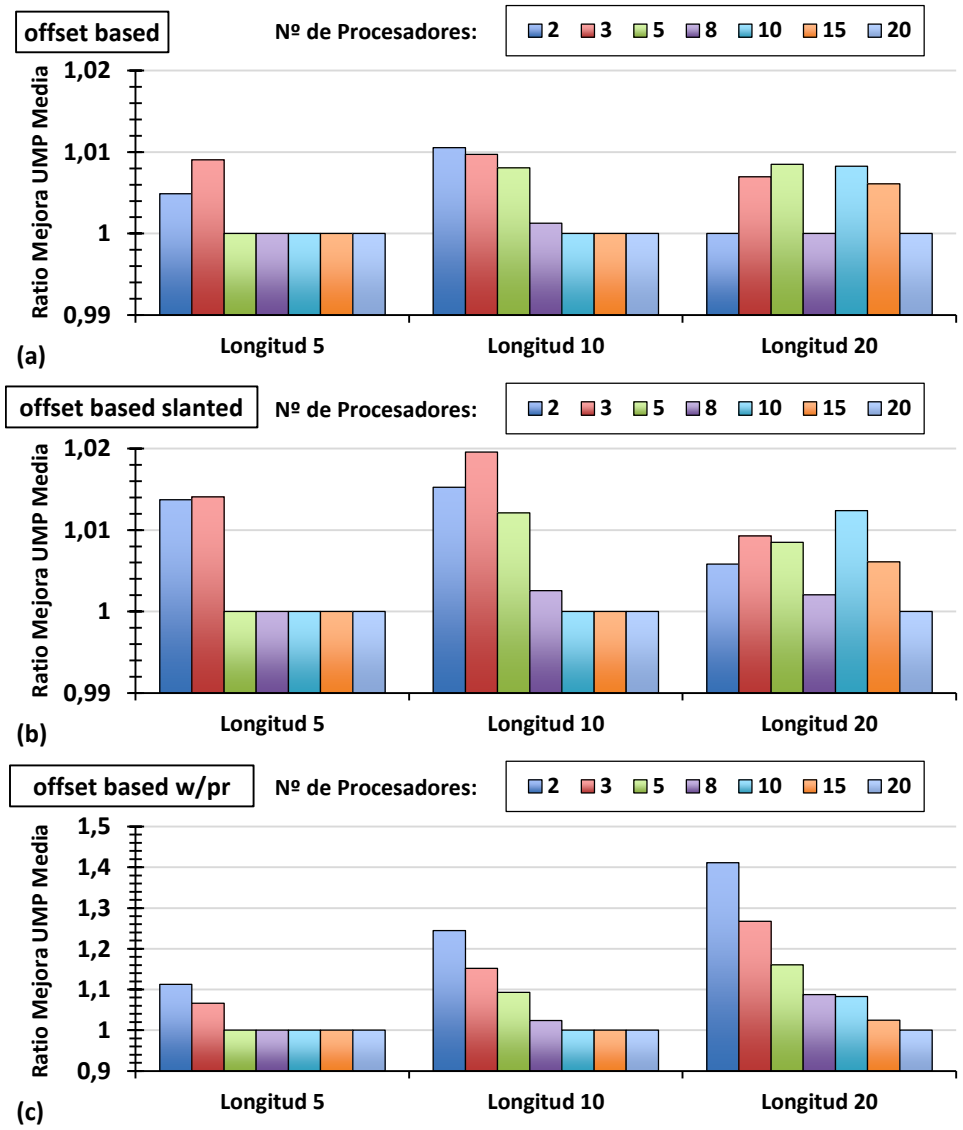


Figura 5-7 Influencia del número de procesadores en las mejoras sobre *Holistic*, en términos de UMP media, de las técnicas de análisis **(a) *offset based,* (b) *offset based slanted*, (c) *offset based w/pr***, para flujos e2e de longitud 5, 10 y 20, y **localización de actividades pseudo-aleatoria**.

En la Figura 5-8(a) se representa la evolución del promedio de los tiempos de respuesta de peor caso normalizados a *holistic* (denominado *R_norm*(*holistic*)) de las técnicas basadas en *offsets*, para los sistemas con 2 procesadores, y flujos de longitud 10. Puede observarse cómo las diferencias observadas con las UMP medias se replican con los tiempos de respuesta. Los análisis *offset based* y *offset based slanted* tienen un rendimiento similar, con una ligera ventaja siempre de éste último. Sin embargo, las optimizaciones de precedencia de *offset based w/pr* hacen que se distancie claramente, sobre todo para utilizaciones más altas. La mayor variabilidad que se observa en las utilizaciones más altas se debe al menor número de puntos promediados, ya que menos sistemas consiguen ser planificados en tales utilizaciones.

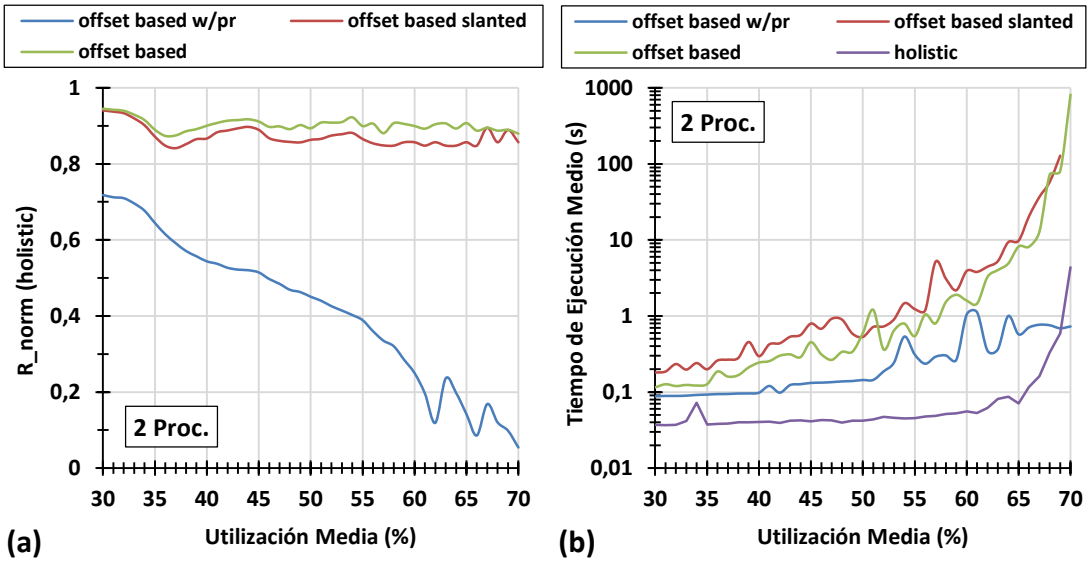


Figura 5-8 Evolución de los **(a) tiempos de respuesta medios normalizados a los valores de *holistic*, y (b) los tiempos de ejecución**, para sistemas con 2 procesadores, flujos e2e de longitud 10, y **localización de actividades pseudo-aleatoria**.

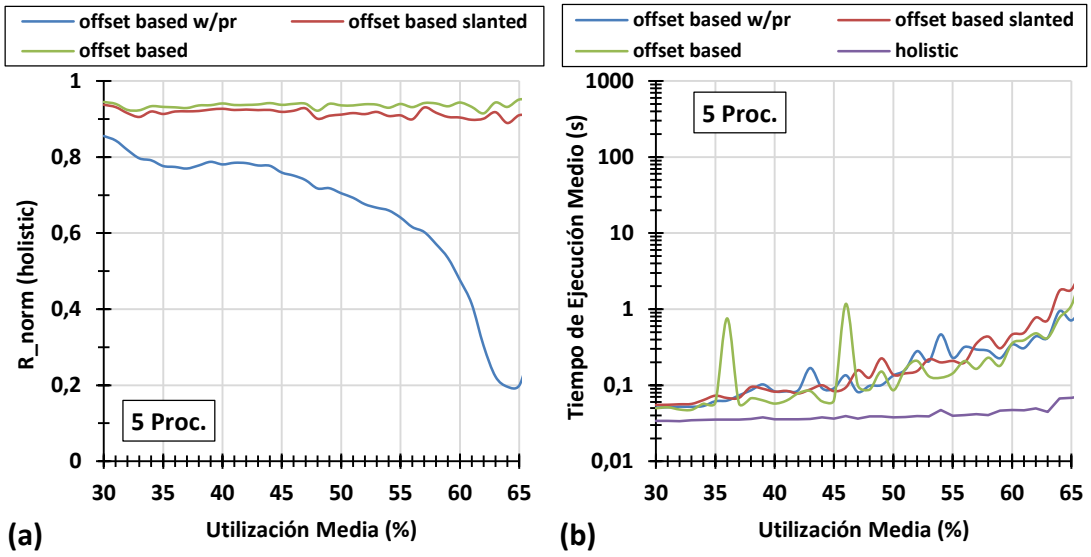


Figura 5-9 Evolución de los **(a) tiempos de respuesta medios normalizados a los valores de *holistic*, y (b) los tiempos de ejecución**, para sistemas con 5 procesadores, flujos e2e de longitud 10, y **localización de actividades pseudo-aleatoria**.

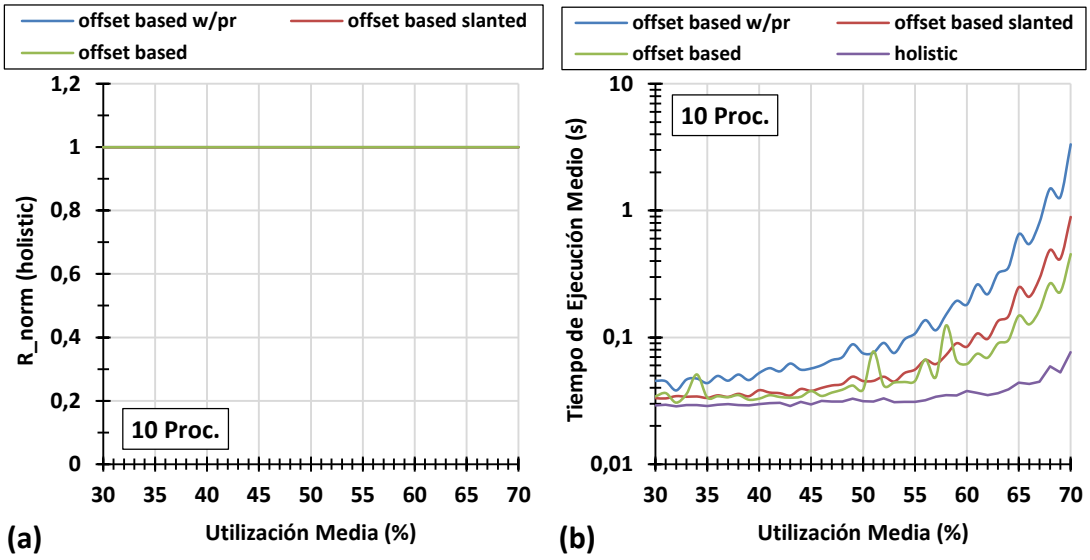


Figura 5-10 Evolución de los **(a) tiempos de respuesta medios normalizados a los valores de *holistic*, y (b) los tiempos de ejecución,** para sistemas con 10 procesadores, flujos e2e de longitud 10, y localización de actividades pseudo-aleatoria.

La menor cota de los tiempos de respuesta de peor caso que ofrecen las tres técnicas basadas en *offsets* se obtienen mediante algoritmos de análisis más complejos, que requieren más tiempo de ejecución. En la Figura 5-8(b) se pueden observar estos incrementos de tiempo con respecto a los requeridos por *holistic*. Una conclusión llamativa de la figura es observar que los algoritmos *offset based* y *offset based slanted*, a pesar de no ser los que obtienen cotas más ajustadas, son los que en promedio requirieron mayores tiempos de ejecución. Las optimizaciones por las relaciones de precedencia consiguen comparativamente unos análisis más rápidos para sistemas con 2 procesadores.

Repetimos el proceso seguido para obtener la figura anterior, pero para las situaciones con 5 y 10 procesadores, y lo representamos en la Figura 5-9 y en la Figura 5-10 respectivamente. Para el caso de 5 procesadores, la menor cota en los tiempos de respuesta de peor caso obtenidos por *offset based w/pr* continúa siendo evidente, pero la diferencia con respecto del resto de técnicas se ha visto reducida con respecto al caso con 2 procesadores. En los tiempos de ejecución asociados comprobamos cómo en esta ocasión las técnicas *offset based* y *offset based slanted* ahora requieren un tiempo comparable a *offset based w/pr*.

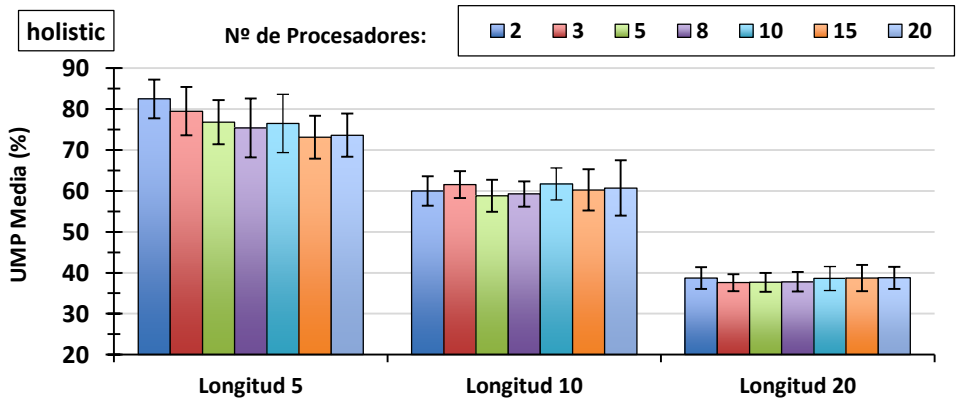


Figura 5-11 Influencia del número de procesadores en la UMP media de sistemas FP analizados con *holistic*, para flujos e2e de longitud 5, 10 y 20, y **localización de actividades aleatoria**.

Con los sistemas con 10 procesadores (Figura 5-10) observamos que los tiempos de respuesta son iguales para todas las técnicas. Recordando que los flujos e2e de este sub-estudio poseen 10 actividades localizadas de manera pseudo-aleatoria, este resultado indica que las técnicas basadas en *offsets* no aportan ningún tipo de mejora sobre *holistic* en los sistemas en los que los flujos e2e no repiten procesador. En los tiempos de ejecución asociados observamos que el análisis *offset based w/pr* es ahora el más lento.

Hasta ahora la localización se realizaba de forma pseudo-aleatoria, de manera que en la medida de lo posible los flujos e2e recorrían todos los procesadores al menos en una ocasión. Cuando la localización se realiza de manera totalmente aleatoria se fomenta la aparición de situaciones en las que se repiten los procesadores dentro de un mismo flujo e2e, con la contrapartida de aumentar la probabilidad de dejar algún procesador libre de actividades.

En la Figura 5-11 se muestran las UMP medias obtenidas por *holistic* con localización aleatoria. Puede observarse cómo el análisis *holistic* no se ve influenciado por realizar

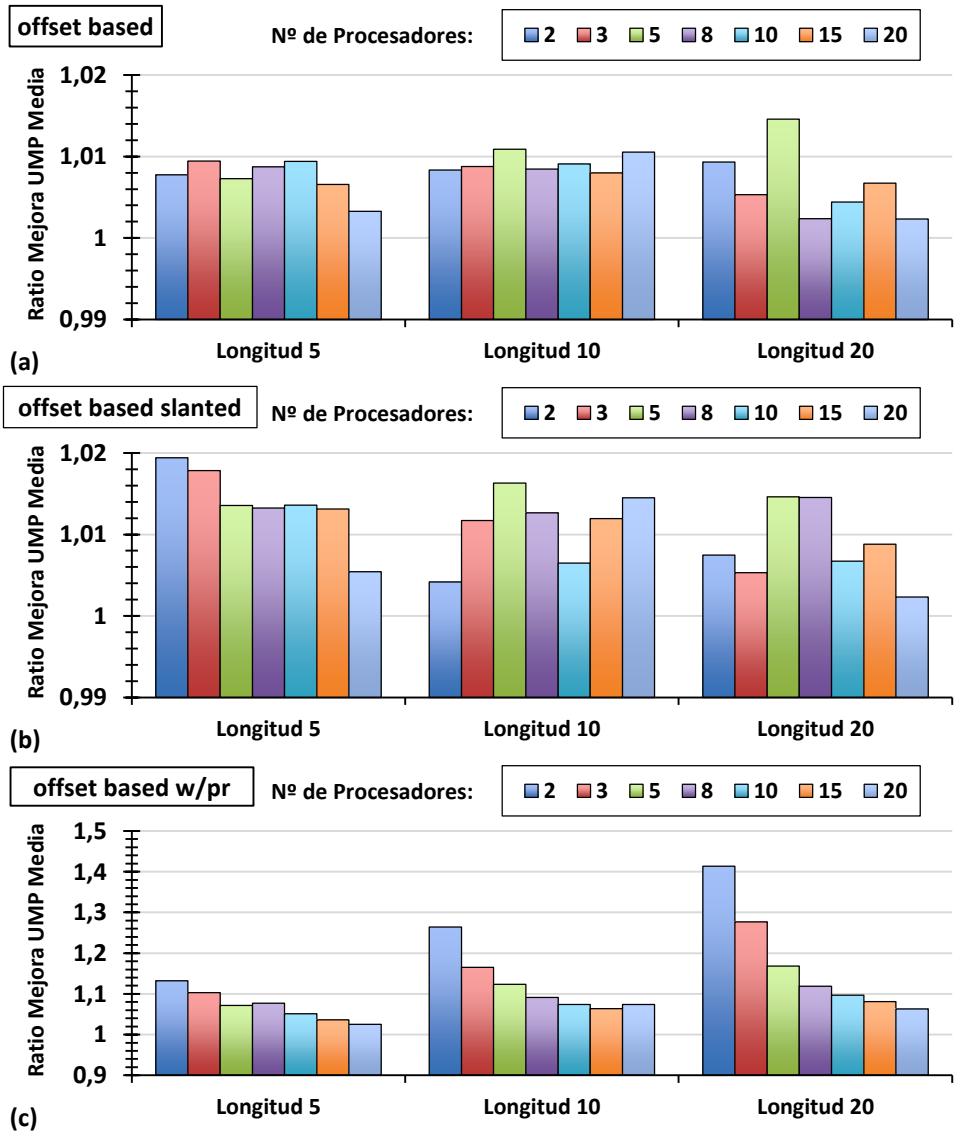


Figura 5-12 Influencia del número de procesadores en las mejoras sobre *holistic,* en términos de UMP medio, de las técnicas de análisis **(a)** ***offset based,*** **(b)** ***offset based slanted*****, (c)** ***offset based w/pr*****,** para flujos e2e de longitud 5, 10 y 20, y **localización de actividades aleatoria**.

una localización aleatoria de actividades, ya que estos resultados son similares a los obtenidos cuando la localización era pseudo-aleatoria.

Sin embargo, y como era de esperar, los análisis basados en *offsets* sí muestran mejoras en su funcionamiento para todo el rango de número de procesadores estudiado, tal y como se ilustra en la Figura 5-12. Esta mejora se debe a que la localización aleatoria fomenta la aparición de situaciones las que los análisis basados en *offsets* funcionan mejor, esto es, situaciones con repetición de procesadores en el flujo e2e.

5.3.2 Influencia de la longitud de los flujos e2e

El estudio anterior de la influencia del número de procesadores se llevó a cabo probando flujos e2e con 3 longitudes distintas, y en los resultados se pudo comprobar cómo los flujos e2e más largos tienen una influencia negativa en la planificabilidad. En esta sección se profundizará en esta influencia de la longitud, probando sistemas con 3, 5, 10, 15 y 20 actividades. El efecto del *jitter* inducido por el aumento de la longitud se modulará planteando dos situaciones distintas: actividades con tiempos de ejecución de mejor caso iguales a 0, y actividades con tiempos de ejecución de mejor caso iguales a los tiempos de ejecución de peor caso. El resto de parámetros se mantienen constantes e iguales a los valores del sistema base.

En la Figura 5-13 se muestran, para esta situación, las UMP medias obtenidas por el análisis *holistic*, y como se puede comprobar el efecto de aumentar la longitud de los flujos e2e incide negativamente en la planificabilidad. Por otra parte, se puede observar cómo la reducción del *jitter* que produce la disponibilidad de los tiempos de ejecución de mejor caso influye, aunque de manera limitada, en los flujos e2e más cortos. Hay que recordar que se está utilizando una técnica trivial de cálculo de los tiempos de respuesta de mejor caso.

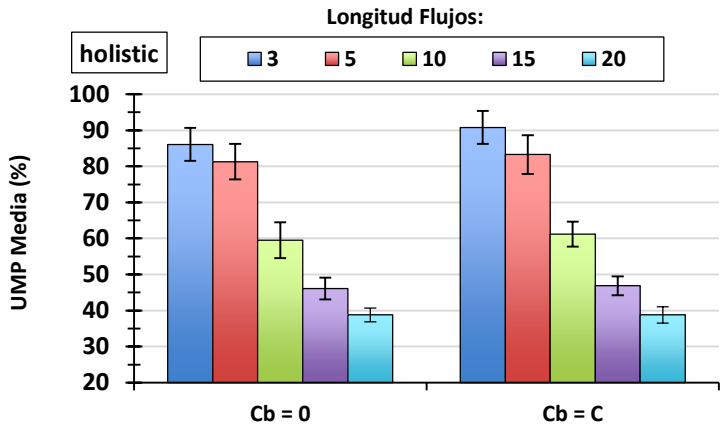


Figura 5-13 Influencia de la longitud de los flujos e2e en la UMP media de sistemas FP analizados con *holistic*, para dos situaciones: tiempos de ejecución de mejor caso iguales al 0%, y al 100% de los tiempos de ejecución de peor caso.

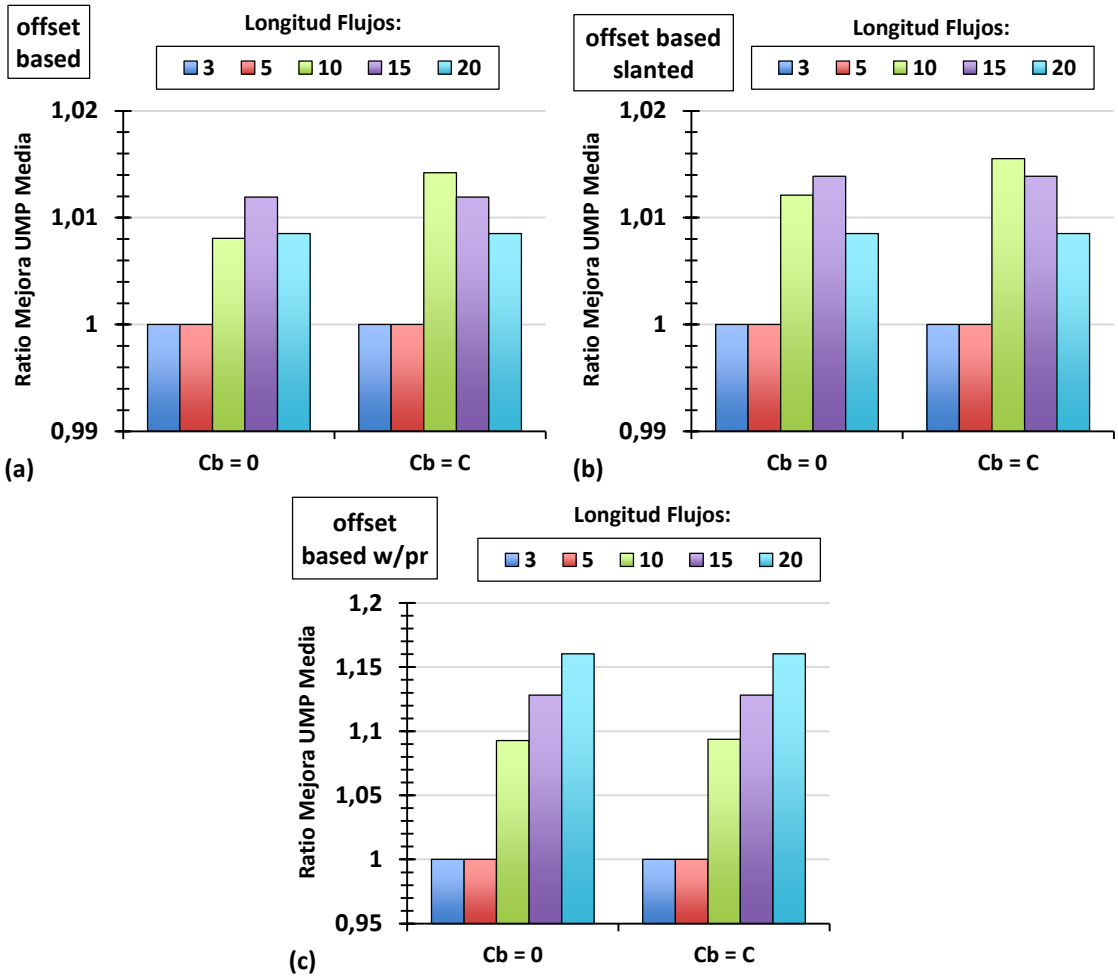


Figura 5-14 Influencia de la longitud de los flujos e2e en las mejoras sobre *holistic,* en términos de UMP media, de las técnicas de análisis **(a)** ***offset based,*** **(b)** ***offset based slanted*****,** **(c)** ***offset based w/pr*****,** para tiempos de ejecución de mejor caso iguales a 0 e iguales a los tiempos de ejecución de peor caso.

En la Figura 5-14 se representan las ratios de mejoras en las UMP medias obtenidas por las técnicas basadas en *offsets* para los mismos sistemas. Tal y como ya se observó al estudiar la influencia del número de procesadores, las mejoras de los análisis basados en *offsets* se manifiestan cuando se repiten procesadores en los flujos. En la figura, teniendo en cuenta que se lleva a cabo una localización pseudo-aleatoria con 5 procesadores, estas situaciones se dan cuando los flujos poseen más de 5 actividades. La mejora observada cuando las actividades poseen tiempos de ejecución de mejor caso no nulos es limitada.

En la Figura 5-15 y en la Figura 5-16 se muestran los promedios de los tiempos de respuesta de peor caso normalizados a los obtenidos por *holistic* para los sistemas con flujos e2e con longitud 15 y 20 respectivamente. Se prueban en cada caso sistemas con tiempos de mejor caso nulos, e iguales a los de peor caso. Comprobamos que aumentar la longitud de los flujos e2e, manteniendo constante el número de procesadores, produce un efecto similar en los tiempos de respuesta al de disminuir el número de procesadores manteniendo la longitud constante. Observamos que la cota de los tiempos de respuesta de peor caso obtenida por *offset based w/pr* mejora al aumentar la longitud de los flujos e2e, esto es, aumentando la repetición de los procesadores. Cuando las actividades poseen tiempos de ejecución de mejor caso iguales a los de mejor caso, se observa que el promedio de los tiempos de respuesta normalizados a *holistic* obtenidos por *offset based*

w/pr aumenta, mientras que la distancia entre *offset based* y *offset based slanted* prácticamente desaparece, tanto para sistemas con 15 como con 20 actividades en los flujos e2e.

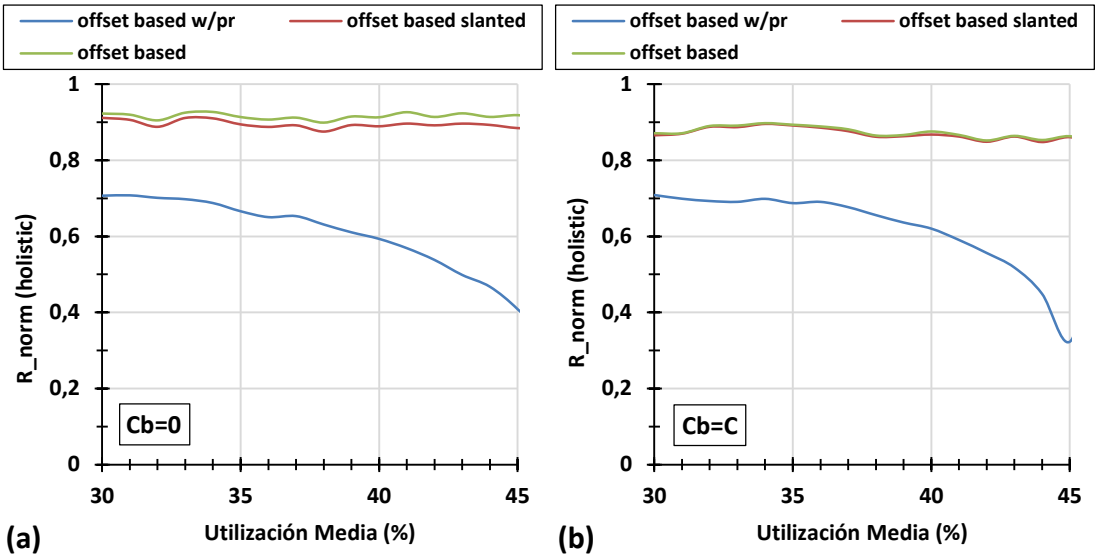


Figura 5-15 Evolución de los tiempos de respuesta medios normalizados a los valores de *holistic*, para sistemas de longitud 15, con **(a) tiempos de ejecución de mejor caso iguales a 0**, y **(b) tiempos de ejecución de mejor caso iguales a los de peor caso.**

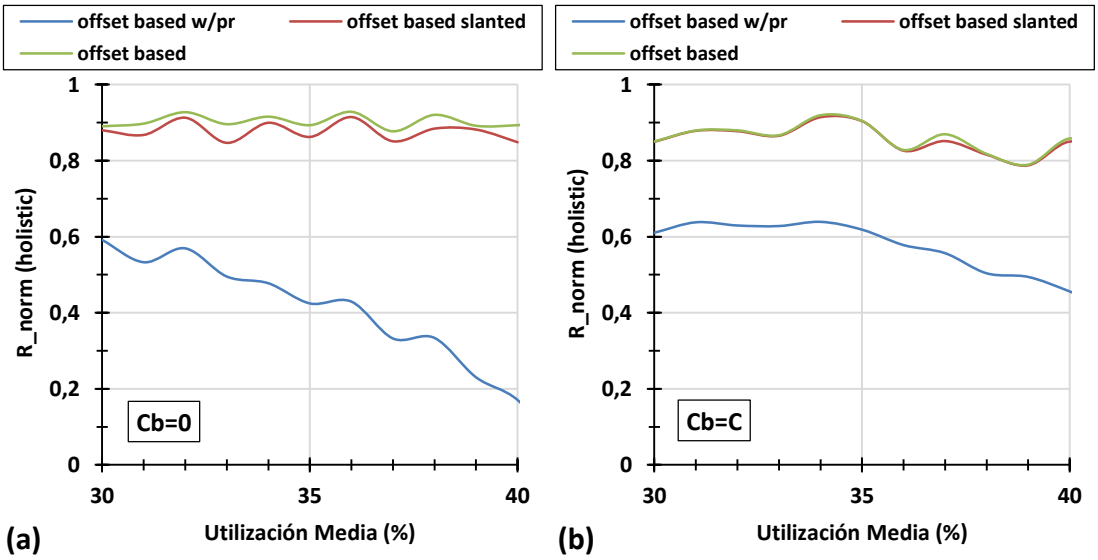


Figura 5-16 Evolución de los tiempos de respuesta medios normalizados a los valores de *holistic*, para sistemas de longitud 20, con **(a) tiempos de ejecución de mejor caso iguales a 0,** y **(b) tiempos de ejecución de mejor caso iguales a los de peor caso.**

5.3.3 Influencia del número de flujos e2e

En esta sección se estudiará la influencia de repartir una misma carga computacional total en distintos números de flujos e2e de longitud constante. Para ello se crean sistemas con 1, 5, 10, 15 y 20 flujos, en tres situaciones distintas: flujos e2e de longitud 5, 10 y 20. El resto de parámetros se mantienen constantes a los valores del sistema base.

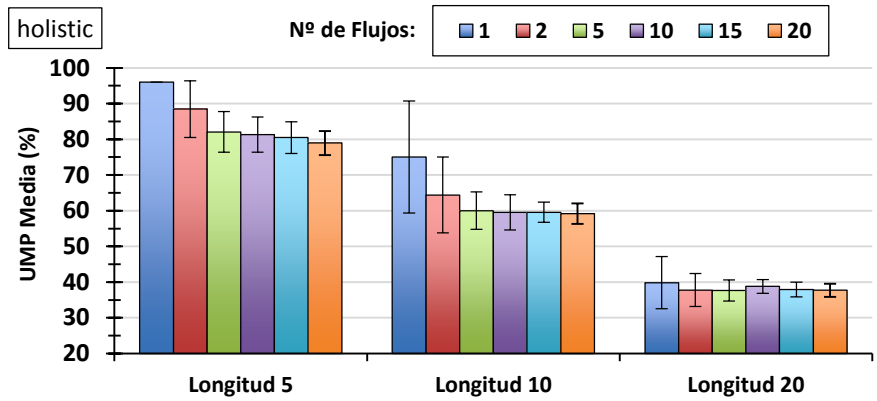


Figura 5-17 Influencia del número de flujos en la UMP media de sistemas FP analizados con *holistic*, para tres situaciones: flujos e2e de longitud 5, 10 y 20.

En la Figura 5-17 se muestran las UMP medias obtenidas con el análisis *holistic*. En primer lugar, y como era de esperar, cuando sólo existe un flujo, y éste tiene una longitud

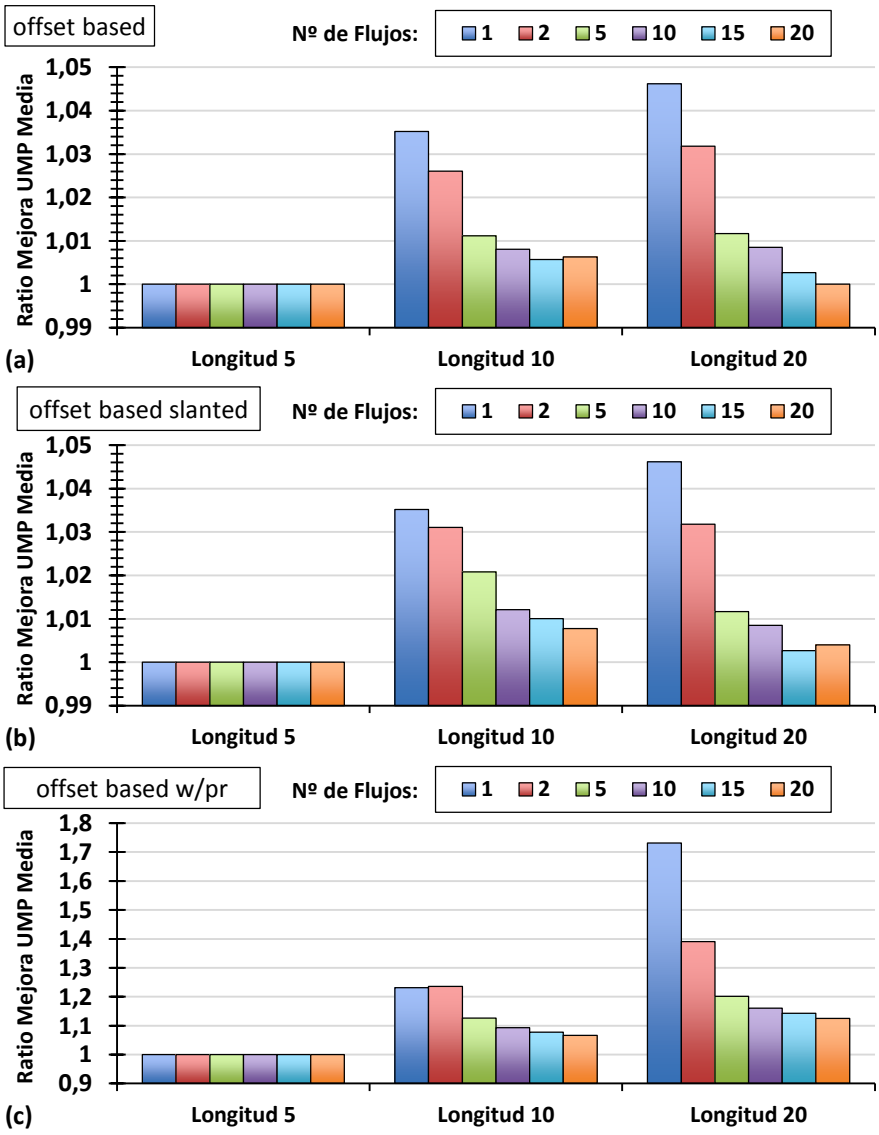


Figura 5-18 Influencia del número de flujos en las mejoras sobre *holistic,* en términos de UMP medio, de las técnicas de análisis **(a)** ***offset based,*** **(b)** ***offset based slanted*****,** **(c)** ***offset based w/pr***, para flujos de longitud 5, 10 y 20.

igual al número de procesadores (5) con localización pseudo-aleatoria (1 actividad en total por procesador), se alcanzan utilizaciones planificables del 96%, que es la utilización mayor generada. Aumentar la longitud de este sistema con un único flujo e2e provoca un gradual deterioro de la planificabilidad. Cabe notar la elevada desviación típica del caso de un flujo e2e de longitud 10 y 20. Esto es un indicativo de que la planificabilidad de un flujo e2e solitario es altamente dependiente de la forma de localizar sus actividades. Adicionalmente, en la figura puede observarse cómo el número de flujos e2e no induce una gran influencia en los resultados del análisis *holistic*, una vez que los sistemas tienen más de un flujo e2e.

En la Figura 5-18 se representan las ratios de las UMP medias obtenidas por las técnicas basadas en *offsets* con respecto a *holistic*. Se puede apreciar que las mejoras introducidas por los análisis basados en *offsets* son más apreciables cuando los sistemas poseen un menor número de flujos. Adicionalmente, se vuelve a comprobar la menor cota en los tiempos de respuesta de peor caso obtenidos por *offset based w/pr*.

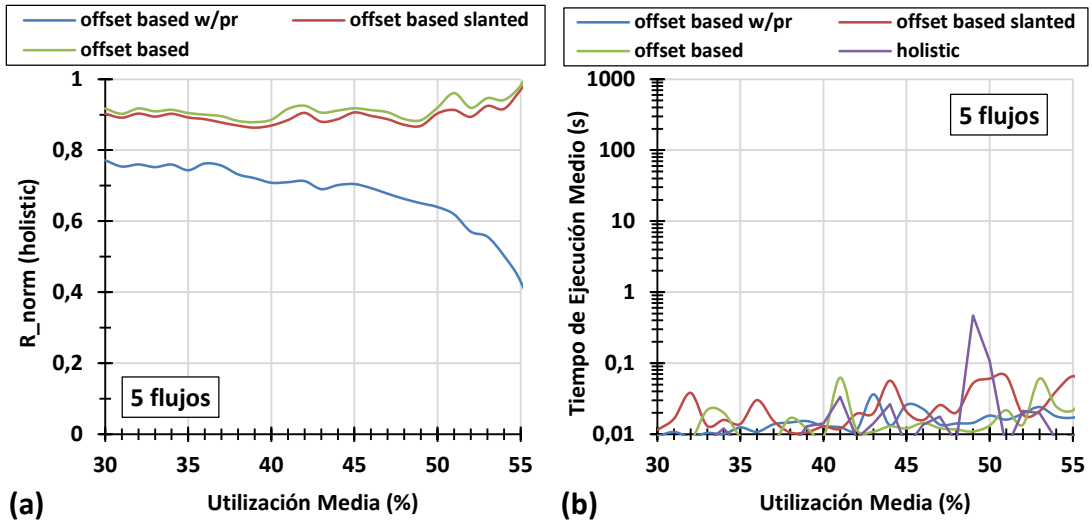


Figura 5-19 Evolución de los **(a) tiempos de respuesta medios normalizados a los valores de *holistic*, y (b) los tiempos de ejecución**, para sistemas con 5 flujos.

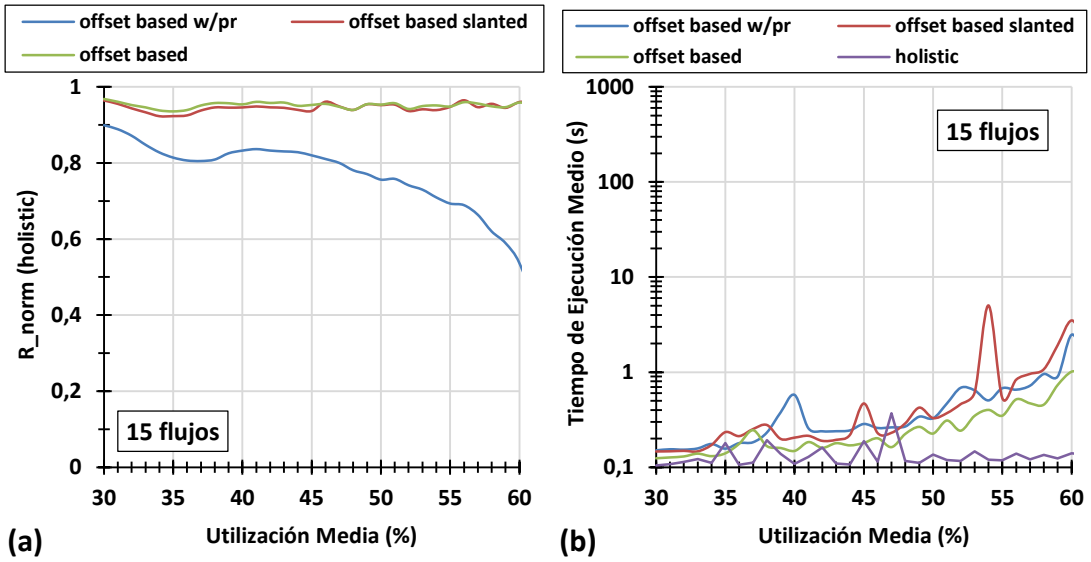


Figura 5-20 Evolución de los (a) tiempos de respuesta medios normalizados a los valores de *holistic*, y (b) los tiempos de ejecución, para sistemas con 15 flujos e2e.

Las conclusiones sobre la Figura 5-18 se pueden replicar cuando observamos los promedios de los tiempos de respuesta de peor caso normalizados a *holistic* obtenidos para 5 flujos e2e (Figura 5-19(a)) y 15 flujos e2e (Figura 5-20(a)): los tiempos de respuesta se distancian en mayor medida de *holistic* cuando los sistemas poseen un menor número de flujos e2e. Esta conclusión es válida para todo el rango de utilizaciones estudiado. Los tiempos de ejecución del análisis asociados (Figura 5-19(b) y Figura 5-20 (b)) indican que añadir más flujos aumenta la complejidad del análisis, hecho esperable porque se aumenta el número total de actividades en el sistema, y por lo tanto, también aumenta el número de casos a tratar en el análisis.

5.3.4 Influencia del rango de los periodos

A continuación estudiamos cómo afecta la variación en el rango de los periodos del sistema a las técnicas de análisis de planificabilidad. Para ello generamos sistemas en los que los periodos se seleccionan en rangos con ratios de 1, 10, 100 y 1000. Puesto que el periodo se le asigna a los flujos e2e, para estudiar el efecto de la amplitud del rango de periodos, se crean sistemas con diferentes números de flujos e2e: 1, 10 y 20. El caso de sistemas con un solo flujo e2e, y por lo tanto un solo periodo asignado, se incluye como referencia.

También se estudiará el efecto de utilizar diferentes distribuciones de probabilidad para seleccionar los periodos, mediante las dos distribuciones disponibles en GEN4MAST: *Log-Uniform* y *Custom-Uniform*. El resto de características se mantienen iguales a los valores del sistema base.

En la Figura 5-21 se muestran las UMP medias obtenidas por el algoritmo *holistic* para ambas funciones de probabilidad. Como cabría esperar, cuando los sistemas poseen un único flujo e2e, ni el rango de los periodos ni la función de probabilidad muestran una influencia destacable. Cuando los sistemas tienen varios flujos e2e, aumentar el rango de los periodos tiene un efecto beneficioso en la planificación en términos de la UMP media. El efecto de ampliar el rango de los periodos se hace más apreciable cuando se utiliza una función de probabilidad *Log-Uniform*, dando lugar también a UMP medias más altas.

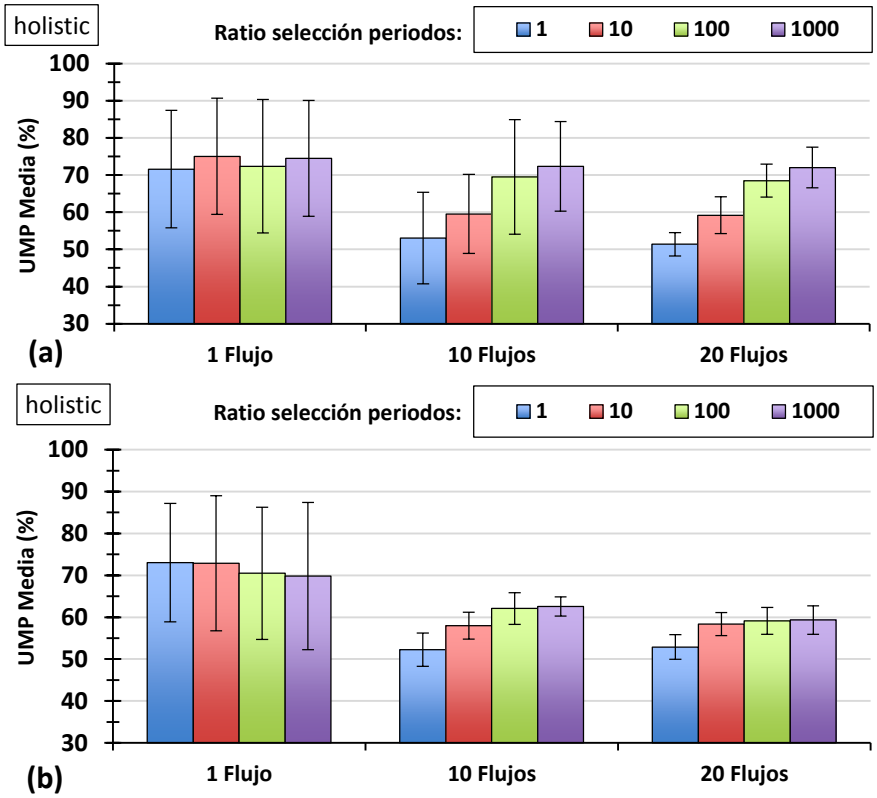


Figura 5-21 Influencia de la ratio de selección de los periodos en la UMP media de sistemas FP analizados con *holistic* para tres situaciones: sistemas con 1, 10 y 20 flujos e2e, y con selección de periodos de tipo **(a) *Log-Uniform* y (b) *Custom-Uniform***

En la Figura 5-22 se presentan las ratios de mejoras de la UMP media obtenidas sobre *holistic* para las técnicas basadas en *offsets*, y para los distintos rangos de periodos estudiados. En primer lugar se comprueba que para sistemas con 10 flujos e2e, la ventaja de las técnicas *offset based* y *offset based slanted* aumenta a medida que los periodos poseen mayor rango. Para sistemas con 20 flujos e2e esta tendencia se repite, salvo cuando el rango de los periodos es el mayor de los generados (1000). En estos casos *offset based* y *offset based slanted* obtienen en promedio los mismos resultados que *holistic*. Por otra parte, la ratio de mejora sobre *holistic* en la UMP media de la técnica *offset based w/pr* se mantiene prácticamente constante para los diferentes rangos de los periodos, cuando los sistemas poseen 10 y 20 flujos e2e.

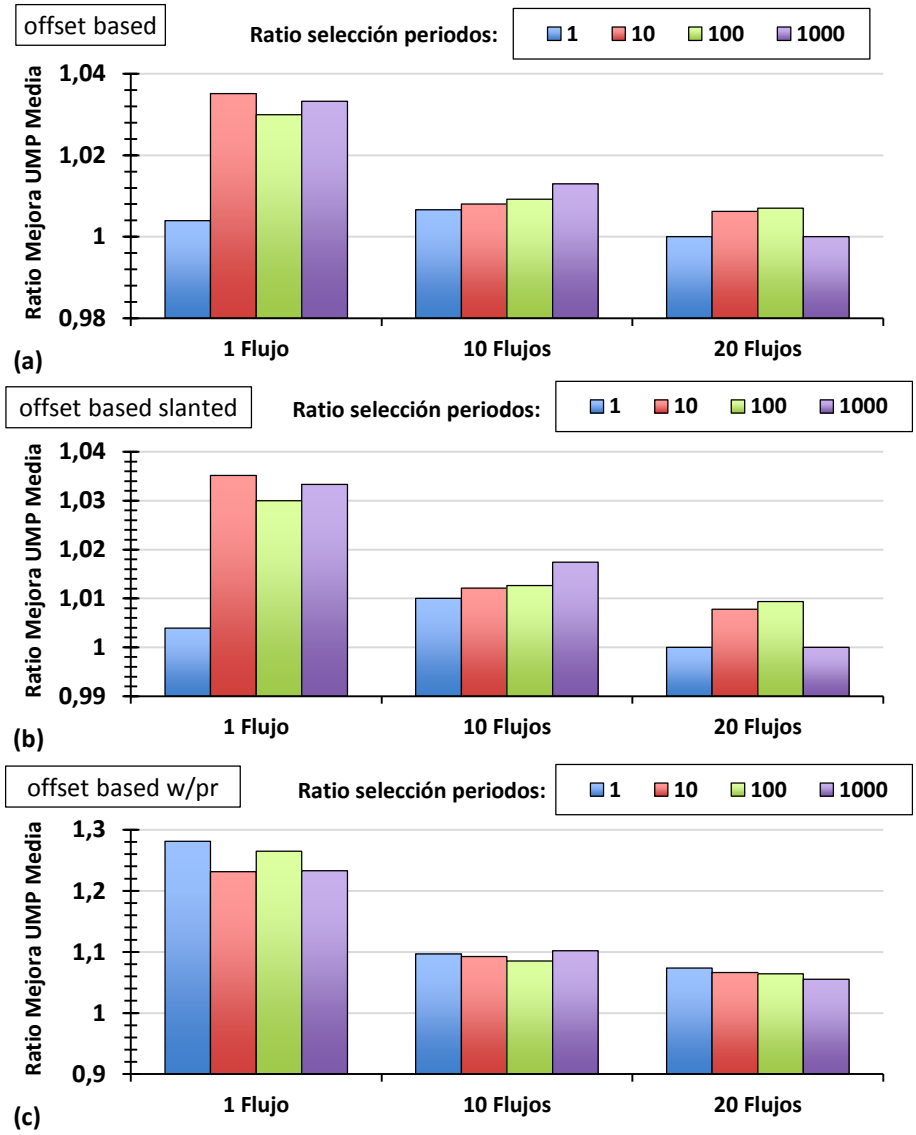


Figura 5-22 Influencia de la ratio de selección de los periodos en los ratios de mejoras sobre *holistic,* en términos de UMP medio, de las técnicas de análisis **(a)** ***offset based,*** **(b)** ***offset based slanted*****, (c)** ***offset based w/pr****,* para selección de periodos de ***Log-Uniform***

En la Figura 5-23(a) y en la Figura 5-24(a) se muestran los promedios de los tiempos de respuesta de peor caso normalizados a *holistic* para ratios de periodos de 1 y 1000 respectivamente. Los análisis *offset based* y *offset based slanted* mantienen su distancia con respecto a *holistic* de manera más o menos constante independientemente de la utilización o de las ratios de los periodos. Sin embargo, para *offset based w/pr* observamos que aumentar la ratio de los periodos provoca que sus tiempos de respuesta de peor caso se asemejen más a los obtenidos por el resto de técnicas.

En cuanto a los tiempos requeridos para el análisis (Figura 5-23(b) y Figura 5-24(b)), observamos que para una misma utilización en el sistema, las mismas técnicas requieren más tiempo cuando el rango de los periodos es mayor.

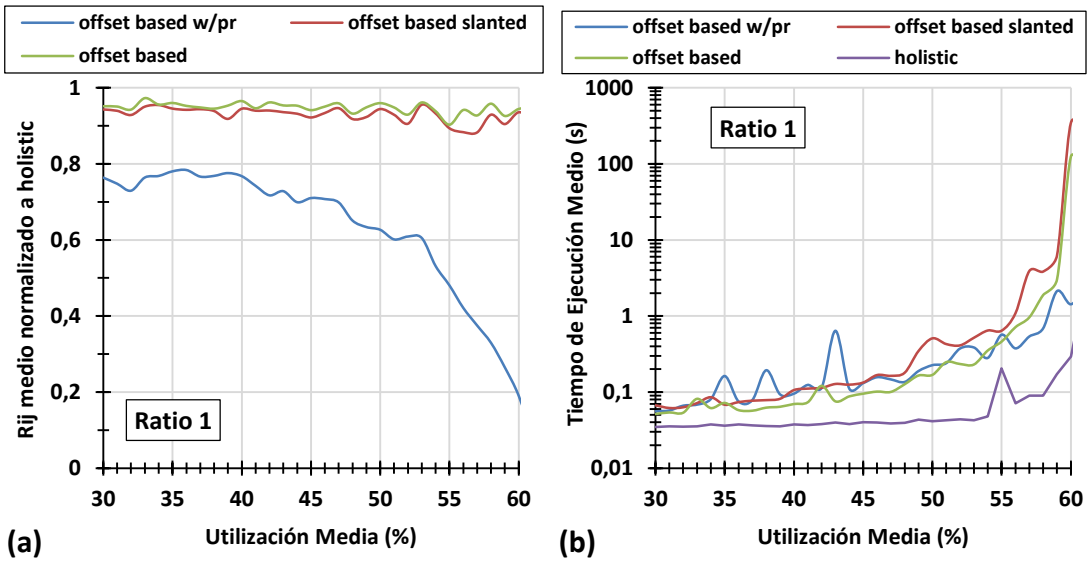


Figura 5-23 Evolución de **los (a) tiempos de respuesta medios normalizados a los valores de *holistic*, y (b) los tiempos de ejecución**, para los sistemas con ratio 1 en la selección de periodos, para sistemas con 10 flujos e2e.

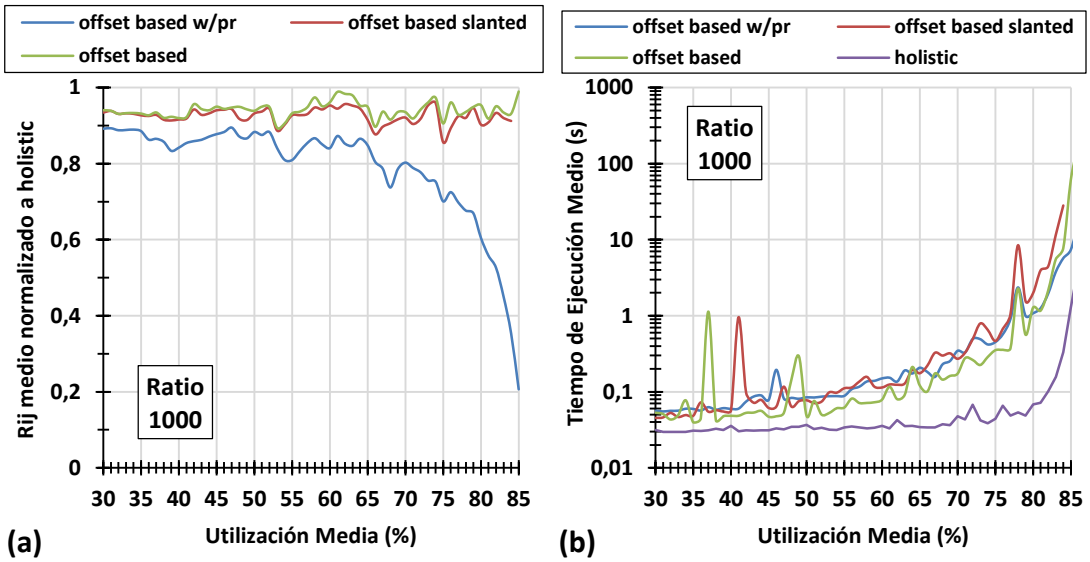


Figura 5-24 Evolución de **los (a) tiempos de respuesta medios normalizados a los valores de *holistic*, y (b) los tiempos de ejecución**, para los sistemas con ratio 1000 en la selección de periodos, para sistemas con 10 flujos e2e.

5.3.5 Análisis basado en *offsets* de fuerza bruta

El análisis basado en *offsets* de fuerza bruta (*offset based brute force*) lleva a cabo un análisis exacto probando todos los posibles instantes críticos que se pueden crear por todas las combinaciones posibles de actividades en el sistema, quedándose con la variación que produce el mayor de los tiempos de respuesta de peor caso calculados de entre todas las combinaciones. Este análisis es computacionalmente no tratable salvo para sistemas pequeños. Por este motivo, llevamos a cabo su estudio de forma separada sobre un conjunto de sistemas simplificado tanto en extensión como en la complejidad del mismo.

El conjunto de sistemas a tratar se describe según los parámetros de la Tabla 5-6. Nos centramos en las variaciones del número de procesadores, puesto que ya hemos comprobado que éste es un parámetro que nos permite observar las diferencias entre las diferentes técnicas. El resto de características son las mismas que las del sistema base del estudio general.

Nº de Flujos e2e	Nº Actividades por Flujo e2e	Nº Procesadores	Ratio Periodos
5	10	10,8,6,4	100

Tabla 5-6 Características del dominio de estudio simplificado para *offset based brute force*.

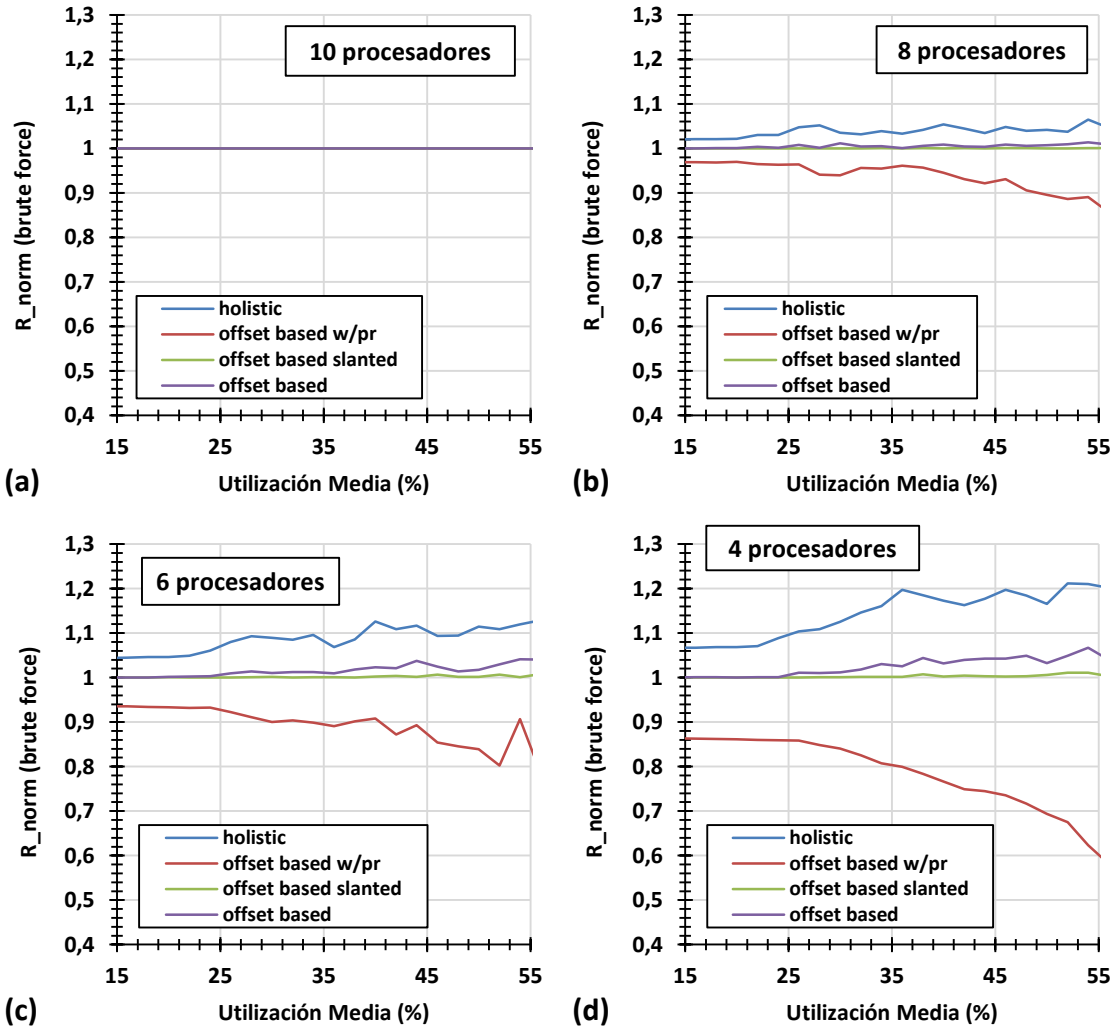


Figura 5-25 Tiempos de respuesta de peor caso normalizados a los obtenidos por *offset based brute force*, para sistemas con **(a) 10, (b) 8, (b) 6 y (d) 4 procesadores**.

En la Figura 5-25 se representan los promedios de los tiempos de respuesta de peor caso de las diferentes técnicas de análisis, normalizados a los obtenidos por *offset based brute force* (*R_norm*(*offset based brute force*)), para los diferentes números de procesadores desde 10 hasta 4. Cabe hacer notar que un valor normalizado menor a 1 indica tiempos de respuesta de peor caso en promedio inferiores a los obtenidos por *offset based brute force*. Para sistemas con 10 procesadores (Figura 5-25(a)), en los que no se producen repeticiones de procesadores en los flujos e2e, observamos que todas las técnicas obtienen los mismos tiempos de respuesta de peor caso. A medida que disminuye

el número de procesadores las diferencias de rendimiento van aumentando, y podemos comprobar que las optimizaciones con relaciones de precedencia del análisis *offset based w/pr* tienen una eficacia que claramente superan al análisis *offset based brute force*. Hay que recordar que esta última técnica carece de estas optimizaciones. Adicionalmente, el *offset based slanted* tiene un rendimiento muy similar al análisis de fuerza bruta.

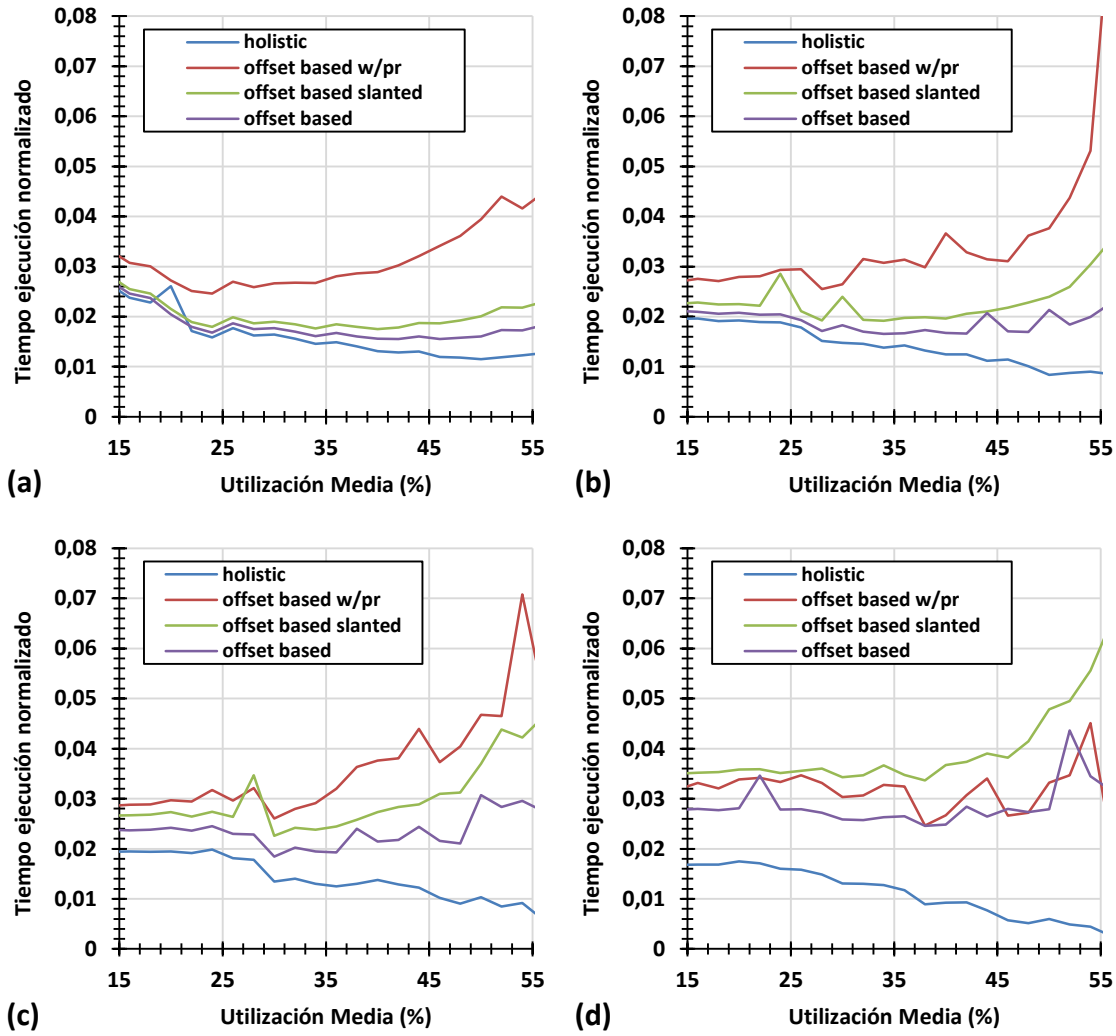


Figura 5-26 Tiempos de análisis normalizados a los requeridos por *offset based brute force*, para sistemas con **(a) 10, (b) 8, (b) 6 y (d) 4 procesadores**.

Completamos el estudio con los tiempos de análisis normalizados a los necesitados por *offset based brute force* (Figura 5-26) para los diferentes números de procesadores. En estos resultados se confirma la complejidad del algoritmo de fuerza bruta, con tiempos de análisis claramente superiores. Observamos que los tiempos de análisis del resto de técnicas son siempre al menos 10 veces inferiores (ratio de 0.1) que los obtenidos por el análisis de fuerza bruta en el rango de utilizaciones estudiado.

Con este estudio simplificado creado *ad-hoc* para evaluar el rendimiento del análisis de fuerza bruta podemos concluir que los beneficios de calcular de forma exacta la combinación de actividades que crean el instante crítico al considerar las actividades independientes se pueden ver claramente mejorados por la técnica que utiliza las relaciones de precedencia (*offset based w/pr*).

5.4 Planificación GC-EDF

Para estudiar las técnicas de análisis en GC-EDF se repite el proceso seguido para FP en la sección anterior, con la salvedad de que para GC-EDF sólo hay disponibles dos técnicas de análisis: la técnica holística (*holistic*), y una única basada en *offsets* (*offset based*).

5.4.1 Influencia del número de recursos procesadores

Para estudiar el efecto de la variación del número de procesadores, repetimos el proceso seguido para llevar a cabo el mismo estudio en sistemas FP, con el mismo conjunto de sistemas.

En la Figura 5-27 se muestran las UMP medias obtenidas por el análisis *holistic*, en el que se observa que independientemente de la longitud de los flujos e2e, o número de procesadores, utilizar GC-EDF como política de planificación en sistemas con plazos $D_i=N_i*T_i$ permite planificar cargas de al menos el 96%, que es el valor de utilización máximo generado. En este caso, el uso de las UMP medias no permite observar las diferencias en el comportamiento de las técnicas de análisis para GC-EDF. Por esta razón, la comparación se realizará utilizando los promedios de los tiempos de respuesta de peor caso, que constituyen los resultados principales de las técnicas de análisis.

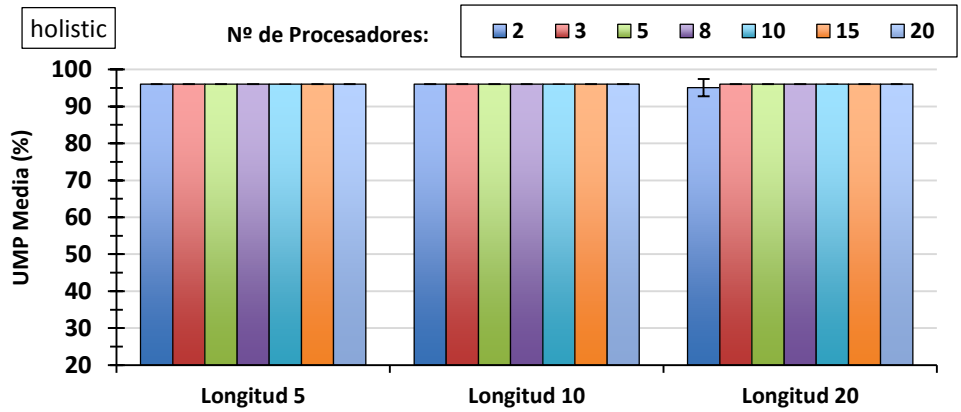


Figura 5-27 Influencia del número de procesadores en la UMP media de sistemas GC-EDF analizados con *holistic*, para flujos de longitud 5, 10 y 20, y **localización de actividades pseudo-aleatoria**.

En la Figura 5-28 se muestran los promedios de los tiempos de respuesta de peor caso normalizados a *holistic* (*R_norm*(*holistic*)), para flujos e2e de longitud 10 y localización pseudo-aleatoria de las actividades. Al igual que lo que ocurría con FP, la cota de los tiempos de respuesta de peor caso del análisis basado en *offsets* en comparación a *holistic* disminuye a medida que los sistemas tienen menos procesadores, y por consiguiente, éstos se repiten con mayor frecuencia en los flujos e2e. La mayor diferencia con *holistic* se observa en utilizaciones medias en torno al 60%, situación en la que, para sistemas con dos procesadores, los tiempos de respuesta obtenidos por *offset based* son aproximadamente 0.83 veces los obtenidos por *holistic*. Como era de esperar, cuando la localización se realiza de manera totalmente aleatoria (Figura 5-29), la técnica *offset*

based obtiene menores cotas de los tiempos de respuesta de peor caso con respecto a *holistic,* para todo el rango de número de procesadores estudiado.

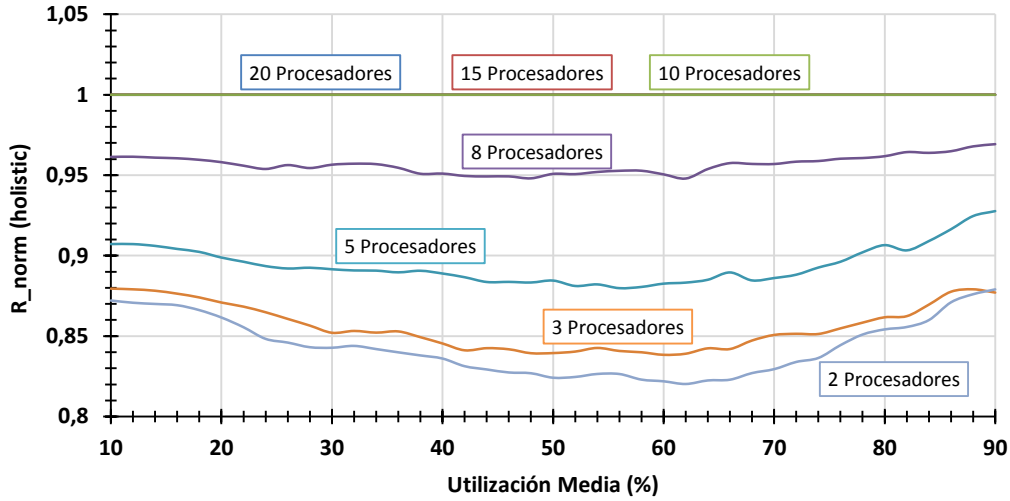


Figura 5-28 Evolución de los tiempos de respuesta medios obtenidos por *offset based* normalizados a los valores de *holistic* en sistemas GC-EDF, para distintos números de procesadores, flujos e2e de **longitud 10**, y **localización pseudo-aleatoria** de actividades

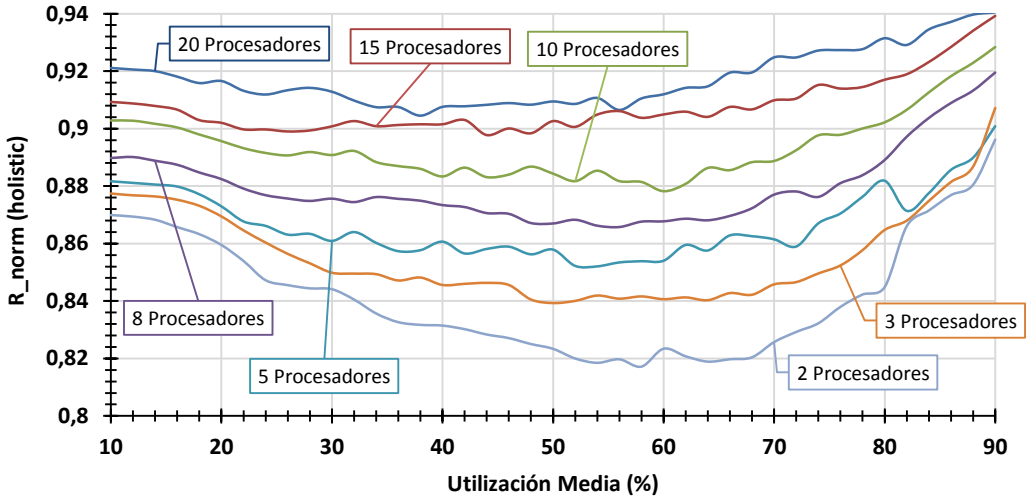


Figura 5-29 Evolución de los tiempos de respuesta medios obtenidos por *offset based* normalizados a los valores de *holistic* en sistemas GC-EDF, para distintos números de procesadores, flujos e2e de **longitud 10**, y **localización aleatoria** de actividades

Para completar el estudio, en la Figura 5-30 se reproducen los resultados para sistemas con flujos e2e de longitud 20 y localización pseudo-aleatoria. En esta figura se observa que al aumentar el número de actividades en los flujos e2e, la cota superior de los tiempos de respuesta de peor caso ofrecidos por *offset based* se distancia aún más de *holistic*. Para sistemas con 2 o 3 procesadores, los tiempos de respuesta de peor caso obtenidos por *offset based* se sitúan en torno a 0.8 veces los obtenidos por *holistic* para utilizaciones medias del 45%.

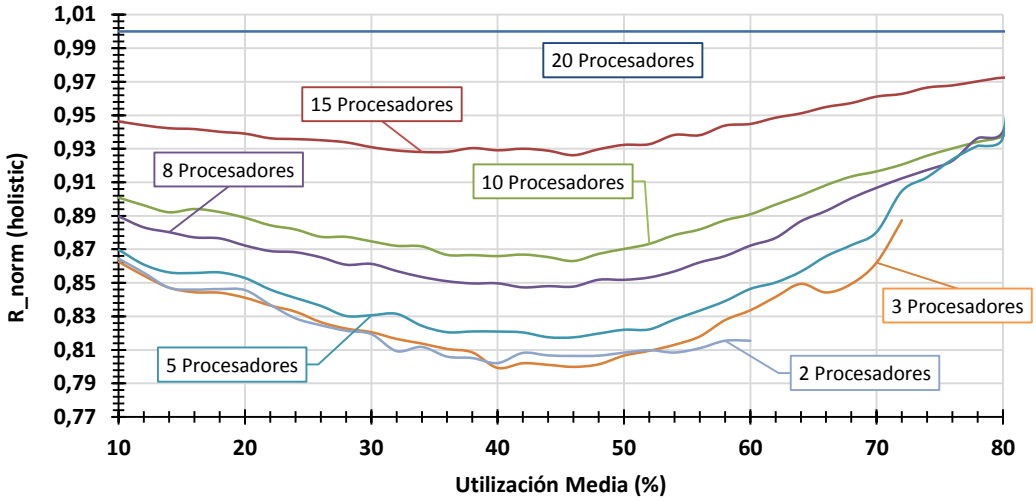


Figura 5-30 Evolución de los tiempos de respuesta medios obtenidos por *offset based* normalizados a los valores de *Holistic* en sistemas GC-EDF, para distintos números de procesadores, flujos e2e de **longitud 20**, y **localización pseudo-aleatoria** de actividades

5.4.2 Influencia de la longitud de los flujos e2e

Ya hemos determinado que en GC-EDF, al igual que en FP, aumentar la repetición de los procesadores en los flujos hace que los análisis basados en *offsets* obtengan cotas más bajas de los tiempos de respuesta de peor caso con respecto a *holistic*. Previamente habíamos inducido este efecto modificando el número de procesadores en el sistema. Ahora, modificando la longitud de los flujos e2e, manteniendo el número de procesadores fijo en un valor de 5, y con tiempos de ejecución de mejor caso nulos (Figura 5-31), observamos que la cota de los tiempos de respuesta de peor caso ofrecidos por *offset based* con respecto a *holistic* disminuye a medida que los flujos e2e son más largos. Éste resultado es comparable a lo observado previamente con flujos e2e de longitud fija 10, variando el número de procesadores.

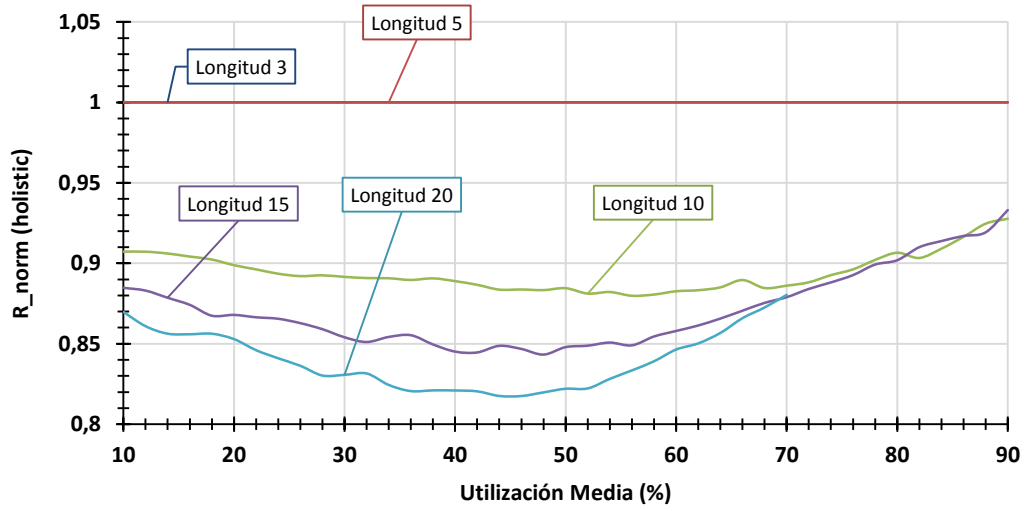


Figura 5-31 Evolución de los tiempos de respuesta medios obtenidos por *offset based* normalizados a los valores de *holistic* en sistemas GC-EDF, para distintos longitudes de los flujos e2e, y **tiempos de ejecución de mejor caso nulos.**

Cuando los tiempos de ejecución de mejor caso son iguales a los de peor caso (Figura 5-32), la técnica *offset based* obtiene cotas de los tiempos de respuesta de peor caso aún más distanciadas de las ofrecidas por *holistic*. Para flujos e2e con longitud 20, los tiempos de respuesta de peor caso calculados por *offset based* son en promedio 0.8 veces los calculados por *holistic*, para utilizaciones en torno al 50%.

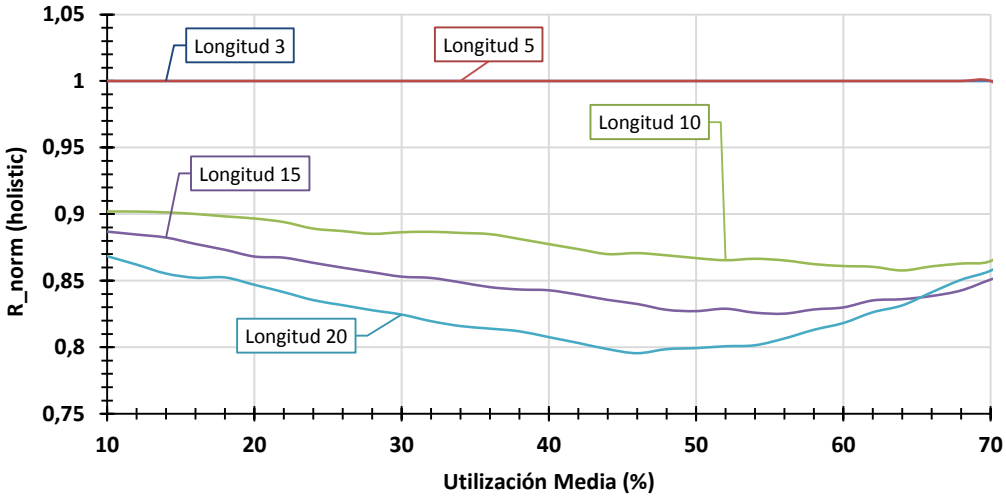


Figura 5-32 Evolución de los tiempos de respuesta medios obtenidos por *offset based* normalizados a los valores de *holistic* en sistemas GC-EDF, para distintos longitudes de los flujos e2e, y **tiempos de ejecución de mejor iguales a los de peor caso.**

5.4.3 Influencia del número de flujos e2e

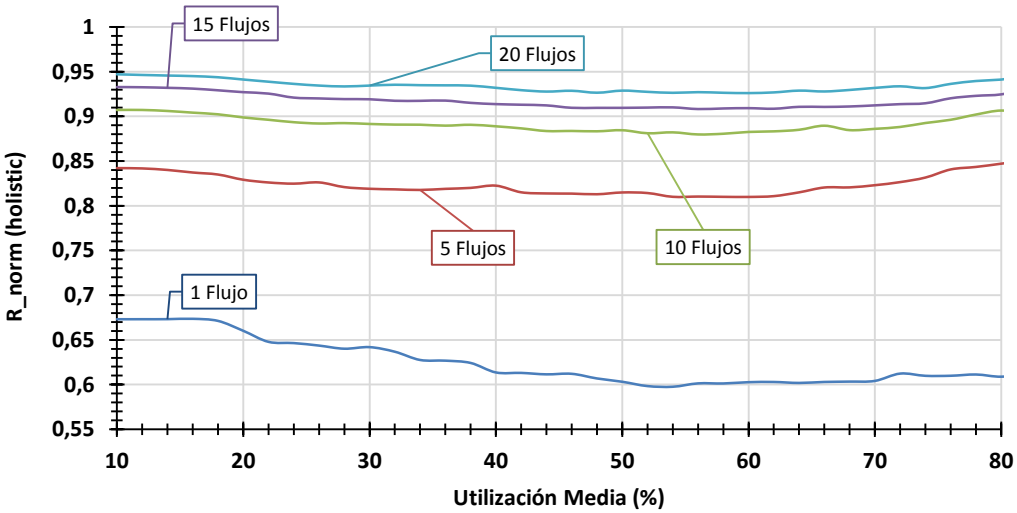


Figura 5-33 Evolución de los tiempos de respuesta medios obtenidos por o*ffset based* normalizados a los valores de *holistic* en sistemas GC-EDF, para distintos números de flujos e2e, siempre con **longitud 10**.

En el estudio de sistemas FP se comprobó que los beneficios de aplicar una técnica basada en *offsets* resultaban más evidentes cuando los sistemas poseían menos flujos. Como se observa en la Figura 5-33 para sistemas con distinto número de flujos e2e y siempre con longitud 10, este comportamiento se repite en sistemas GC-EDF. Observamos que, en comparación con *holistic*, la cota de los tiempos de respuesta de

peor caso calculados por *offset based* disminuyen a medida que los sistemas poseen menor número de flujos e2e. Para sistemas con utilizaciones medias del 60%, punto en el que se observan las mayores diferencias, y para sistemas con 10 flujos e2e los tiempos de respuesta de peor caso de *offset based* son en promedio 0.87 veces los calculados por *holistic*. Si se disminuye el número de flujos a 5 y 1, la diferencia aumenta a una ratio de 0.82 y 0.6 respectivamente.

Cuando los sistemas poseen flujos e2e de longitud 20 (Figura 5-34), se observa la misma tendencia observada para flujos e2e con longitud 10, pero con un mayor distanciamiento en las cotas de los tiempos de respuesta de peor caso ofrecidos por *offset based*. Para sistemas con 1 flujo e2e, los tiempos de respuesta calculados por *offset based* son en promedio 0.3 veces los calculados por *holistic*, para utilizaciones medias en torno al 70%.

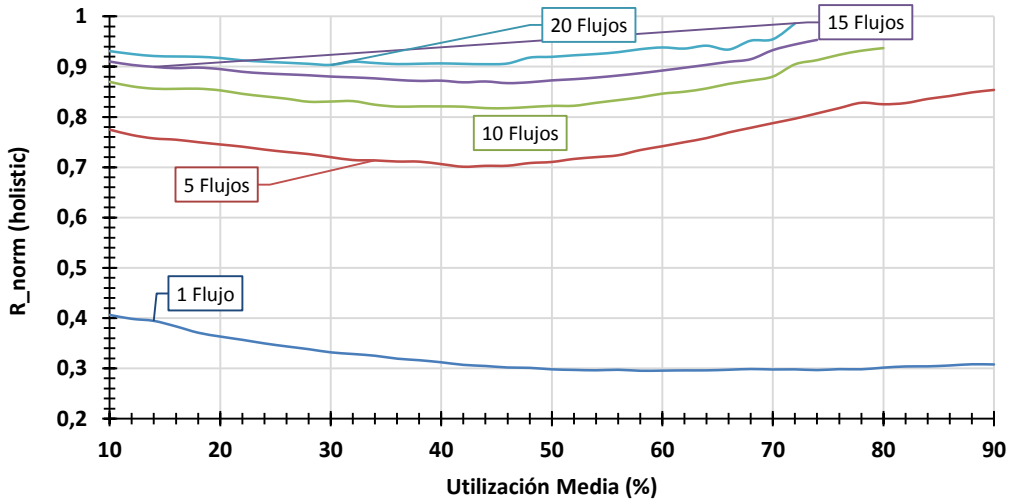
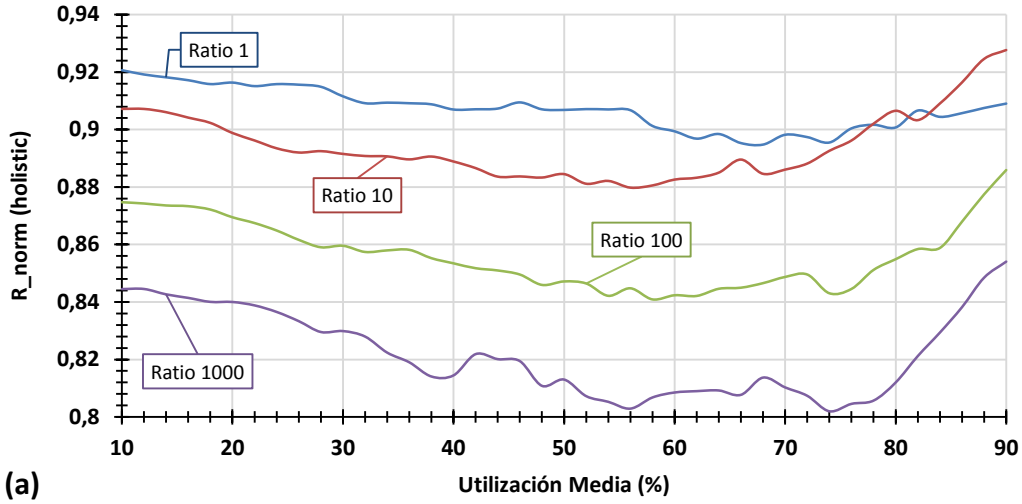


Figura 5-34 Evolución de los tiempos de respuesta medios obtenidos por *offset based* normalizados a los valores de *holistic* en sistemas GC-EDF, para distintos números de flujos, y flujos e2e de **longitud 20**.

5.4.4 Influencia del rango de los periodos

A continuación estudiamos el efecto de los diferentes rangos de periodos en las técnicas de análisis de GC-EDF. Para sistemas con 10 flujos (Figura 5-35) observamos que *offset based* se distancia de *holistic* a medida que los periodos aumentan su rango. Las diferencias son más pronunciadas utilizando una selección de periodos *Log-Uniform*, observando que para una ratio de periodos de 1000, los tiempos de respuesta de peor caso calculados por *offset based* son hasta 0.8 veces los calculados por *holistic*.

Si aumentamos el número de flujos e2e a 20 (Figura 5-36), comprobamos cómo la distancia en el promedio de los tiempos de respuesta obtenidos entre *offset based* y *holistic* disminuye en comparación a los obtenidos con sistemas con 10 flujos e2e.



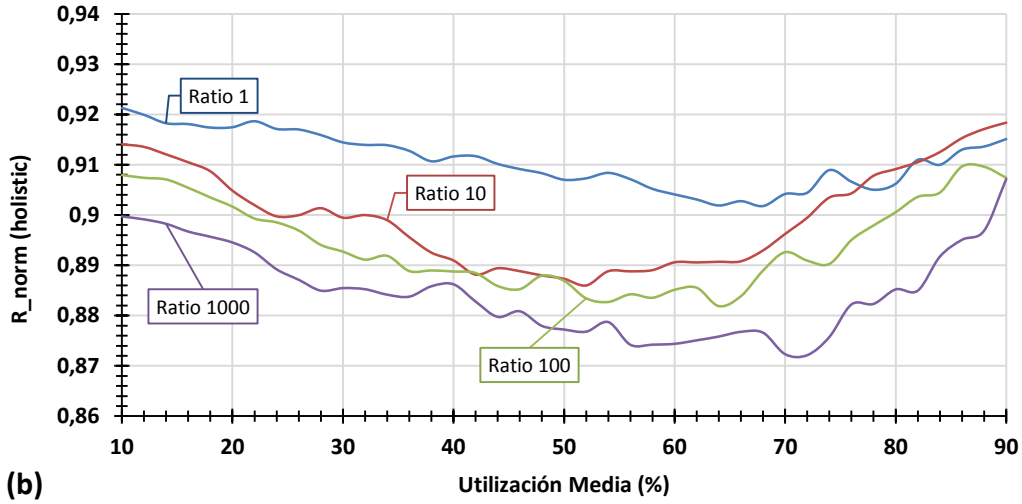


Figura 5-35 Evolución de los tiempos de respuesta medios obtenidos por *offset based* normalizados a los valores de *holistic* en sistemas GC-EDF, para distintos ratios de selección de periodos, sistemas de 10 flujos e2e, y utilizando selección **(a) *Log-Uniform* y (b) *Custom-Uniform*.**

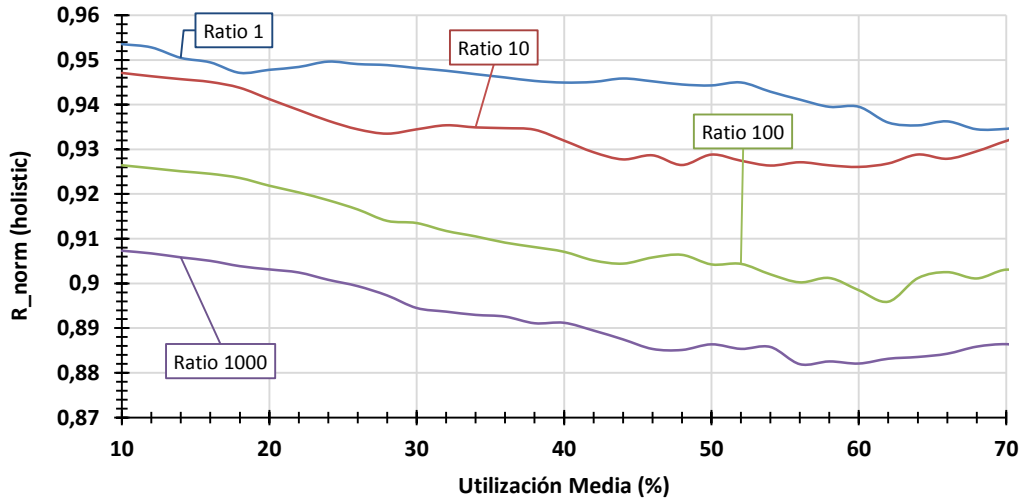


Figura 5-36 Evolución de los tiempos de respuesta medios obtenidos por *offset based* normalizados a los valores de *holistic* en sistemas GC-EDF, para distintos ratios de selección de periodos, sistemas de 20 flujos e2e, y utilizando selección ***Log-Uniform***

5.5 Planificación LC-EDF

Repetimos los mismos cuatro sub-estudios llevados a cabo para FP y GC-EDF, pero en esta ocasión para LC-EDF. Una particularidad importante que nos presenta LC-EDF es que sólo tenemos disponible una técnica de análisis de planificabilidad, *holistic*. Por ello, el presente estudio para LC-EDF se incluye para completar los realizados anteriormente para FP y GC-EDF. En el Capítulo 6 de esta tesis se verá cómo estos resultados para LC-EDF se ven influenciados negativamente por la técnica de asignación de plazos de planificación utilizada (PD).

5.5.1 Influencia del número de recursos procesadores

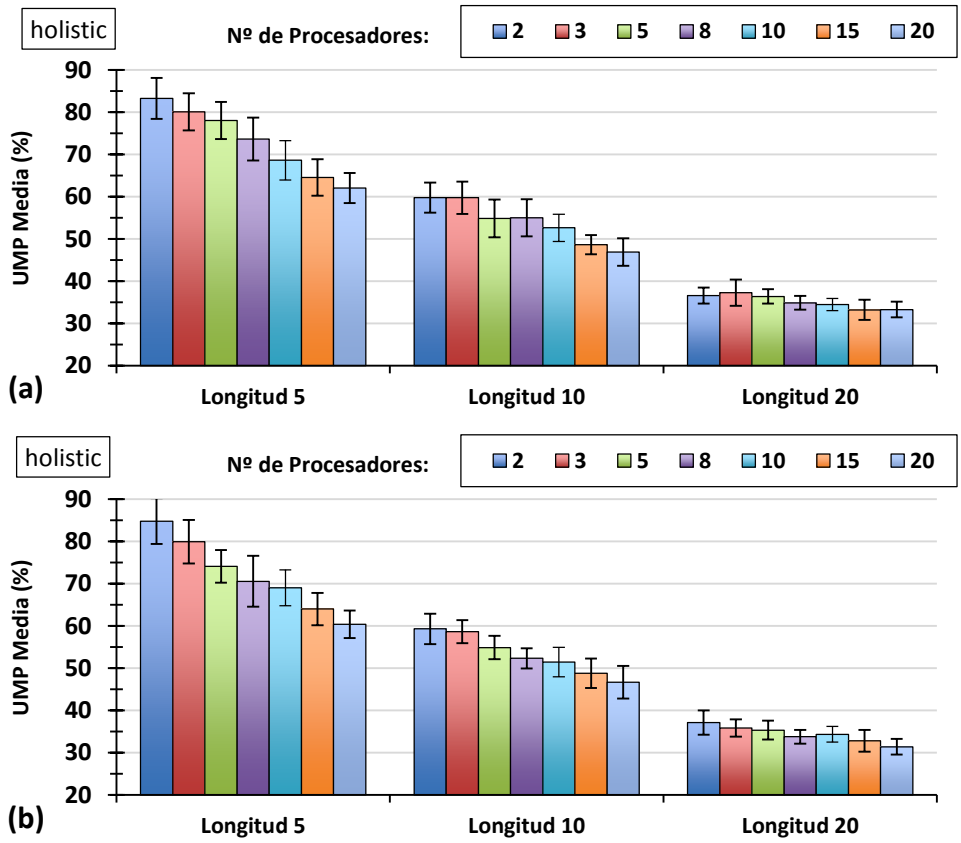


Figura 5-37 Influencia del número de procesadores en la UMP media de sistemas LC-EDF analizados con *holistic*, para flujos e2e de longitud 5, 10 y 20, con **(a) localización de actividades pseudo-aleatoria, y (b) localización de actividades aleatoria**

En la Figura 5-37 se representa la evolución de la UMP media de *holistic* para las variaciones del número de procesadores y los dos métodos de localización bajo estudio (pseudo-aleatoria, y aleatoria). Observamos que, al contrario de lo que sucede con FP, en LC-EDF sí se observa una influencia del número de procesadores en el análisis *holistic*, aumentando la planificabilidad a medida que los sistemas poseen menos procesadores. Este comportamiento, aunque se asemeje al observado para los análisis basados en *offsets*, es independiente de las repeticiones de los procesadores en los flujos, ya que se observa en mayor medida en los sistemas con flujos e2e cortos con localización pseudo-

aleatoria, en los que no se produce ninguna repetición de procesador. Adicionalmente, no se observan claras diferencias entre ambos métodos de localización de las actividades.

5.5.2 Influencia de la longitud de los flujos e2e

Aumentar la longitud de los flujos e2e incrementa la propagación del *jitter* y su consiguiente efecto negativo en la planificabilidad. En los sistemas FP y GC-EDF estudiados previamente este efecto negativo se manifestó claramente, y es también aparente en sistemas LC-EDF (Figura 5-38). Adicionalmente, disponer de los tiempos de ejecución de mejor caso (iguales a los de peor caso en este estudio) produce un efecto limitado a la hora de reducir el *jitter*, y el beneficio sólo es aparente para sistemas con flujos e2e más cortos, comportamiento parejo al observado para FP y GC-EDF.

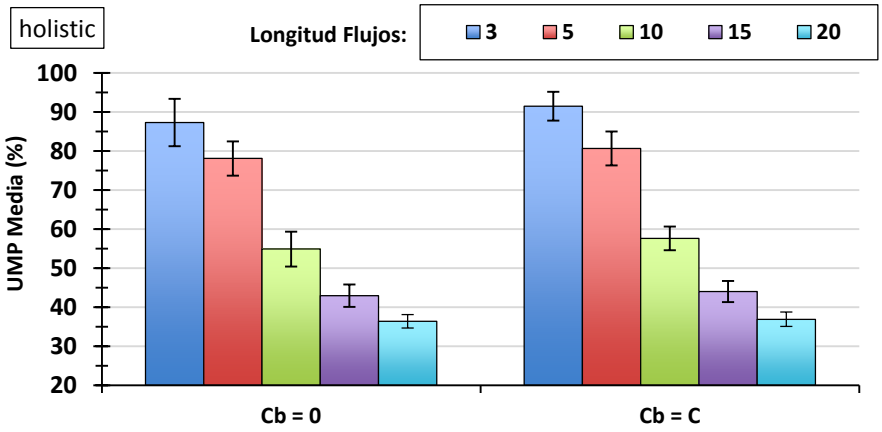


Figura 5-38 Influencia de la longitud de los flujos e2e en la UMP media de sistemas LC-EDF analizados con *holistic*, para dos situaciones: tiempos de ejecución de mejor caso iguales al 0%, y al 100% de los tiempos de ejecución de peor caso.

5.5.3 Influencia del número de flujos e2e

En el estudio de la influencia del número de flujos e2e en LC-EDF (Figura 5-39) observamos que la planificabilidad aumenta ligeramente cuando los sistemas poseen más flujos. Si aumentamos el número de flujos e2e, y por lo tanto también el número total de actividades, para conseguir la misma utilización en el sistema cada actividad tendrá unos tiempos de ejecución de peor caso inferiores. El resultado de esta figura indica que existe una cierta tendencia en el análisis *holistic* de LC-EDF a obtener mejores resultados cuando las actividades están atomizadas y son más cortas.

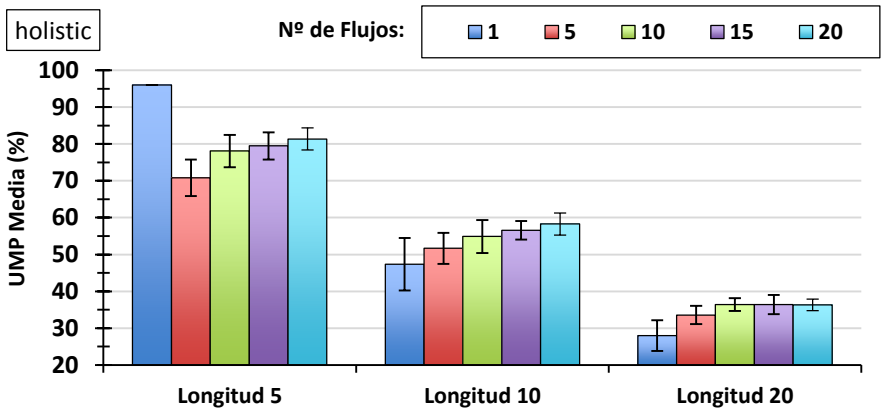


Figura 5-39 Influencia del número de flujos e2e en la UMP media de sistemas LC-EDF analizados con *holistic*, para tres situaciones: flujos de longitud 5, 10 y 20.

5.5.4 Influencia del rango de los periodos

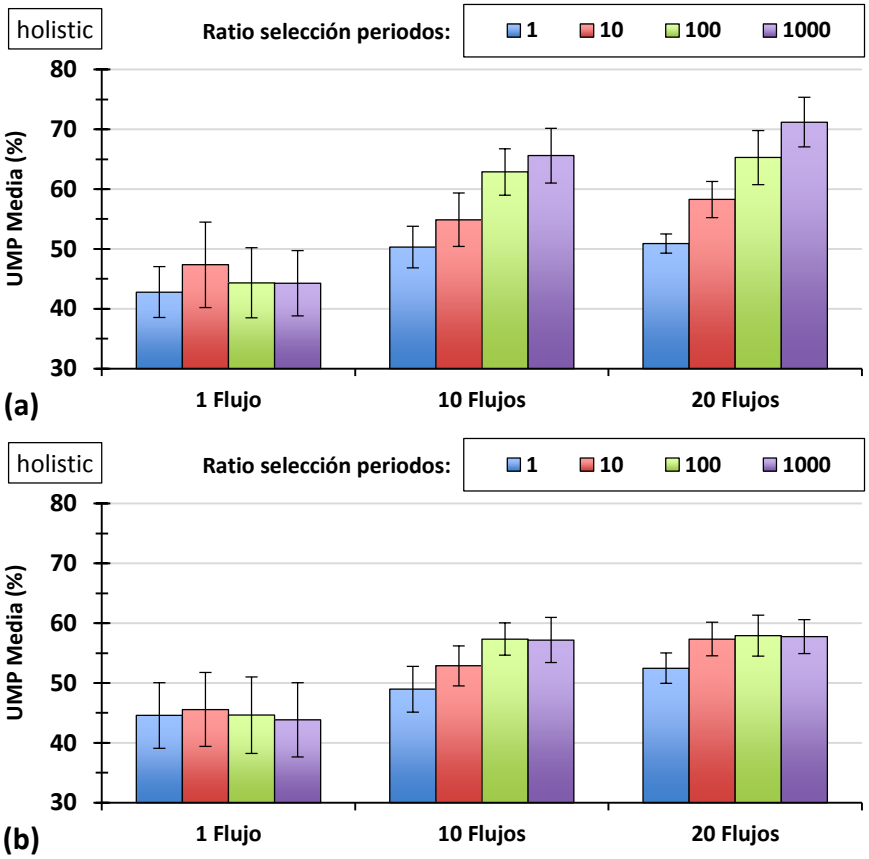


Figura 5-40 Influencia de la ratio de selección de los periodos en la UMP media de sistemas LC-EDF analizados con *holistic* para tres situaciones: sistemas con 1, 10 y 20 flujos e2e, y con selección de periodos de tipo **(a) Logarítmica-Uniforme (*Log-Uniform*) y (b) Uniforme (*Uniform*)**

En la Figura 5-40 se muestran los resultados obtenidos por *holistic* en LC-EDF para distintos rangos de los periodos. Estos resultados son similares a los observados para las políticas de planificación estudiadas previamente, es decir, a medida que se incrementa la dispersión de los periodos, aumenta la planificabilidad de los sistemas. La dispersión de

los periodos se consigue tanto ampliando el rango de los mismos, como utilizando una distribución *Log-Uniform*.

5.6 Conclusiones

En el presente capítulo nos planteamos un estudio comparativo de técnicas de análisis de planificabilidad de sistemas distribuidos. Para una mayor eficiencia en la ejecución, se llevó a cabo utilizando la herramienta GEN4MAST, ejecutada en el *cluster* de computación Tres Mares.

El capítulo comenzó con la definición del conjunto de sistemas que formaron parte del estudio (Sección 5.1). El grupo de sistemas se seleccionó balanceando dos requisitos opuestos: la necesidad de tener un conjunto de ejemplos lo suficientemente extenso para obtener resultados representativos, pero a la vez que pudieran ser analizados en tiempos totales razonables. Se seleccionó un conjunto de sistemas que pivotaban sobre un sistema base con 10 flujos e2e, 10 actividades por flujo e2e, y 5 recursos procesadores. Sobre este sistema base se variaron cuatro características principales, y sobre cada variación se realizó un sub-estudio diferente: el número de recursos procesadores, la longitud de los flujo e2e, el número de flujos e2e, y el rango de los periodos.

En total, considerando los cuatro sub-estudios y sus correspondientes variaciones, se generaron alrededor de un millón de sistemas, lo que se traduce en aproximadamente 86 *gigabytes* de ficheros descriptores de sistemas. El tiempo total de CPU requerido para ejecutar este estudio se situó en alrededor de 300 días. Al repartir esta ejecución en 300 procesadores del *cluster* Tres Mares, el tiempo real empleado fue de algo menos de dos días. Hay que tener en cuenta que los trabajos enviados al *cluster* requieren diferentes tiempos de ejecución total, por lo que no se consigue tener a los 300 procesadores trabajando durante todo el transcurso de la ejecución.

En sistemas FP se comprobó que la técnica de análisis basada en *offsets* con relaciones de precedencia (*offset based w/pr*) es la que en términos generales obtiene los mejores resultados, esto es, obtiene las cotas superiores de los tiempos de respuesta de peor caso más ajustadas. Las técnicas basadas en *offsets* sólo obtienen mejores resultados que la técnica *holistic* cuando los flujos e2e repiten algún recurso procesador. Esta ventaja se acentúa con sistemas con menor número de flujos e2e. La técnica basada en *offsets slanted* obtiene tiempos de respuesta de peor caso ligeramente inferiores a los obtenidos por la técnica basada en *offsets*. Por último, la técnica de análisis basada en *offsets* de fuerza bruta requiere tiempos de ejecución dos órdenes de magnitud superiores a los utilizados por el resto de técnicas, para obtener tiempos de respuesta de peor caso similares a los obtenidos por *offset based slanted*.

En sistemas GC-EDF se comparan dos técnicas de análisis, el análisis holístico, y el basado en *offsets*. En primer lugar se comprobó que para plazos $D_i=N_i*Ti$, las utilizaciones máximas planificables son del 96%, que es la máxima utilización generada. Este comportamiento es similar al que se consigue con planificación EDF y actividades periódicas independientes con plazos iguales a los periodos en sistemas monoprocesadores. Adicionalmente, el análisis basado en *offsets* sólo obtiene tiempos de respuesta de peor caso inferiores a los obtenidos por el análisis holístico cuando los flujos e2e repiten algún recurso procesador, y ésta diferencia es mayor cuando los sistemas poseen menor número de flujos e2e. Para sistemas con un solo flujo e2e, los tiempos de

respuesta de peor caso obtenidos por el análisis basado en *offsets* son hasta 0,6 veces los obtenidos por el análisis holístico, siendo esta la mayor diferencia entre ambas técnicas para los sistemas evaluados.

Para sistemas LC-EDF sólo se dispone del análisis holístico. Con esta técnica, el rendimiento de LC-EDF en términos de la UMP obtenida se asemeja al de FP en los sistemas con flujos e2e más cortos y menor número de procesadores. En el resto de situaciones, el rendimiento observado es inferior a FP, y marcadamente inferior a GC-EDF. En los próximos capítulos se observará que este pobre rendimiento de LC-EDF proviene en gran parte de las técnicas de asignación de parámetros de planificación existentes en la actualidad, y se aportarán soluciones que intenten disminuir estas diferencias de rendimiento entre políticas reportadas.

6 Estudio Comparativo de Técnicas de Asignación de Parámetros de Planificación en Sistemas Distribuidos

En el capítulo anterior se llevó a cabo un estudio comparativo de las técnicas de análisis de planificabilidad para sistemas distribuidos disponibles en MAST, en el que se comprobó el rendimiento obtenido por cada una de ellas para una amplia variedad de especificaciones de sistemas. Igual importancia en sistemas distribuidos que las técnicas de análisis de planificabilidad, la tienen las técnicas de asignación de parámetros de planificación, cuya labor es la de optimizar la ejecución de las actividades de forma que el conjunto del sistema cumpla siempre sus plazos. Esta optimización se realiza asignando a todas las actividades del sistema un parámetro de planificación que en tiempo de ejecución será utilizado por el planificador para decidir qué actividad de entre todas las activas debe ejecutarse en cada momento. En MAST, y por extensión en este trabajo, las asignaciones de parámetros de planificación se realizan *off-line* y de forma estática, esto es, los parámetros de planificación son valores fijos que se calculan antes de poner en funcionamiento el sistema. En el presente capítulo estudiaremos el rendimiento de las técnicas de asignación de parámetros de planificación para sistemas distribuidos, que fueron introducidas en el Capítulo 2.

La motivación de llevar a cabo este tipo de estudio es similar al que originó el estudio de las técnicas de análisis del capítulo anterior: no existe ningún estudio en el que se comparen, bajo las mismas circunstancias y ante una amplia variedad de tipos de sistemas, los distintos algoritmos de asignación de parámetros de planificación disponibles para sistemas distribuidos. De hecho, algunos de los algoritmos que aquí van a estudiarse (EQS y EQF) han sido originalmente propuestos para su aplicación *on-line*, y

hasta donde sabemos, nunca se han aplicado para asignar parámetros de planificación *off-line* en sistemas distribuidos de tiempo real estricto.

6.1 Dominio del estudio

El dominio del estudio se plantea con la misma premisa que el presentado en el Capítulo 5 para la comparativa de técnicas de análisis de planificabilidad, esto es, abarcar el mayor conjunto posible de tipos de sistemas, con complejidades que sean a la vez lo suficientemente altas para ser representativas de los sistemas reales, pero también resolubles en tiempos razonables. Puesto que la premisa es la misma, estructuramos el dominio de manera similar al capítulo previo: definimos un sistema base a partir del cual generamos nuevos sistemas variando alguna de sus características cada vez. Las características del sistema base son las mismas que las utilizadas en el capítulo anterior, y se resumen a continuación:

- Número de flujos e2e: 10.
- Número de actividades para todos los flujos e2e (longitud): 10
- Número de recursos procesadores: 5.
- Localización de las actividades: Si es posible, no se repite un recurso procesador en el flujo (pseudo-aleatoria).
- Plazos de principio a fin: $D_i=N_i*T_i$
- Periodos: Se seleccionan aleatoriamente en el rango [100, 1000] (Ratio=10), mediante una distribución de probabilidad logarítmico-uniforme (*Log-Uniform*)
- Utilización de las actividades calculadas mediante el algoritmo *UUnifast*.
- Tiempos de ejecución de mejor caso de las actividades igual a 0.
- Todos los procesadores poseen la misma utilización en todo momento.
- Se generan series de utilizaciones en el rango [10, 96] (en %).

Para formar el dominio generamos sistemas variando una serie de características del sistema base que clasificamos en dos categorías: características primarias, y secundarias. Las características primarias son aquellas que definen la morfología básica del sistema: el número de flujos e2e y sus longitudes, número de procesadores, y el rango de los periodos. En el Capítulo 5 ya se estudió la influencia que estas variaciones tienen en las técnicas de análisis de planificabilidad, y en base a los resultados obtenidos, ahora nos centraremos en un subconjunto representativo de esas variaciones. Las características secundarias representan diferentes situaciones especiales a las que puede enfrentarse el algoritmo de asignación de parámetros de planificación.

La mayoría de los algoritmos de asignación de parámetros de planificación presentados se basan en la misma idea fundamental: repartir el plazo de principio a fin entre las actividades que componen cada flujo e2e utilizando algún criterio. Es por ello que en las variaciones de características primarias nos centraremos en los plazos de principio a fin y en el número de actividades por flujo e2e, generando a partir del sistema base sistemas con los siguientes valores para dichas características:

- Plazos de principio a fin: ratio D_i / T_i = {1, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20}. Teniendo en cuenta que el sistema base que toman como referencia posee 10 actividades por flujo (N_i=10), este rango equivale a plazos de principio a fin entre

$D_i=T_i$ y $D_i=2*N_i*T_i$. Estas variaciones se generan con el parámetro de GEN4MAST "DEADLINES 1 2 3 4 6 8 10 12 14 16 18 20".

- Número de actividades por flujo e2e: {4, 6, 8, 10, 12, 14, 16, 18, 20}. Al conjunto de sistemas con una longitud concreta de entre estos valores lo llamaremos L-X, donde X es el número de actividades en los flujos e2e. Por ejemplo, el grupo L-4 representa el conjunto de sistemas generados con 4 actividades por flujo e2e. Es importante hacer notar que al estar basados en el sistema base, que posee plazos $D_i=N_i*T_i$, los plazos de principio a fin de los flujos e2e se escalan a las diferentes longitudes generadas. Estas variaciones se generan con el parámetro de GEN4MAST "N_STEPS 4 6 8 10 12 14 16 18 20".

Para el resto de características primarias creamos 3 grupos de ejemplos, cada uno formado por sistemas en los que, tomando el sistema base, se modifica el valor de una de sus características básicas. Estos grupos son los siguientes:

- P-10: Sistemas con 10 procesadores (parámetro de GEN4MAST "N_PROCESSORS 10").
- F-5 : Sistemas con 5 flujos (parámetro de GEN4MAST "N_FLOWS 5").
- R-1000 : Sistemas con periodos seleccionados en un rango cuyo ratio es 1000 (parámetro de GEN4MAST "PERIOD_RATIOS 1000").

Los valores con los que se generan estos grupos se resumen en la Tabla 6-1:

	Base	**P-10**	**F-5**	**R-1000**
Número de Procesadores	**5**	**10**	5	5
Número de flujos e2e	**10**	10	**5**	10
Ratio de los periodos	**10**	10	10	**1000**

Tabla 6-1 Características de los sistemas de los grupos P-10, F-5 y R-1000.

A partir del sistema base se van a generar otros grupos de sistemas modificando sus características secundarias. En este estudio se plantean 5 tipos de estas situaciones, con los siguientes nombres y descripciones:

- L-Rand : En el sistema base todos los flujos e2e tienen longitud 10. Mantener fija la longitud de todos los flujos del sistema resulta útil para poder estudiar el efecto de la longitud de los flujos en la planificación, pero no representa una situación que se encuentre con normalidad en las aplicaciones reales. Por esta razón creamos el grupo L-Rand en el que la longitud de los flujos se elige aleatoriamente en el rango [2,10]. Este grupo se genera con el parámetro de GEN4MAST FIXED_LENGTH a valor False.
- S-50 : Es habitual encontrarse con sistemas de tiempo real en los que flujos distribuidos conviven con actividades independientes dedicadas a llevar a cabo por ejemplo tareas de control local o de interfaz de usuario. Para modelar estas situaciones creamos el grupo S-50 en el que el 50% de los flujos e2e tiene longitud 1. Se genera con el parámetro de GEN4MAST SINGLE_FLOWS con valor 50.
- UP-Rand : En el sistema base todos los procesadores tienen la misma utilización. En UP-Rand cada recurso puede tener distinta carga para cumplir con la misma

utilización total buscada. Se genera con el parámetro de GEN4MAST UNIFORM_UTILIZATION a valor False.

- B-100 : En el sistema base todas las actividades tienen un tiempo de ejecución de mejor caso igual a 0. En el grupo B-100 los tiempos de ejecución de mejor caso se hacen iguales a los de peor caso ($Cb_{ij} = C_{ij}$). Se genera con el parámetro de GEN4MAST BEST_CASE con valor 100 (porcentaje sobre el tiempo de ejecución de peor caso).
- SCALE : Los sistemas base tienen una carga generada utilizando el método *UUnifast*. El grupo SCALE se genera a partir de los sistemas base, pero con cargas generadas con el algoritmo *SCALE-WCET*, en el que todas las actividades tienen la misma utilización. Se genera con el parámetro de GEN4MAST WORKLOAD con valor "SCALE-WCET".

Los valores de generación de estos grupos secundarios se resumen en la Tabla 6-2.

	Base	**L-Rand**	**S-50**	**UP-Rand**	**B-100**	**SCALE**
Tipo de longitud de los flujos	**Fija**	**Aleatoria**	Fija	Fija	Fija	Fija
% de flujos simples	**0 %**	0 %	**50 %**	0 %	0 %	0 %
Distribución de la utilización entre los procesadores	**Uniforme**	Uniforme	Uniforme	**Aleatoria**	Uniforme	Uniforme
Tiempos de ejecución de mejor caso	**0**	0	0	0	**100% de C_{ij}**	**0**
Método de generación de la carga	***UUnifast***	*UUnifast*	*UUnifast*	*UUnifast*	*UUnifast*	***SCALE-WCET***

Tabla 6-2 Características de los sistemas L-Rand, S-50, UP-Rand, B-100 y SCALE.

Para todos los grupos descritos (salvo el grupo SCALE) generamos dos versiones distintas con cargas computaciones calculadas con *SCALE-WCET* y *UUnifast*. De la misma manera, y salvo para las variaciones de plazos de principio a fin, en todos los grupos generados aplicamos 4 tipos de plazos de principio a fin distintos (ED={T, T1, T2, NT} en la nomenclatura de GEN4MAST). Para obtener resultados estadísticamente relevantes, generamos 30 series de utilizaciones con cada combinación de parámetros de generación (parámetro de GEN4MAST "POPULATION 30"), que acotan la desviación absoluta de la UMP al 1% (ver Sección 4.5).

6.2 Ejecución del estudio

Una vez creado el dominio del estudio aplicamos las diferentes técnicas de asignación de parámetros de planificación disponibles en MAST sobre el conjunto de ejemplos generado. Para cuantificar el rendimiento de los algoritmos de asignación utilizaremos principalmente la utilización máxima que consiguen planificar (Utilización Máxima Planificable, UMP) dentro de cada serie de utilizaciones generada. Como para el estudio de los algoritmos de análisis de planificabilidad, esta métrica nos proporciona un método directo y sencillo de comparación de las técnicas de optimización.

El estudio se lleva a cabo en sistemas FP, GC-EDF y LC-EDF. Para calcular los tiempos de respuesta de peor caso se utiliza el análisis de planificabilidad holístico, ya

que es el único disponible para los tres tipos de planificación, de modo que la técnica de análisis no tenga influencia en los resultados. Este hecho nos permite comparar directamente los resultados obtenidos entre las diferentes políticas de planificación. Adicionalmente, el análisis holístico es el que obtiene sus resultados más rápidamente, por lo que el tiempo total de ejecución del estudio se verá reducido.

Como se describió en el Capítulo 3, el algoritmo HOSPA posee una serie de parámetros que configuran su funcionamiento. Como paso previo a llevar a cabo la comparación de técnicas de asignación, en la siguiente sección se determinará la influencia de estos parámetros en la obtención de soluciones, con el objetivo de determinar qué valores son los que se deben utilizar en HOSPA para obtener los mejores resultados en el caso general.

Una vez determinados estos parámetros, en la Sección 6.4 se realizará el estudio comparativo de las técnicas de asignación de forma independiente para cada política de planificación, y utilizando para la configuración de HOSPA los parámetros que hemos determinado como los más apropiados.

6.3 Estudio de los parámetros de optimización de HOSPA

La técnica HOSPA, presentada en el Capítulo 3 de este trabajo, es un algoritmo heurístico para la asignación y optimización de parámetros de planificación en sistemas distribuidos heterogéneos (FP, GC-EDF y LC-EDF). Se trata de una técnica iterativa, tal como se describió en la Figura 3-7, y su formulación se basa en la utilizada por los algoritmos HOPA y HOSDA de asignación de prioridades fijas y plazos de planificación respectivamente.

El funcionamiento de HOSPA posee cuatro parámetros de configuración controlables por el usuario, heredados de HOPA y HOSDA:

- Los parámetros (k_a,k_r). Estos parámetros controlan el peso en la nueva asignación de parámetros de planificación de aspectos ortogonales: el margen hasta la planificabilidad de cada actividad dentro de su flujo (k_a), y el margen hasta la planificabilidad del recurso procesador en el que reside cada actividad.
- El método utilizado para el cálculo de la asignación de parámetros de planificación inicial. HOSPA tiene disponibles dos técnicas, PD y NPD, aparte del uso de una asignación concreta definida por el usuario.
- Número de iteraciones máximas a llevar a cabo para buscar una solución planificable.
- Número de sobre-iteraciones sobre una solución ya planificable para mejorarla.

Hasta ahora, para estos parámetros de configuración se han utilizado valores obtenidos en la práctica que funcionaban suficientemente bien para las situaciones en las que han sido puestos a prueba. Como valores apropiados para los parámetros (k_a,k_r), en la presentación del algoritmo HOPA [GUT95] se propone el rango entre 1,5 y 3. En la implementación de HOPA en MAST se llevan a cabo un máximo de 30 iteraciones con cada pareja de parámetros k que se prueban, y para la asignación inicial se utiliza PD. Aunque con estos valores se obtienen resultados satisfactorios, nunca se ha llevado a cabo

un estudio exhaustivo de la influencia que tienen estos parámetros en el funcionamiento del algoritmo HOSPA, ni en sus predecesores (HOPA y HOSDA).

En esta sección llevaremos a cabo este estudio, con el objetivo de obtener una guía sobre los valores más apropiados de estos parámetros para sistemas con distintas topologías. El estudio se lleva a cabo de manera independiente para las tres políticas de planificación, y sobre el mismo dominio de estudio general que utilizaremos para comparar las diferentes técnicas de asignación (ver Sección 6.1).

6.3.1 Configuración de HOSPA en sistemas FP

Procedemos a estudiar el comportamiento de los parámetros de configuración de HOSPA en sistemas FP.

6.3.1.1 Parámetros *k*

En primer lugar nos planteamos determinar qué valores para la pareja (k_a,k_r) son los más apropiados en sistemas FP. Para ello aplicamos HOSPA sobre el dominio del estudio generado en la Sección 6.1, probando distintos valores para esta pareja entre 0,01 y 100, con especial atención al rango que originalmente se establece como el más apropiado (entre 1,5 y 3). Como trabajo previo se ha probado un mayor rango de valores, y con distintos valores para k_a y k_r. Los resultados que aquí se presentan recogen las conclusiones más representativas.

Para esta prueba inicial establecemos 40 iteraciones como máximo. El algoritmo HOPA implementado en MAST itera un máximo de 30 veces por cada pareja (k_a,k_r) con resultados satisfactorios. En esta prueba inicial elevamos este número a 40 para asegurarnos de que eliminamos el efecto del número de iteraciones en este estudio inicial. En la Sección 6.3.1.3 se estudiará la influencia del número de iteraciones. Utilizamos una asignación inicial fija para crear siempre la misma situación de partida, en este caso con PD. En la Sección 6.3.1.2 se estudiará el efecto de la asignación inicial.

En la Tabla 6-3 se muestran las UMP medias obtenidas para los sistemas base y los grupos de variaciones primarias, en los que para las variaciones de plazos de principio a fin se incluye únicamente el grupo D=T, y para las variaciones de longitud de los flujos e2e se incluyen los valores extremos L-4 y L-20. Se añade como referencia el resultado obtenido por la asignación inicial PD sin iteraciones. Para flujos más largos (L-20) la tendencia es a obtener mejores resultados con valores de *k* en torno a 1,5 y 1,2. Para el grupo D=T, los valores de los parámetros *k* mayores son los que mejores resultados obtienen, mientras que en el grupo Base, la pareja *k* con valores iguales a 1,5 es la que mayor UMP obtiene. Sin embargo, desde un punto de vista global, podemos concluir que los valores de los parámetros *k* bajo estudio que obtienen mejores resultados son k_a=k_r=2.

En la Tabla 6-4 se muestran las UMP medias obtenidas por los mismos valores de los parámetros *k*, pero en esta ocasión para los sistemas pertenecientes a los grupos de variaciones secundarias. En esta situación los valores óptimos oscilan en el rango entre 1,5 y 10.

	Base	P-10	L-4	L-20	F-5	R-1000	D=T
PD	60,87	62,73	83,27	38,53	63,13	73,93	30,4
Ka=Kr=0,01	60,87	62,73	83,27	38,73	63,67	73,93	30,4
Ka=Kr=0,1	61	63,13	83,27	39,6	64,13	74	30,13
Ka=Kr=1	71,8	70,6	87,53	45,67	71,2	79,87	30,47
Ka=Kr=1,2	72,87	73,93	88,2	45,93	71	79,87	30,53
Ka=Kr=1,5	**74,87**	73,73	89,4	45,47	71,93	81,47	30,73
Ka=1,5 Kr=1,2	73,33	**74,07**	89,27	**47,13**	71	80,2	30,73
Ka=1,2 Kr=1,5	73,87	73,27	88	46,2	71,27	80,2	30,67
Ka=Kr=2	73,87	**74,07**	**89,6**	45	**72,07**	**81,53**	31
Ka=Kr=5	74,2	72,8	89,6	44	71,87	81,2	**31,53**
Ka=Kr=10	70,07	71,73	89,33	42,73	71,13	80,4	31,47
Ka=Kr=100	62,93	64,6	85,53	39,47	66	75,67	31,33

Tabla 6-3 UMP media obtenida por HOSPA para diferentes parejas de parámetros *k* con sistemas FP, para los grupos de variaciones primarias

A la vista de estos resultados podemos determinar que usar un valor para k_a y k_r de 2 es apropiado para el caso general, ya que o bien es el valor óptimo entre todos los valores probados, o cuando no es el óptimo se sitúa a menos de un 2,5% en la UMP media del que sí lo es. Puesto que las diferencias observadas en el entorno de los mejores valores de *k* son mínimas, no consideramos necesario llevar a cabo un estudio más exhaustivo probando más valores de *k*, ya que la posible mejora que se podría obtener sería previsiblemente baja.

	L-Rand	S-50	UP-Rand	SCALE	B-100
PD	76,6	58,6	54,47	51,33	61,93
Ka=Kr=0,01	76,6	58,6	54,47	51,33	62
Ka=Kr=0,1	76,73	58,6	54,6	51,33	62
Ka=Kr=1	83,4	74,53	63,67	69,27	73,2
Ka=Kr=1,2	84,2	76,2	64,93	70,53	75,2
Ka=Kr=1,5	85,47	78,4	**66,6**	71,73	76,2
Ka=1,5 Kr=1,2	85,2	77,93	65,4	71,6	75,4
Ka=1,2 Kr=1,5	84,27	76,8	65,2	71,2	76,13
Ka=Kr=2	85,73	**79,8**	65,73	72,33	**76,33**
Ka=Kr=5	**85,93**	78,2	64,13	**72,93**	75,73
Ka=Kr=10	85,4	75,6	61,53	**72,93**	73,4
Ka=Kr=100	79,87	63,67	55,93	59,6	64,07

Tabla 6-4 UMP media obtenida por HOSPA para diferentes parejas de parámetros *k* con sistemas FP, para los grupos de variaciones secundarias.

6.3.1.2 Asignación inicial

Los resultados presentados previamente se obtuvieron utilizando una asignación inicial PD. La otra asignación inicial soportada en HOSPA es NPD, que es un algoritmo que como se vio en el Capítulo 2, tiene en cuenta la utilización del procesador en el que reside la actividad. Cuando todos los procesadores poseen la misma utilización, las asignaciones PD y NPD obtienen el mismo resultado. Las diferencias entre PD y NPD se pueden observar cuando la utilización no es uniforme entre los recursos (grupo UP-Rand).

En la Tabla 6-5 se muestra la UMP media obtenida por HOSPA con los diferentes valores de los parámetros *k* para el grupo UP-Rand, cuando se utiliza NPD para la asignación inicial. Como referencia se incluye el resultado obtenido por HOSPA inicializado con PD con los valores de los parámetros *k* iguales a 2, y los resultados obtenidos por las asignaciones PD y NPD.

En esta tabla puede comprobarse que la UMP media obtenida por NPD es ligeramente superior a la que se obtiene con NPD. Sin embargo, esta ligera ventaja de NPD no se traduce en una UMP media mayor cuando se utiliza como inicialización para el algoritmo HOSPA, ya que se observa que con ambos tipos de inicialización se obtiene la misma UMP media, 65,73%.

A la vista de estos resultados, y teniendo en cuenta que PD y NPD son equivalentes para todos los grupos salvo UP-Rand, la combinación de valores de los parámetros *k* y tipo de inicialización que funciona mejor en el conjunto del dominio es $k_a=k_r=2$ con una inicialización PD o NPD.

HOSPA(PD) Ka=Kr 2	PD	NPD	Ka=Kr 0,01	Ka=Kr 0,1	Ka=Kr 1	Ka=Kr 1,2	Ka=Kr 1,5	Ka=1,5 Kr=1,2	Ka=1,2 Kr=1,5	Ka=Kr 2	Ka=Kr 5	Ka=Kr 10	Ka=Kr 100
65,73	54,47	55	55	55,2	62,4	64,4	65,53	65,53	**65,73**	65,2	63,2	61,73	56,4

Tabla 6-5 UMP media obtenida por HOSPA para distintos valores de *k* en sistemas FP utilizando NPD como asignación inicial, para los sistemas del grupo UP-Rand

6.3.1.3 Número de iteraciones para planificabilidad

Todas las ejecuciones de HOSPA hasta ahora se han llevado a cabo limitando a 40 el número de iteraciones que puede llevar a cabo para buscar una asignación planificable. Limitar el número de iteraciones es útil para acotar el tiempo de ejecución máximo de HOSPA, pero asignarle un valor muy bajo puede tener efectos negativos en su capacidad de buscar soluciones.

Aquí queremos determinar qué límite de iteraciones es el apropiado para que HOSPA pueda funcionar satisfactoriamente. Como primer paso, realizamos un estudio previo con sistemas con las características del sistema base definido en la Sección 6.1, y ejecutamos HOSPA con distintos números de iteraciones máximas, entre 1 y 160. Las UMP medias obtenidas para cada número de iteraciones máximas se representan en la Figura 6-1, en la que se puede comprobar que en promedio no se aprecian mejoras permitiendo a HOSPA iterar más de 40 veces.

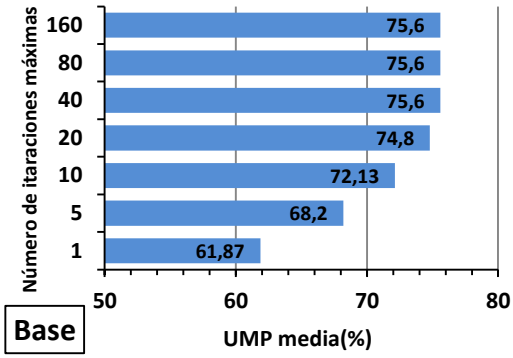


Figura 6-1 UMP media obtenida por HOSPA con diferentes límites de iteraciones en sistemas FP.

Una vez que hemos confirmado que 40 iteraciones es un límite apropiado para HOSPA, a continuación estudiamos cómo se comportaron las ejecuciones de HOSPA realizadas con la pareja de valores k_a=k_r=2 y con límite en 40 iteraciones. En la Figura 6-2 se representa el número de ejecuciones de HOSPA en porcentaje (eje de ordenadas izquierdo) para un total de aproximadamente 230000, respecto del número de iteraciones realizadas por HOSPA en cada ejecución para encontrar una solución planificable (eje de abscisas). Se puede comprobar que en un 75,36% de los casos en los que HOSPA encontró una solución planificable, ésta fue encontrada en la primera iteración, esto es, en la asignación inicial. Sin embargo, el porcentaje de casos en los que se encuentra una solución en su segunda iteración se reduce al 7%, y a menos de 0,1% de los casos para 16 iteraciones en adelante. Por último, HOSPA encontró una solución planificable en su iteración número 40 en menos de 1 de cada 10000 casos.

En la Figura 6-2 también se representa el promedio de las utilizaciones de los sistemas planificados (eje de ordenadas derecho) respecto del número de iteraciones realizadas por HOSPA, y se comprueba que las ejecuciones para los casos en los que las utilizaciones son más bajas requieres menos iteraciones o pueden ser resueltos por el algoritmo de asignación inicial (caso de 1 iteración).

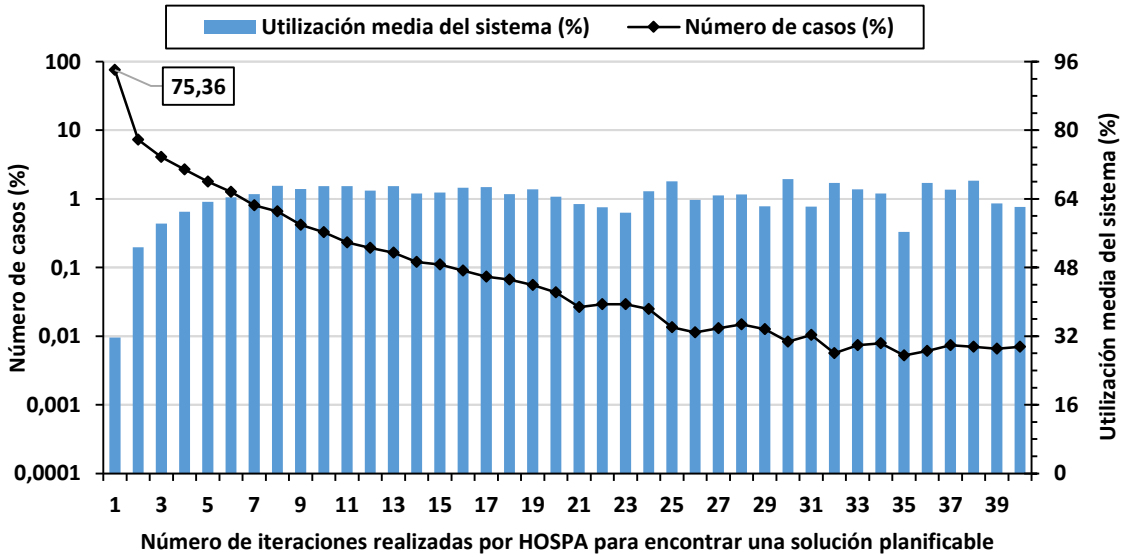


Figura 6-2 (Eje vertical izquierdo) Histograma en porcentaje del número de iteraciones llevadas a cabo por HOSPA para encontrar una solución planificable, y (Eje vertical derecho), el promedio de las utilización de los sistemas planificados.

6.3.1.4 Número de sobre-iteraciones de optimización

A continuación evaluaremos, para sistemas FP, la capacidad de HOSPA para optimizar una solución ya planificable sobre-iterando sobre ella. Con este objetivo, ejecutamos HOSPA con cuatro valores distintos de sobre-iteraciones: 0, 1, 5 y 10, siendo 0 sobre-iteraciones el caso en el que HOSPA finaliza en el momento en el que se encuentra una solución planificable.

La mejor figura de mérito para comparar dos asignaciones de parámetros de planificación es el *slack* del sistema que se obtiene con cada una de ellas. A modo de recordatorio, el *slack* del sistema para el caso de que éste sea planificable, se define como el porcentaje máximo en el que se pueden aumentar todos los tiempos de ejecución de peor caso de forma que el sistema siga siendo planificable (ver Sección 1.4). Como criterio de comparación de soluciones, HOSPA utiliza un criterio similar al del *slack* pero que es computacionalmente más rápido de calcular, llamado el índice de planificabilidad [GUT95]. El índice de planificabilidad, al igual que el *slack,* da una idea del margen que tiene un sistema hasta el límite de planificabilidad, pero se calcula como la diferencia entre el plazo de principio a fin y el tiempo de respuesta de peor caso del flujo e2e.

En la Figura 6-3 se representan los *slacks* de sistema obtenidos por las diferentes ejecuciones de HOSPA, para el grupo Base, y tres modificaciones de éste que resumen

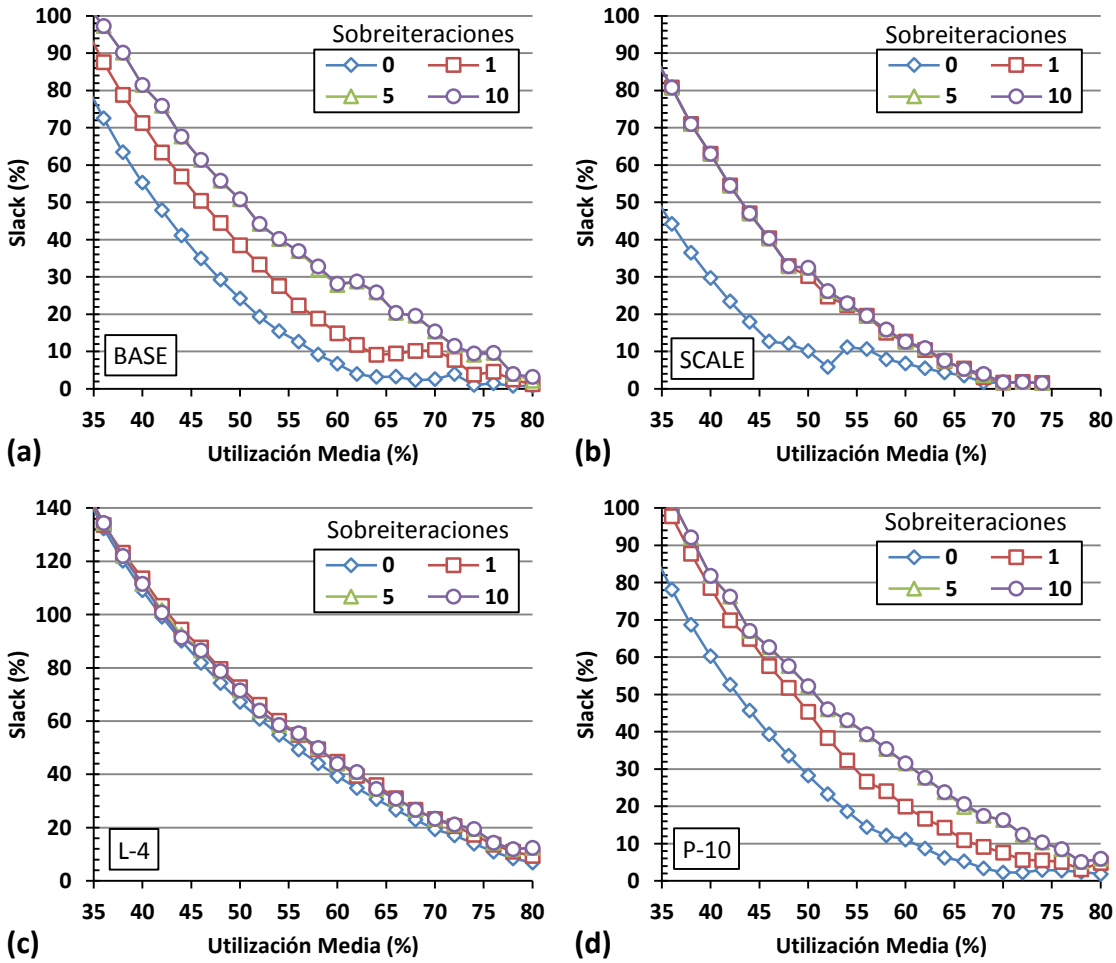


Figura 6-3 *Slack* obtenido por el algoritmo HOSPA en sistemas FP para distintos números de sobreiteraciones de optimización, para los grupos **(a) Base, (b) SCALE, (c) L-4, and (d) P-10**.

las características del dominio del estudio total: SCALE, L-4 (flujos de longitud 4), y P-10. Para el grupo L-4 se observa que HOSPA posee una capacidad limitada de optimizar soluciones en sistemas con flujos e2e cortos. En los resultados del resto de grupos se comprueba que con 5 sobre-iteraciones ya se consiguen prácticamente los beneficios máximos de optimización de HOSPA. Hacer notar que en estos tres grupos, las líneas que representan los casos de 10 y 5 sobre-iteraciones aparecen prácticamente superpuestas. El tiempo de ejecución necesario para sobre-iterar más de 5 veces no se compensaría con las mejoras inapreciables que se obtienen.

Como se vio en la Sección 6.3.1.1, en utilizaciones bajas de por ejemplo 35%, el algoritmo HOSPA típicamente ya ha encontrado una solución planificable en su primera iteración, esto es, con su asignación inicial PD en este caso. En la Figura 6-3 observamos que en los sistemas con utilización media del 35%, llevar a cabo una sola sobre-iteración produce un aumento significativo en el *slack* del sistema (salvo en el grupo L-4). Por tanto, HOSPA también es capaz de optimizar una asignación inicial PD ya planificable, además de encontrar soluciones planificables cuando PD no las obtiene.

6.3.2 Configuración de HOSPA en sistemas GC-EDF

Para estudiar el comportamiento de HOSPA en sistemas GC-EDF seguimos el mismo proceso que hemos llevado a cabo para los sistemas FP, es decir, aplicamos el algoritmo HOSPA con distintas combinaciones de valores para los parámetros *k*, tipo de asignación inicial y número de iteraciones.

En la sección 5.4 se mostró cómo para GC-EDF, los sistemas alcanzaban utilizaciones que prácticamente llegaban al 100%, cuando los flujos e2e poseían plazos de principio a fin iguales a su periodo multiplicado por su longitud ($D_i=N_i*T_i$), y se realizaba una asignación PD. En el estudio de las técnicas de asignación para GC-EDF, utilizar tales plazos de principio a fin no permitiría observar con claridad las diferencias de rendimiento, ya que las UMP medias observadas tendrían valores saturados en torno al 100%.

El dominio de estudio propuesto en la presente comparativa se crea alrededor de un sistema base con $D_i=N_i*T_i$, por lo que no resulta apropiado para su aplicación en sistemas GC-EDF. Por esta razón, en este estudio para sistemas GC-EDF se mostrarán los resultados de los diferentes grupos de sistemas generados, pero utilizando plazos de principio a fin limitados al primer tercio en el segmento $[T_i, N_i*T_i]$, esto es, plazos $D_i=T1$ según la nomenclatura GEN4MAST.

6.3.2.1 Parámetros *k*

Para determinar los valores de los parámetros *k* más apropiados para sistemas GC-EDF, aplicamos HOSPA con distintos valores para (k_a,k_r) sobre el dominio del estudio definido en la Sección 6.1, estableciendo que el sistema base ahora tiene unos plazos de principio a fin $D_i=T1$. Para estas ejecuciones utilizamos la asignación inicial PD, y limitamos a 40 el número de iteraciones que puede llevar a cabo para encontrar una solución, de igual manera a lo utilizado para sistemas FP previamente. Como referencia, se incluyen también los resultados obtenidos por la asignación inicial PD.

Los resultados en forma de la UMP media obtenida por cada pareja de valores de los parámetros *k* para los grupos de variaciones primarias (Tabla 6-6), y secundarias (Tabla 6-7) indican que la pareja k_a=k_r=10 es la más apropiada en la mayoría de situaciones. Para el grupo SCALE (sistemas con cargas generadas con SCALE-WCET), la pareja más apropiada es k_a=k_r=1,5. Para los grupos L-20 y UP-Rand observamos que se obtienen los mejores resultados con un rango amplio de valores de los parámetros (k_a,k_r), en el que también se incluye k_a=k_r=10.

	Base (ED=T1)	P-10 (ED=T1)	L-4 (ED=T1)	L-20 (ED=T1)	F-5 (ED=T1)	R-1000 (ED=T1)	ED=T
PD	90,36	85,27	88,4	95,78	84,4	90,8	29,4
Ka=Kr=0,01	90,67	85,27	88,4	95,78	84,4	90,4	29,4
Ka=Kr=0,1	91,67	85,27	88,4	95,8	84,4	91	29,6
Ka=Kr=1	90,73	85,27	88,6	**96**	84,8	91,11	29,8
Ka=Kr=1,2	91,09	85,27	88,6	**96**	85	91,6	29,8
Ka=Kr=1,5	92,33	85,45	88,6	**96**	85,8	92,4	29,8
Ka=1,5 Kr=1,2	91,67	85,27	88,6	**96**	85	91,56	30
Ka=1,2 Kr=1,5	91,83	85,27	88,4	**96**	85	92	30
Ka=Kr=2	92,55	85,64	90,2	**96**	85,2	92,67	30,4
Ka=Kr=5	93,82	86	92	**96**	86,4	93,56	30,8
Ka=Kr=10	**94,67**	**86,55**	**92,4**	**96**	**86,6**	**93,75**	**31**
Ka=Kr=100	93,27	86,36	91,4	**96**	86	93,14	30,6

Tabla 6-6 UMP media obtenida por HOSPA para diferentes parejas de parámetros *k* con sistemas GC-EDF, para los grupos de variaciones primarias.

	L-Rand (ED=T1)	S-50 (ED=T1)	UP-Rand (ED=T1)	SCALE (ED=T1)	B-100 (ED=T1)
PD	86	87,45	89,67	68,36	77,09
Ka=Kr=0,01	86	88,18	**90**	69,27	77,64
Ka=Kr=0,1	86	88,18	**90**	69,27	78
Ka=Kr=1	87,82	88,18	**90**	70	78,91
Ka=Kr=1,2	87,82	88,36	**90**	70,18	78,91
Ka=Kr=1,5	88,18	88,55	**90**	**70,55**	80
Ka=1,5 Kr=1,2	87,82	88,36	**90**	70,18	79,45
Ka=1,2 Kr=1,5	87,45	88,18	**90**	70,18	79,45
Ka=Kr=2	88,91	89,27	**90**	70	80,36
Ka=Kr=5	90	**90,73**	**90**	69,64	81,09
Ka=Kr=10	**90,73**	**90,73**	**90**	69,09	**82**
Ka=Kr=100	**90,73**	**90,73**	**90**	68,36	80,36

Tabla 6-7 UMP media obtenida por HOSPA para diferentes parejas de parámetros *k* con sistemas GC-EDF, para los grupos de variaciones secundarias.

6.3.2.2 Asignación inicial

Aplicamos el algoritmo HOSPA con inicialización NPD para comprobar si este tipo de inicialización muestra ventajas para sistemas GC-EDF. Puesto que la asignación PD y NPD son idénticas cuando todos los procesadores poseen la misma utilización, aplicamos HOSPA con NPD únicamente para el conjunto de sistemas UP-Rand.

En la Tabla 6-8 se muestran las UMP medias obtenidas por HOSPA inicializado con NPD para los distintos valores de los parámetros *k* bajo estudio, y también se incluyen como referencia los resultados obtenidos por HOSPA inicializado con PD y *ka*=*kr*=10, y los resultados obtenidos por PD y NPD. El número de iteraciones se limita a 40. De estos resultados se extrae la conclusión de que la inicialización NPD no produce ningún efecto beneficioso adicional en los resultados.

A la vista de estos resultados se puede concluir que en términos generales la combinación óptima de valores de los parámetros *k* y tipo de inicialización a utilizar para sistemas GC-EDF es *ka*=*kr*=10 con inicialización PD o NPD.

HOSPA(PD) Ka=Kr=10	PD	NPD	Ka=Kr 0,01	Ka=Kr 0,1	Ka=Kr 1	Ka=Kr 1,2	Ka=Kr 1,5	Ka=1,5 Kr=1,2	Ka=1,2 Kr=1,5	Ka=Kr 2	Ka=Kr 5	Ka=Kr 10	Ka=Kr 100
90	89,67	89,64	89,64	89,64	89,64	89,64	89,64	89,64	89,64	89,64	90	90	90

Tabla 6-8 UMP media obtenida por HOSPA para distintos valores de *k* en sistemas GC-EDF utilizando NPD como asignación inicial, para los sistemas del grupo UP-Rand**.** Se incluyen los resultados de HOSPA(PD) y NPD.

6.3.2.3 Número de iteraciones para planificabilidad

Para determinar cuál es el límite de iteraciones máximas de HOSPA apropiado para sistemas GC-EDF, procedemos de la misma manera que para sistemas FP (ver Sección 6.3.1.3). En primer lugar llevamos a cabo un estudio preliminar, en el que sobre un conjunto de sistemas con las características del sistema base y plazos de principio a fin D_i=T1, ejecutamos HOSPA con diferentes números de iteraciones máximas entre 1 y 160. En la Figura 6-4 se representan las UMP medias obtenidas para cada valor de iteraciones máximas. Se comprueba que con 5 iteraciones máximas, ya se consiguen en promedio prácticamente los mismos resultados que iterando 160 veces. Con un límite de 40 iteraciones, se obtiene una UMP media que está a 0,14% de distancia en la UMP media de la obtenida con un límite de 160 iteraciones.

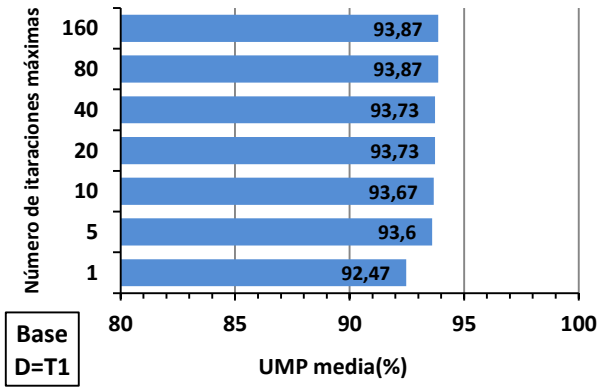


Figura 6-4 UMP media obtenida por HOSPA con diferentes límites de iteraciones, para los sistemas base, y planificación GC-EDF.

A continuación estudiamos cómo se comportaron las ejecuciones de HOSPA con límite en 40 iteraciones, realizadas con la pareja de valores k_a=k_r=10 en el dominio de estudio completo definido en la Sección 6.1 y sistemas GC-EDF.

En la Figura 6-5 se representa el número de ejecuciones de HOSPA en porcentaje (eje de ordenadas izquierdo) para un total de aproximadamente 110000 ejecuciones, respecto del número de iteraciones realizadas por HOSPA para encontrar una solución planificable (eje de abscisas). En la figura se comprueba que en un 97,27% de los casos en los que HOSPA encontró una solución planificable, ésta fue encontrada en su asignación inicial, mientras que en sólo un 0.36% de los casos la encontró en la segunda iteración. Para mayores número de iteraciones, el porcentaje de ocurrencias es menor. Estos resultados confirman que aumentar el límite de iteraciones de HOSPA en GC-EDF tiene un impacto muy pequeño en el resultado promedio.

En el eje de ordenadas derecho de la Figura 6-5 se representa el promedio de las utilizaciones de los sistemas planificados respecto del número de iteraciones realizadas por HOSPA. El reducido número de casos en el que HOSPA encuentra una solución iterando en más de una ocasión hace que el promedio de las utilizaciones medias tengan grandes desviaciones.

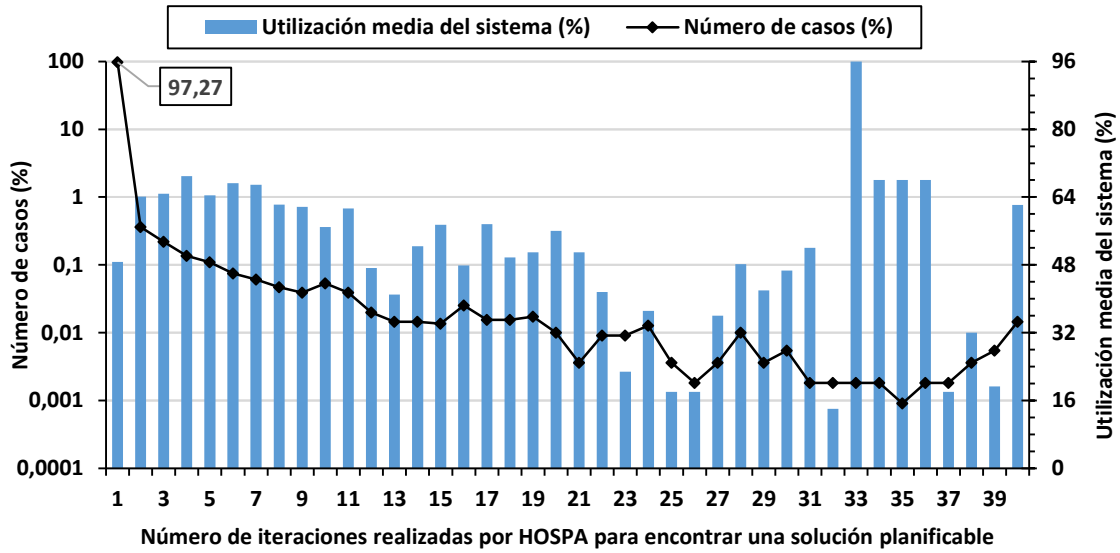


Figura 6-5 Histograma del número de repeticiones que realizó HOSPA para encontrar una solución planificable en sistemas GC-EDF.

6.3.2.4 Número de sobre-iteraciones de optimización

Para evaluar la capacidad de HOSPA de optimizar una solución ya planificable en sistemas GC-EDF, actuamos de la misma manera que en los sistemas FP (Sección 6.3.1.4), ejecutamos HOSPA con cuatro valores distintos de sobre-iteraciones: 0, 1, 5, 10. En la Figura 6-6 se muestran los *slacks* de sistema obtenidos por las diferentes ejecuciones de HOSPA para los grupos Base, SCALE, L-4 y P-10. Puede comprobarse que en GC-EDF, las ejecuciones con diferentes número de sobre-iteraciones obtienen en promedio unos *slacks* similares para todo el rango de utilizaciones bajo estudio. Por lo tanto, se puede concluir que HOSPA apenas tiene capacidad de optimización para planificación GC-EDF.

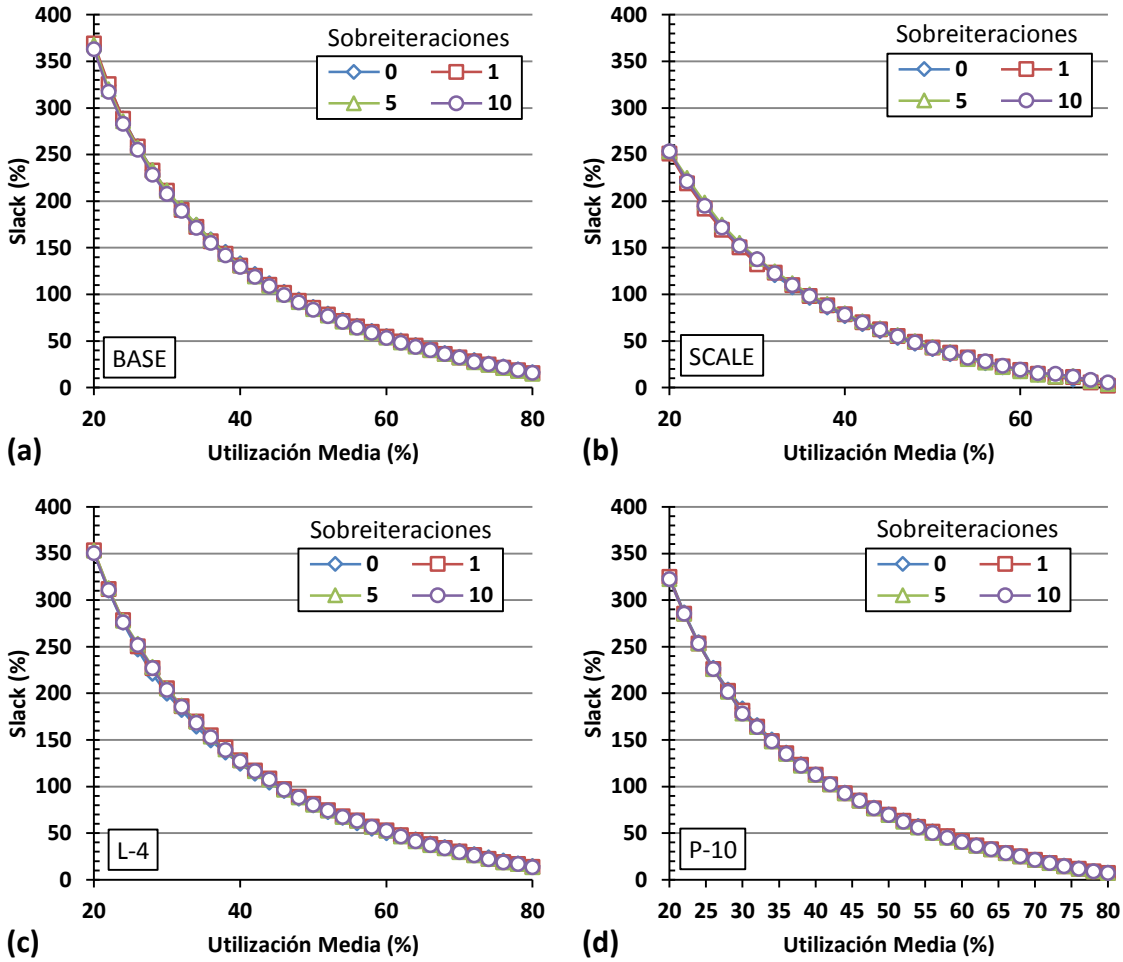


Figura 6-6 *Slack* obtenido por el algoritmo HOSPA en sistemas GC-EDF para distintos números de sobreiteraciones de optimización, para los grupos **(a) Base, (b) SCALE, (c) L-4, and (d) P-10**.

6.3.3 Configuración de HOSPA en sistemas LC-EDF

Para concluir el estudio del algoritmo HOSPA, estudiamos el comportamiento de sus parámetros en sistemas LC-EDF. Para ello seguiremos el mismo proceso llevado a cabo anteriormente para FP y GC-EDF. Al contrario que en GC-EDF, en LC-EDF los plazos $D_i=N_i*T_i$ son apropiados para poder observar la diferencia en la UMP media de las diferentes combinaciones de parámetros, por lo que utilizamos los mismos sistemas utilizados en el estudio para sistemas FP, cuyos plazos de principio a fin son $D_i=N_i*T_i$.

6.3.3.1 Parámetros *k*

En primer lugar queremos determinar qué valores para los parámetros *k* son los más apropiados en sistemas LC-EDF. Para ello aplicamos HOSPA al dominio del estudio, utilizando distintos valores para la pareja de parámetros *k*, y observamos la UMP media obtenida en cada uno de los grupos del dominio. De la misma manera a lo utilizado para sistemas FP y GC-EDF previamente, en esta primera aproximación se utiliza PD como asignación inicial, y se limita a 40 el número de iteraciones. En la Tabla 6-9 se muestran

las UMP medias obtenidas para los grupos de variaciones primarias, donde puede observarse que la pareja de valores k_a=k_r=10 obtiene los mejores resultados en todos los grupos salvo en L-4, en el que se sitúa a sólo 0.07% en la UMP media del mejor valor.

	Base	P-10	L-4	L-20	F-5	R-1000	ED=T
PD	56,13	53,13	81,07	35,53	53,47	66,6	25,2
Ka=Kr=0,01	56,13	53,13	81,07	35,53	53,53	66,6	25,2
Ka=Kr=0,1	56,13	53,13	81,07	35,53	53,53	66,6	25,2
Ka=Kr=1	56,27	53,13	81,93	35,53	53,87	66,73	25,2
Ka=Kr=1,2	56,33	53,27	82,2	35,6	54,13	66,87	25,2
Ka=Kr=1,5	56,33	**53,33**	82,27	35,67	54,13	67,07	25,27
Ka=1,5 Kr=1,2	56,4	**53,33**	82,4	35,73	54	67	25,27
Ka=1,2 Kr=1,5	56,33	**53,33**	82,6	35,67	54	67,07	25,27
Ka=Kr=2	56,47	**53,33**	82,67	35,93	54,2	67,2	25,33
Ka=Kr=5	56,67	**53,33**	**83,27**	35,8	**54,53**	**67,6**	25,87
Ka=Kr=10	**56,87**	**53,33**	83,2	**36**	**54,53**	**67,6**	**26**
Ka=Kr=100	56,67	53,27	82,2	35,8	54,2	66,93	25,87

Tabla 6-9 UMP media obtenida por HOSPA para diferentes parejas de parámetros *k* con sistemas LC-EDF, para los grupos de variaciones primarias.

Para los grupos de variaciones secundarias, cuyos resultados se muestran en la Tabla 6-10, las conclusiones son similares. Se puede concluir por lo tanto que la pareja de valores *ka*=*kr*=10 resulta la mejor para la mayoría de los casos.

	L-Rand	S-50	UP-Rand	SCALE	B-100
PD	74,33	54,13	51,47	38,07	58,2
Ka=Kr=0,01	74,4	54,2	51,47	38,13	58,27
Ka=Kr=0,1	74,33	54,13	51,67	38,07	58,27
Ka=Kr=1	75,87	54,8	51,6	38,2	58,47
Ka=Kr=1,2	75,67	54,93	51,73	38,4	58,47
Ka=Kr=1,5	76,53	55,27	51,67	38,4	58,67
Ka=1,5 Kr=1,2	76,27	54,93	51,67	38,4	58,53
Ka=1,2 Kr=1,5	75,6	54,93	51,73	38,33	58,6
Ka=Kr=2	76,93	55,4	51,87	38,47	58,6
Ka=Kr=5	77,33	55,53	**52,13**	**38,67**	58,8
Ka=Kr=10	**77,4**	**56**	52,07	**38,67**	**59,2**
Ka=Kr=100	76,2	55,27	52	38,33	58,87

Tabla 6-10 UMP media obtenida por HOSPA para diferentes parejas de parámetros *k* con sistemas LC-EDF, para los grupos de variaciones secundarias.

6.3.3.2 Asignación inicial

En el siguiente paso se va a determinar qué tipo de asignación inicial para HOSPA es la que obtiene mejores resultados en sistemas LC-EDF. Para ello aplicamos HOSPA con inicialización NPD, y nos centramos en los resultados obtenidos en el grupo UP-Rand, que es el único con utilización no uniforme entre los procesadores, y por lo tanto, el único en el que se pueden observar las diferencias entre PD y NPD.

En la Tabla 6-11 se muestran los resultados obtenidos en términos de la UMP media, y puede comprobarse cómo utilizar NPD tiene efectos negativos en los sistemas LC-EDF, obteniendo resultados peores a los obtenidos por HOSPA con PD.

A la vista de estos resultados se puede concluir que en sistemas LC-EDF la combinación de valores de los parámetros k y tipo de inicialización con la que en términos generales se obtienen los mejores resultados es k_a=k_r=10 con inicialización PD.

HOSPA(PD) Ka=Kr 10	PD	NPD	Ka=Kr 0,01	Ka=Kr 0,1	Ka=Kr 1	Ka=Kr 1,2	Ka=Kr 1,5	Ka=1,5 Kr=1,2	Ka=1,2 Kr=1,5	Ka=Kr 2	Ka=Kr 5	Ka=Kr 10	Ka=Kr 100
52,07	51,57	48,8	48,87	49	49,53	49,73	49,87	49,73	49,73	50	50,4	50,13	50

Tabla 6-11 UMP media obtenida por HOSPA para distintos valores de k en sistemas LC-EDF utilizando NPD como asignación inicial, para los sistemas del grupo UP-Rand

6.3.3.3 Número de iteraciones para planificabilidad

A continuación estudiamos la influencia del número máximo de iteraciones de HOSPA en sistemas LC-EDF. Para ello seguimos el mismo proceso llevado a cabo previamente para sistemas FP y GC-EDF. En primer lugar realizamos un estudio previo sobre un conjunto de sistemas con las características del sistema base, y sobre ellos ejecutamos HOSPA con distintos números de iteraciones máximas, entre 1 y 160. En la Figura 6-7 se representan las UMP medias obtenidas en este experimento, y se comprueba cómo en efecto el rendimiento de HOSPA en LC-EDF no está limitado por el número de iteraciones, obteniendo la práctica totalidad de las mejoras en las primeras 5 iteraciones.

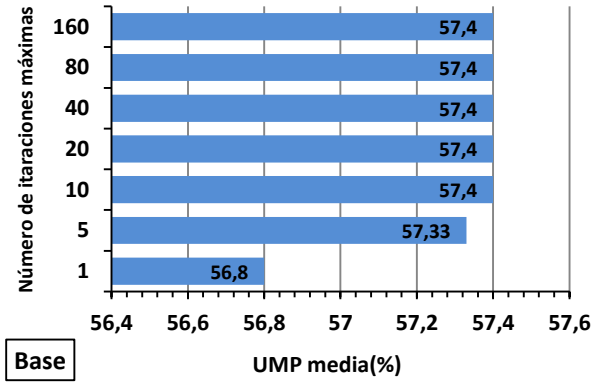


Figura 6-7 UMP media obtenida por HOSPA con diferentes límites de iteraciones, para los sistemas base, y planificación LC-EDF.

A continuación estudiamos cómo se comportaron las ejecuciones de HOSPA con límite en 40 iteraciones, realizadas con la pareja de valores k_a=k_r=10 en el dominio de estudio completo definido en la Sección 6.1 y sistemas LC-EDF.

En la Figura 6-8 se representa el número de ejecuciones de HOSPA en porcentaje (eje de ordenadas izquierdo) para un total de aproximadamente 150000 ejecuciones, respecto del número de iteraciones realizadas por HOSPA para encontrar una solución planificable (eje de abscisas). En la figura se comprueba que en un 91,54% de los casos en los que HOSPA encontró una solución planificable, ésta fue encontrada en su asignación inicial. Este porcentaje se reduce ya al 0,7% para la segunda iteración, y el porcentaje de ocurrencias a partir de 7 iteraciones se reduce por debajo del 0.1%. Estos resultados muestran que aumentar el límite de iteraciones de HOSPA en LC-EDF tiene un impacto limitado en el resultado promedio.

En el eje de ordenadas izquierdo de la Figura 6-8 se representa el promedio de las utilizaciones de los sistemas planificados con respecto el número de iteraciones realizadas por HOSPA. Al igual que lo observado para sistemas GC-EDF, el reducido número de casos en el que HOSPA encuentra una solución iterando en más de una ocasión provoca que el promedio de las utilizaciones medias tengan grandes desviaciones.

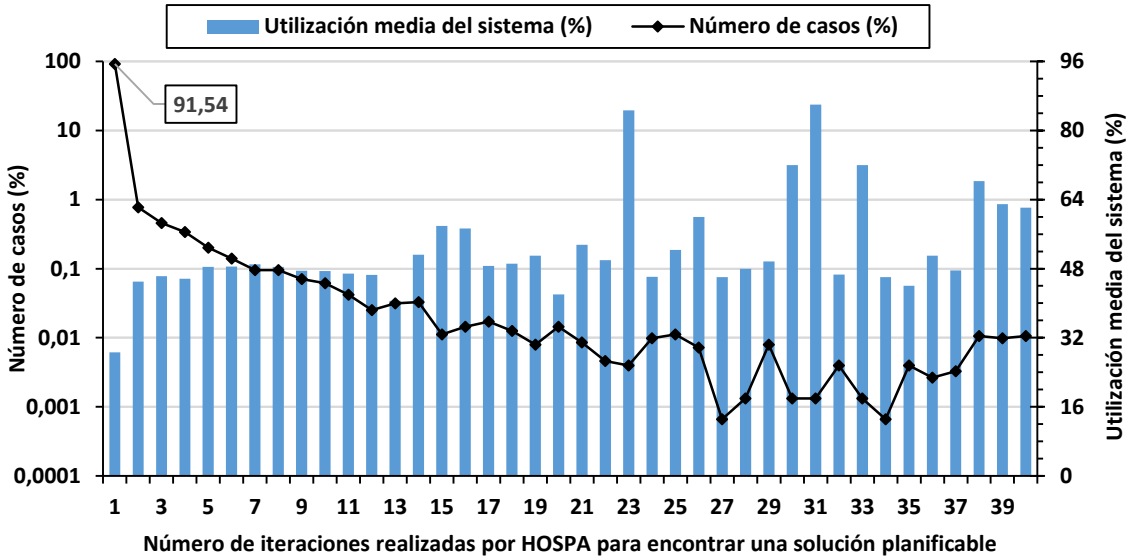


Figura 6-8 Histograma del número de repeticiones que realizó HOSPA para encontrar una solución planificable en sistemas LC-EDF.

6.3.3.4 Número de sobre-iteraciones de optimización

Para completar el estudio del comportamiento de HOSPA en sistemas LC-EDF, procedemos a estudiar su capacidad de optimizar soluciones ya planificables, siguiendo el mismo proceso llevado a cabo previamente para sistemas FP y GC-EDF. Para el caso de LC-EDF, en pruebas previas comprobamos que se observan mejoras relevantes sobre-iterando más de 10 veces. Por ello, en el presente estudio ejecutamos HOSPA con 5 valores distintos de sobre-iteraciones: 0, 1, 5, 10 y 20. En la Figura 6-9 se muestran los resultados en forma del *slack* del sistema para los grupos de sistemas Base, SCALE, L-4 y P-10. Al contrario que en GC-EDF, en LC-EDF observamos que HOSPA sí posee capacidad de optimizar soluciones. En los sistemas con flujos más cortos (L-4), la capacidad de mejora en promedio es reducida, aunque es superior a la observada para sistemas FP. Para el resto de grupos, y especialmente en los sistemas del grupo Base, se comprueba cómo 10 sobre-iteraciones mejora los resultados obtenidos con 5 sobre-iteraciones. Las mejoras observadas al sobre-iterar más de 10 veces son en promedio

reducidas (notar que las líneas que representan las ejecuciones con 10 y 20 sobre-iteraciones aparecen prácticamente superpuestas en los resultados de los grupos SCALE, L-4 y P-10.

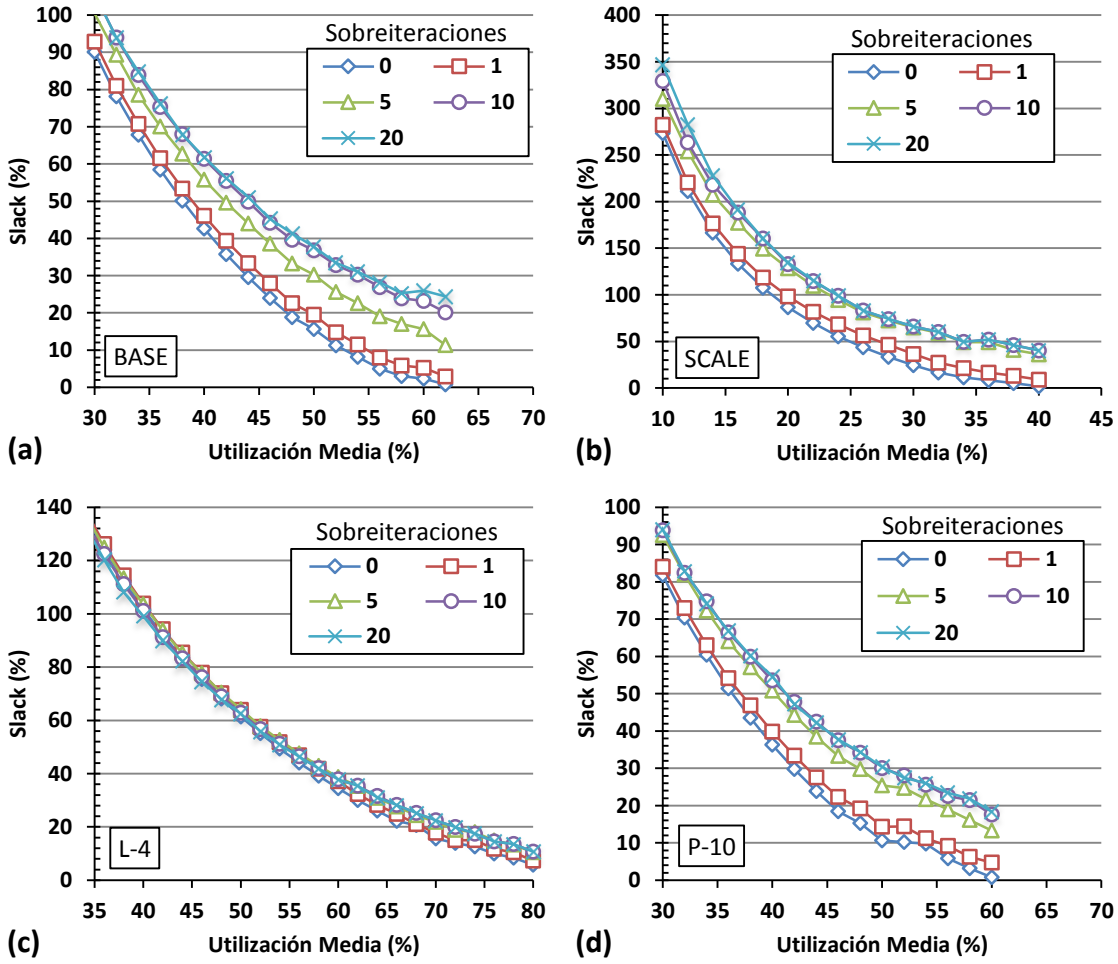


Figura 6-9 *Slack* obtenido por el algoritmo HOSPA en sistemas LC-EDF para distintos números de sobreiteraciones de optimización, para los grupos **(a) Base, (b) SCALE, (c) L-4, and (d) P-10**.

6.4 Comparativa de técnicas de asignación

El objetivo principal de este capítulo es el de llevar a cabo un estudio comparativo de las técnicas de asignación de parámetros de planificación para las distintas políticas de planificación de sistemas distribuidos que planteamos en este trabajo: FP, GC-EDF y LC-EDF. Para ello, en la sección 6.1 comenzamos definiendo el dominio sobre el que se va a aplicar el estudio, de forma que comprendiera un espectro lo más amplio posible de sistemas.

Las técnicas de asignación que serán objeto del estudio son las presentadas en los capítulos 2 y 3, esto es, UD, ED, PD, EQS, EQF y HOSPA. La técnica NPD no se incluye en el estudio ya que es equivalente a PD cuando todos los recursos procesadores poseen la misma utilización, y cuando esta condición no se cumple, las diferencias observadas en los resultados son mínimas, como ya se pudo comprobar en la Sección 6.3.

El algoritmo heurístico HOSPA tiene la peculiaridad de que su funcionamiento puede ser configurado por una serie de parámetros. En la sección anterior se estudió el efecto de estos parámetros, y con qué valores se obtiene en promedio los mejores resultados. En el presente estudio comparativo haremos uso de esos valores que hemos determinado como los más apropiados en el caso promedio, y que aparecen resumidos en la Tabla 6-12:

	k_a	k_r	Inicialización	Nº máximo de iteraciones
FP	2	2	PD	40
GC-EDF	10	10	PD	40
LC-EDF	10	10	PD	40

Tabla 6-12 Valores apropiados para los parámetros de ejecución de HOSPA

6.4.1 Planificación FP

Aplicamos a los sistemas del dominio del estudio las técnicas de asignación disponibles para FP: UD, ED, PD, EQS, EQF y HOSPA. En el caso de HOSPA usamos los parámetros que hemos determinado como los más apropiados, resumidos en la Tabla 6-12.

En la Figura 6-10 se muestran las UMP medias obtenidas por las diferentes técnicas en función de la variación de los plazos de principio a fin entre $D_i=T_i$ y $D_i=20*T_i$ sobre el sistema base. Cuando se utiliza *SCALE-WCET* como método de generación de la carga (Figura 6-10a), los algoritmos HOSPA, EQS y EQF son los que mejores resultados obtienen (notar que sus líneas aparecen prácticamente superpuestas), salvo para los plazos de principio a fin más pequeños, situación en la que HOSPA tiene una ligera ventaja. La asignación ED se sitúa en un término intermedio en cuando a rendimiento, mientras que UD y PD obtienen los peores resultados. Es especialmente llamativo comprobar cómo PD posee un rendimiento similar a UD, que es un algoritmo mucho más trivial del que en principio se debería esperar un rendimiento limitado. El pobre rendimiento de PD en esta situación no es impedimento para que HOSPA sea capaz de obtener unos buenos resultados utilizándolo como asignación inicial.

Cuando la carga se genera con *UUnifast* (Figura 6-10b) se obtienen conclusiones ligeramente distintas. Para plazos de principio a fin pequeños, HOSPA y su asignación inicial PD obtienen los mejores resultados. Sin embargo, a medida que aumentan estos

plazos de principio a fin, el algoritmo que consigue mejores planificaciones es EQF, con una clara distancia sobre HOSPA. También se observa cómo ED adelanta a PD en torno a $D_i=12*T_i$. Como era de esperar *a priori*, el algoritmo UD es el que peores resultados obtiene. Independientemente del método de generación de la carga utilizada, se comprueba que a medida que aumentan los plazos de principio a fin, aumenta la capacidad de HOSPA para mejorar su asignación inicial PD.

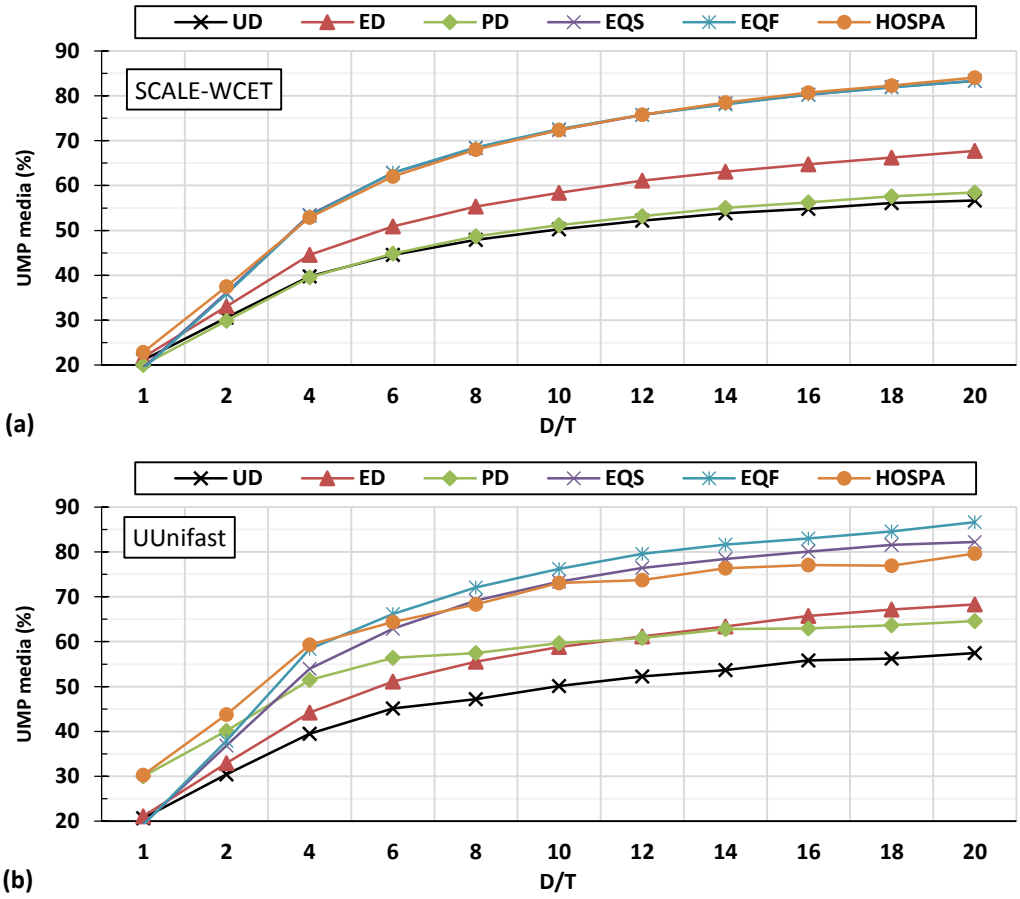


Figura 6-10 UMP media obtenida por los algoritmos de asignación de prioridades para FP, para distintos plazos de principio a fin y cargas generadas con **(a) *SCALE-WCET* y (b) *UUnifast***

En la Figura 6-11 se muestran las UMP medias obtenidas para diferentes longitudes de los flujos e2e. En primer lugar se puede observar cómo, a pesar de que los plazos de principio a fin se escalan a cada longitud generada, el *jitter* inducido al ir aumentando la longitud de los flujos e2e tiene un efecto negativo en la planificabilidad. Hay que recordar que el número de procesadores se mantiene constante en 5 en todos los casos. En cuanto al rendimiento de las distintas técnicas, se vuelve a comprobar cómo bajo cargas *SCALE-WCET* (Figura 6-11a), los algoritmos HOSPA, EQS y EQF obtienen rendimientos similares. Sin embargo, cuando las cargas son generadas con *UUnifast* (Figura 6-11b), EQF se distancia del resto en los casos con flujos e2e más largos. Es especialmente llamativo el resultado obtenido con 20 actividades/flujo, donde HOSPA posee una desventaja del 15% en la UMP media con respecto a EQF, y es igualado por el algoritmo ED. HOSPA en cambio obtiene los mejores resultados cuando los flujos e2e son más cortos.

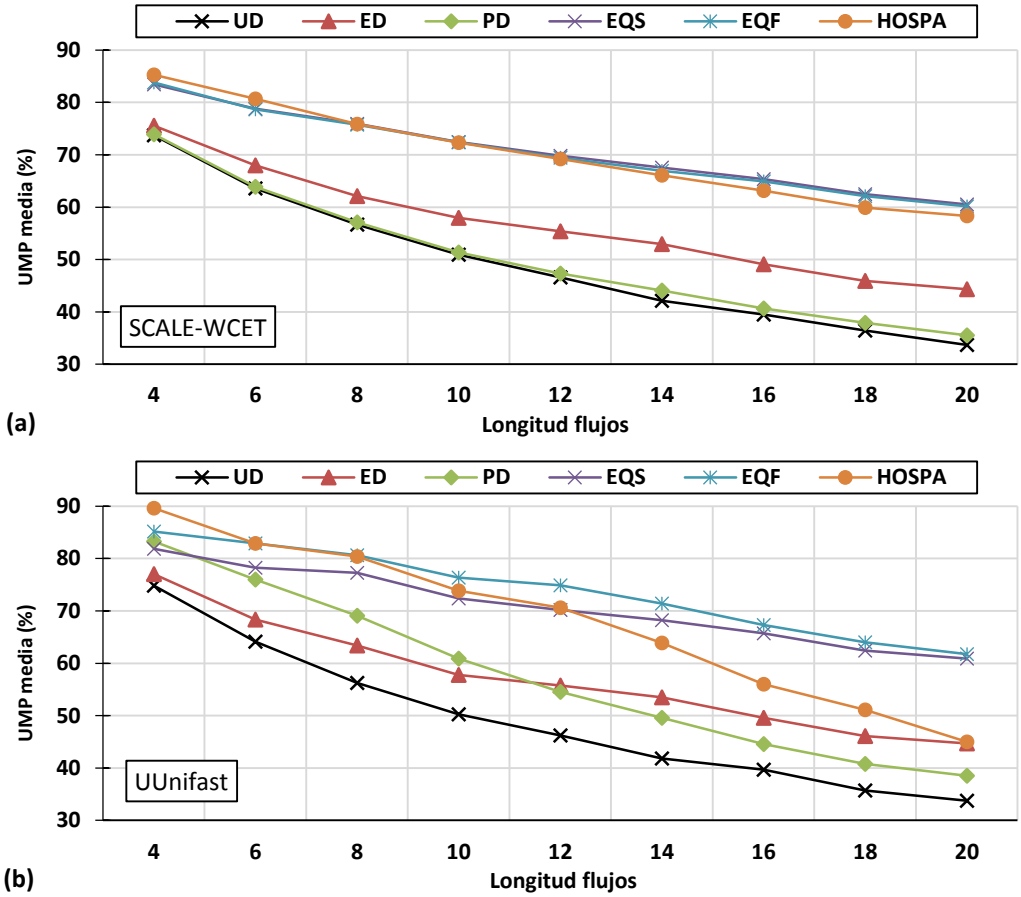


Figura 6-11 UMP media obtenida por los algoritmos de asignación de prioridades para FP, para distintas longitudes de los flujos, y cargas generadas con **(a) *SCALE-WCET* y (b) *UUnifast***

En la Figura 6-12 se muestran las UMP medias obtenidas en los grupos de 10 procesadores (P-10), 5 flujos (F-5) y periodos con ratio de 1000 (R-1000). Se puede comprobar que modificar el número de flujos e2e o los periodos no producen ningún cambio en el rendimiento relativo de unas técnicas respecto a otras, mostrándose EQF superior al resto. Sin embargo, aumentando el número de procesadores, el algoritmo HOSPA posee una ligera ventaja sobre EQF. Este resultado, combinado con la tendencia observada para flujos e2e más cortos, nos indica que HOSPA funciona mejor cuando los flujos e2e no repiten recurso procesador en su recorrido por el sistema.

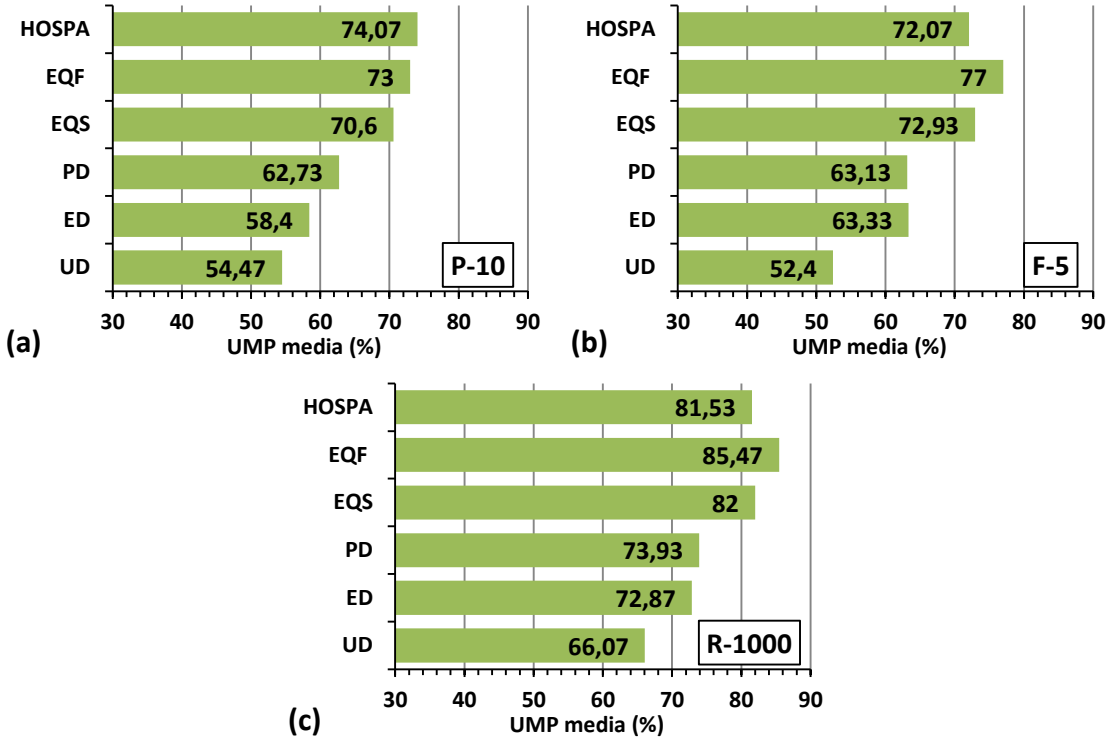


Figura 6-12 UMP media obtenida por los algoritmos de asignación de prioridades para FP, para los grupos de sistemas **(a) P-10, (b) F-5, y (c) R-1000**

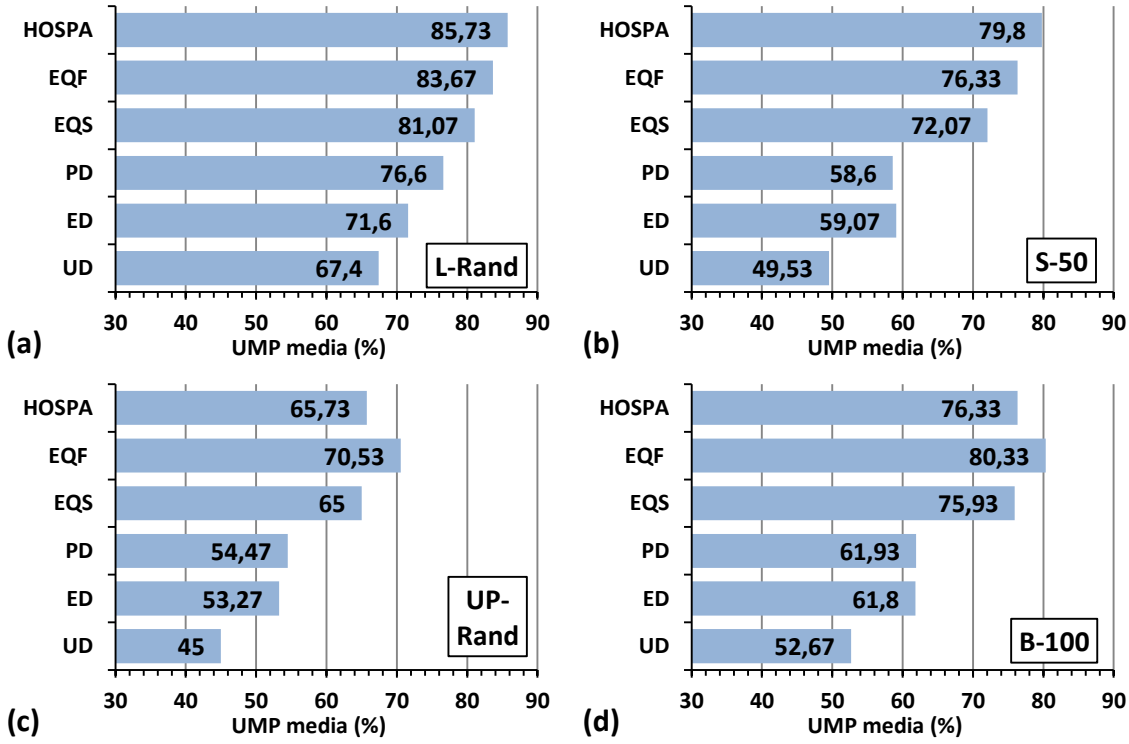


Figura 6-13 UMP media obtenida por los algoritmos de asignación de prioridades para FP, para los grupos de sistemas **(a) L-Rand, (b) S-50, (c) UP-Rand, y (d) B-100.**

Las UMP medias obtenidas en los grupos de variaciones secundarias (L-Rand, S-50, UP-Rand y B-100) se representan en la Figura 6-13. En estos grupos se observan las mismas tendencias que en los grupos anteriores; así, los algoritmos que mejor rendimiento obtienen son HOSPA, EQF y EQS. Otro fenómeno que se repite es que en los grupos en los que los flujos repiten procesador con menos frecuencia, esto es, los grupos L-Rand y S-50 al tener flujos e2e más cortos, el algoritmo HOSPA tiende a ser la técnica que obtiene los mejores resultados.

A la vista de los resultados para FP aquí presentados se puede concluir que existen tres factores que influyen principalmente en los resultados de las técnicas de asignación de prioridades fijas: la distribución de la carga entre las actividades, la dimensión de los plazos de principio a fin, y la relación entre la longitud de los flujos e2e y el número de procesadores en el sistema.

Para sistemas con cargas generadas con *SCALE-WCET*, en los que todas las actividades poseen la misma utilización (C_i/T_i), los mejores algoritmos son HOSPA, EQF y EQS, con rendimientos muy similares. Para cargas con distribuciones más uniformes entre las actividades (calculadas con *UUnifast*), el algoritmo con mejor rendimiento depende del tipo de sistema.

Para sistemas con cargas *UUnifast,* en las situaciones con plazos más restrictivos, como el caso de plazos $D_i=T_i$, el algoritmo que mejores resultados obtiene es HOSPA. La ventaja que posee sobre el resto es limitada debido a que el carácter tan restrictivo de estos plazos no deja mucho margen para la optimización a los algoritmos de asignación.

Se observó también que la relación entre la longitud de los flujos e2e y el número de procesadores en el sistema, influye en el rendimiento relativo de los algoritmos. Esta influencia se estudió desde dos ángulos distintos, (1) modificando la longitud de los flujos e2e y manteniendo el número de procesadores fijo, y (2) modificando el número de procesadores manteniendo la longitud de los flujos e2e fija. En ambos casos se observó que para flujos e2e que en promedio debían repetir recurso procesador con menos frecuencia, el algoritmo HOSPA era la asignación que mejores resultados obtenía. Sin embargo, si los flujos repetían recurso con mayor frecuencia, el algoritmo EQF resultaba más apropiado.

6.4.2 Planificación GC-EDF

Aplicamos las técnicas de asignación de plazos disponibles para GC-EDF sobre el dominio del estudio. Estas técnicas son UD, ED, PD, EQS, EQF y HOSPA. Para el caso de HOSPA, se ejecuta con los parámetros de configuración resumidos en la Tabla 6-12: asignación inicial PD, $k_a=k_r=10$, y un límite de 40 iteraciones.

En la Figura 6-14 se representan las UMP medias obtenidas por las diferentes técnicas de asignación variando los plazos de principio a fin entre $D_i=T_i$ y $D_i=20*T_i$ para cargas calculadas con *SCALE-WCET* y *UUnifast*. En ambos casos es fácil comprobar cómo las técnicas que mejores resultados obtienen son HOSPA y PD. Se observa también que HOSPA apenas consigue mejorar su asignación inicial PD, con mejoras entre el 1% y el 2% en la UMP media. Por otra parte, EQF y EQS poseen un rendimiento similar a HOSPA y PD en los plazos de principio a fin mayores, pero claramente inferior en el resto de sistemas. Esto es un indicativo de que la interpretación que hacemos de los

algoritmos EQS y EQF (ver Sección 2.4.3), en la que no se asegura que los plazos sean siempre ascendentes en el flujo e2e, produce un deterioro en su rendimiento en GC-EDF.

Otro resultado relevante de la Figura 6-14 es que se observa cómo a partir de plazos de principio a fin en torno a $D_i=8*T_i$, los algoritmos HOSPA y PD ya consiguen planificar sistemas con la máxima utilización total estudiada. Hay que recordar que este límite se fijó en 96% para reducir los tiempos de ejecución del análisis.

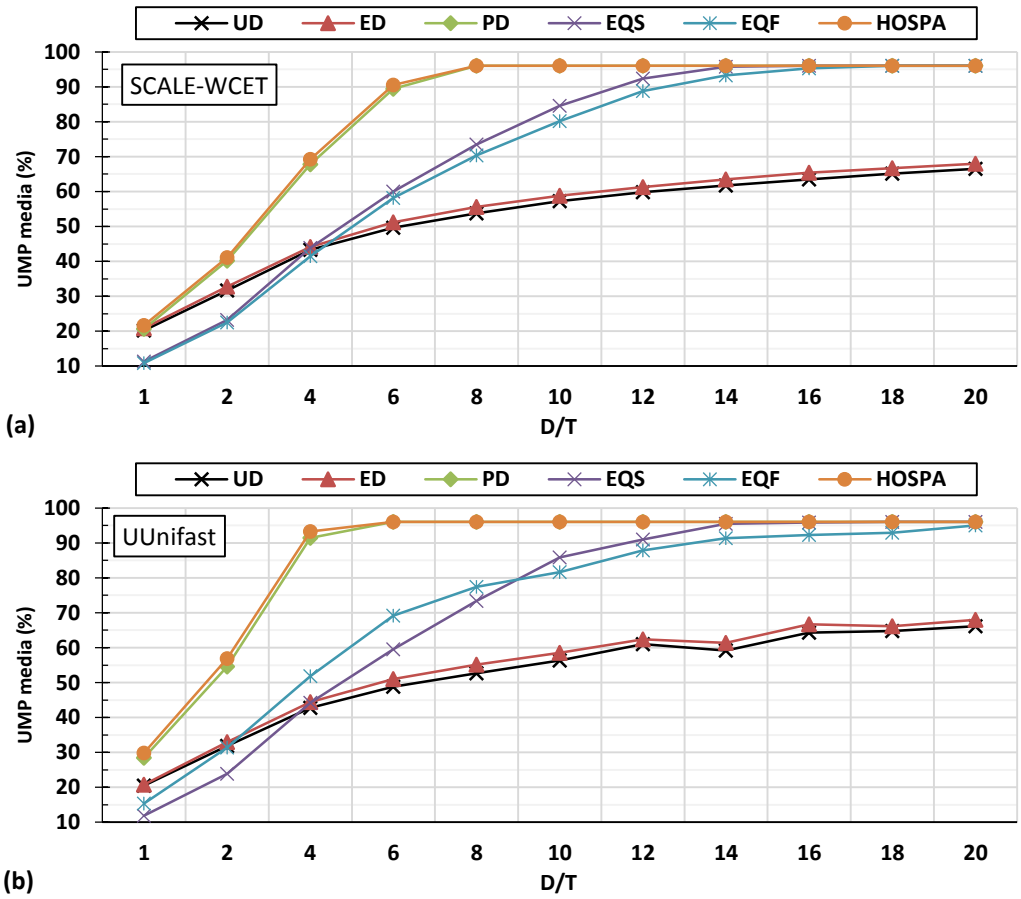


Figura 6-14 UMP media obtenida por los algoritmos de asignación para GC-EDF, para distintos plazos de principio a fin y cargas generadas con **(a) SCALE-WCET y (b) UUnifast**

En la Figura 6-15 se muestran las UMP medias obtenidas por los algoritmos de asignación estudiados para un rango de longitudes de los flujos e2e entre 4 y 20, y cargas calculadas con *SCALE-WCET* y *UUnifast*. Para evitar saturar los resultados obtenidos por HOSPA y PD, y poder observar mejor las diferencias entre los distintos algoritmos, se muestran los resultados para sistemas con plazos $D_i=T1$ en vez de $D_i=N_i*T_i$. En la figura se vuelve a comprobar que HOSPA y PD son los algoritmos que mejores resultados obtienen, mostrando una gran distancia con el resto. Es especialmente llamativo el caso de cargas *UUnifast*, en el que HOSPA y PD muestran una ligera tendencia a mejorar las UMP medias a medida que los flujos e2e son más largos.

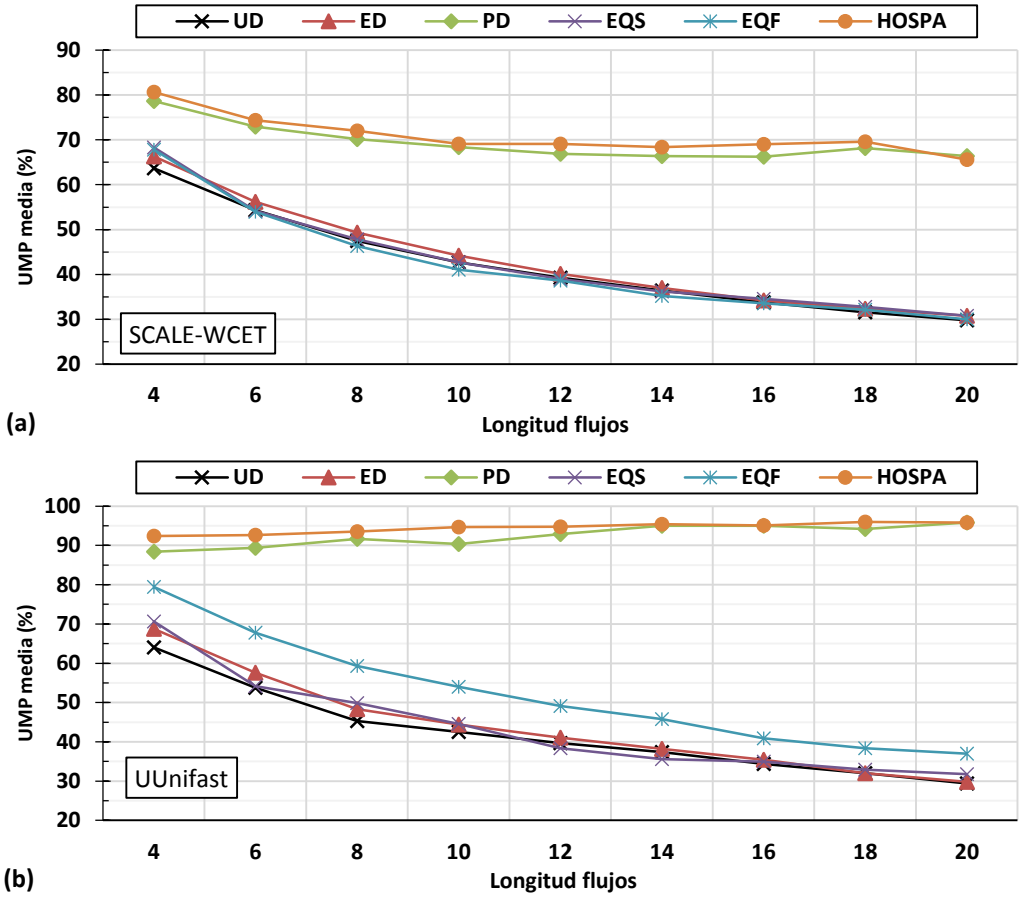


Figura 6-15 UMP media obtenida por los algoritmos de asignación para GC-EDF, para distintas longitudes de los flujos, y cargas generadas con **(a) SCALE-WCET y (b) UUnifast**

En la Figura 6-16 se presentan las UMP medias obtenidas para los grupos P-10, F-5 y R-1000. Los sistemas analizados en esta figura también poseen unos plazos de principio a fin D_i=T1 para evitar saturar los resultados de HOSPA y PD. Estos resultados vuelven a mostrar el mismo orden de rendimiento establecido en la figura anterior, en la que HOSPA es la técnica con mejores resultados, seguida por PD. El rendimiento de EQF y EQS es especialmente bajo en los sistemas con un rango amplio de periodos (grupo R-1000), ya que sus resultados son incluso peores que los obtenidos por los algoritmos triviales UD y ED.

Para finalizar el estudio para sistemas GC-EDF, en la Figura 6-17 se representan las UMP medias obtenidas por los grupos restantes (L-Rand, S-50, UP-Rand y B-100), cuando los flujos e2e poseen plazos D_i=T1. Los resultados de esa figura reafirman la conclusión de que independientemente de la topología del sistema, si la política de planificación es GC-EDF las técnicas que mejores resultados obtienen son HOSPA y su asignación inicial PD.

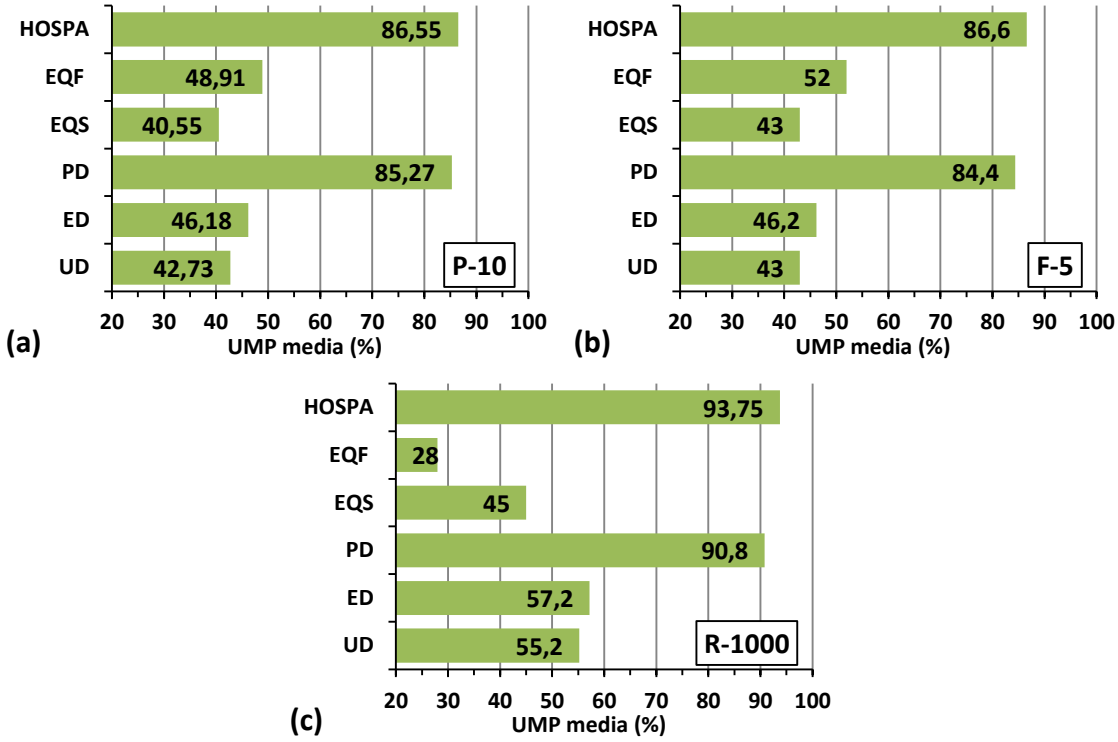


Figura 6-16 UMP media obtenida por los algoritmos de asignación para GC-EDF, para los grupos de sistemas **(a) P-10, (b) F-5, y (c) R-1000**

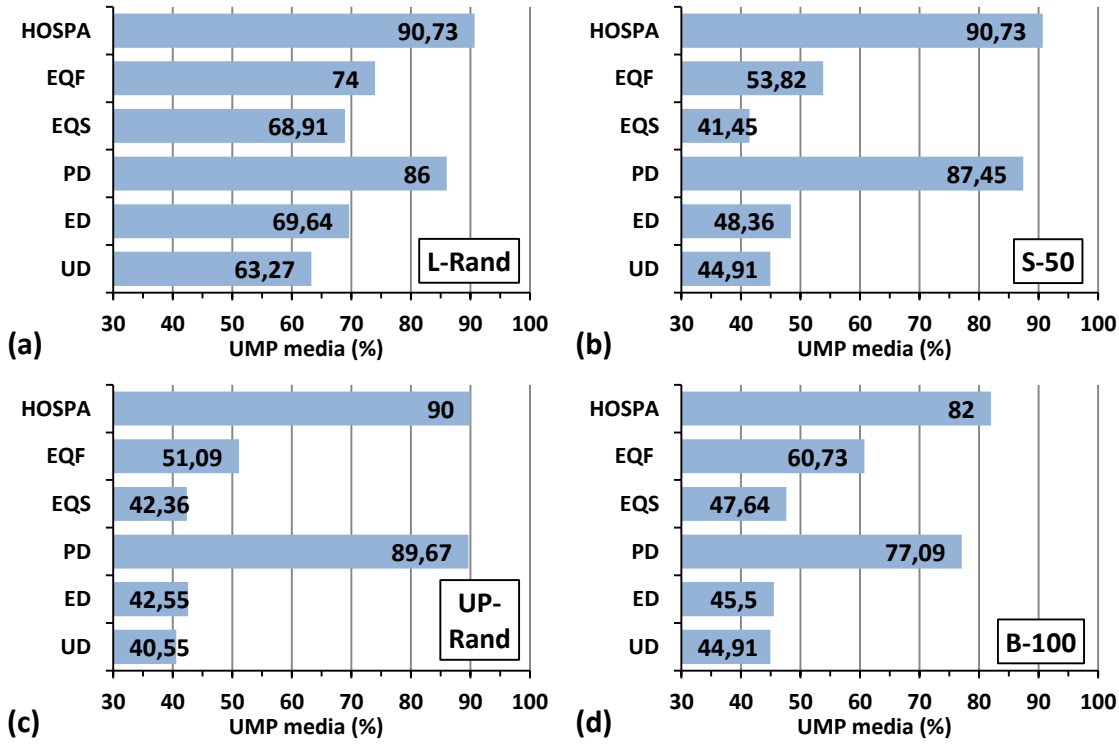


Figura 6-17 UMP media obtenida por los algoritmos de asignación para GC-EDF, para los grupos de sistemas **(a) L-Rand, (b) S-50, (c) UP-Rand, y (d) B-100**

A la vista de los resultados presentados en esta comparativa de técnicas de asignación de plazos globales de planificación para GC-EDF, se puede concluir que la asignación PD obtiene resultados especialmente buenos en cualquier tipo de sistema. Esta asignación se puede mejorar ligeramente iterando sobre ella utilizando el algoritmo heurístico HOSPA. Adicionalmente, se comprueba que las asignaciones EQS y EQF no consiguen un buen rendimiento en GC-EDF. Cabe recordar que estos dos últimos algoritmos producen una asignación de plazos globales de planificación no siempre ascendentes en el flujo e2e (ver Sección 2.4.3)

6.4.3 Planificación LC-EDF

Al contrario de lo que sucede con la planificación GC-EDF, en LC-EDF las UMP no alcanzan el máximo generado (96%) cuando tienen flujos e2e con plazos $D_i=N_i*T_i$ o próximos. Por esta razón, para el estudio de las técnicas de asignación en planificación LC-EDF utilizamos el dominio del estudio propuesto, con un sistema base con plazos $D_i=N_i*T_i$.

Sobre este dominio de estudio aplicamos las técnicas compatibles con la asignación de plazos locales de planificación conformes al plazo de principio a fin: PD y HOSPA. Para el caso de HOSPA, utilizamos una inicialización PD, con parámetros $k_a=k_r=10$, con 40 iteraciones como límite, tal y como se resume en la Tabla 6-12. Aunque en principio los plazos de planificación que se obtienen con los algoritmos UD y ED no tendrían sentido en la planificación LC-EDF, ya que la suma de los plazos asignados violaría el requisito

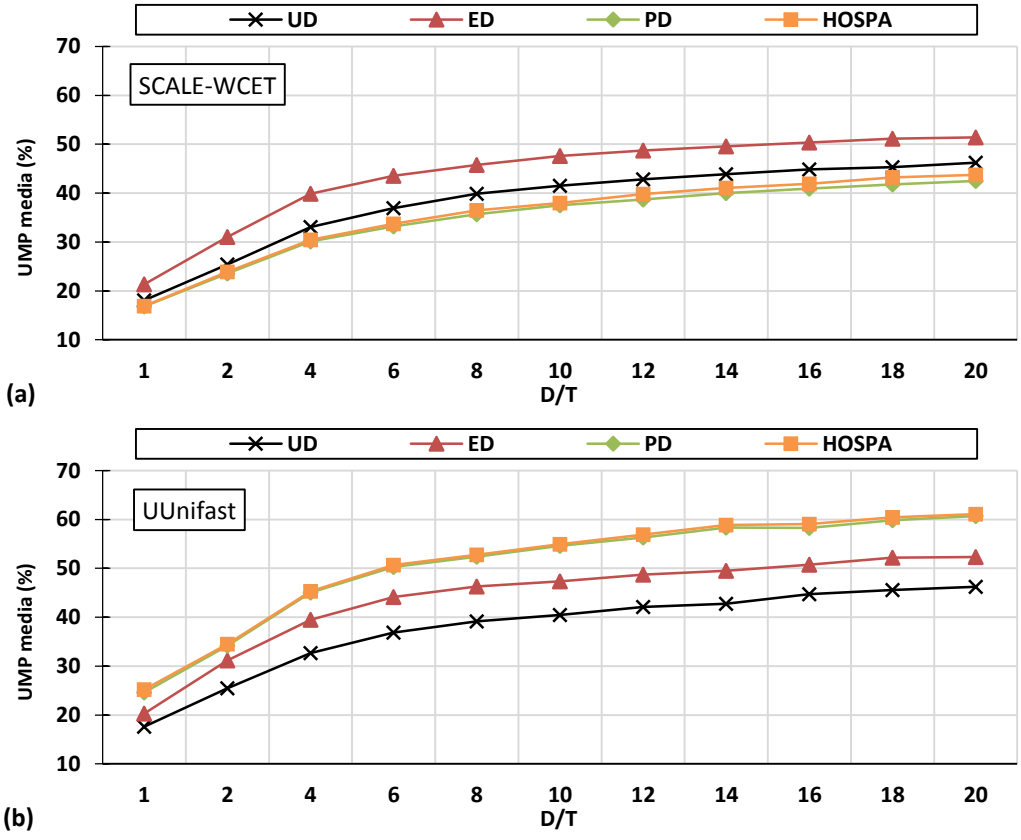


Figura 6-18 UMP media obtenida por los algoritmos de asignación para LC-EDF, para distintos plazos de principio a fin y cargas generadas con **(a) SCALE-WCET y (b) UUnifast**

impuesto por el plazo de principio a fin (no son conformes al plazo de principio a fin), vamos a estudiar también estos dos algoritmos triviales con objeto de completar la comparativa y añadir una referencia al rendimiento que obtengamos con PD y HOSPA.

En primer lugar en la Figura 6-18 se muestran las UMP medias obtenidas por las diferentes técnicas de asignación para diferentes plazos de principio a fin de los flujos e2e y cargas calculadas con *SCALE-WCET* y *UUnifast*. El pobre rendimiento general de LC-EDF en comparación con FP y GC-EDF observado en la comparativa de técnicas de análisis utilizando asignación PD (ver Sección 5.5), se vuelve a observar en estos resultados, en los que además se comprueba que el algoritmo HOSPA no es capaz de mejorar significativamente los resultados de su asignación inicial PD.

En la figura se observa también un resultado inesperado: utilizando *SCALE-WCET* como método de generación de las cargas, se puede observar que tanto UD como ED obtienen resultados mejores que los algoritmo que asignan plazos conformes a los plazos de principio a fin. Este fenómeno se observa para todo el rango de plazos de principio a fin estudiado. Sin embargo, con cargas *UUnifast*, HOSPA y PD son los algoritmos que mejores resultados obtienen, aunque su distancia con ED y UD no es muy amplia (entre un 10% y un 15% en las UMP medias). Adicionalmente resulta llamativo observar cómo la UMP media apenas aumenta una vez sobrepasado $D_i=10*T_i$, alcanzando unos máximos de sólo un 50% y 60% para cargas generadas con *SCALE-WCET* y *UUnifast* respectivamente, valores que están lejos de los alcanzados para FP y GC-EDF (ver Figura 6-10 para FP y Figura 6-14 para GC-EDF).

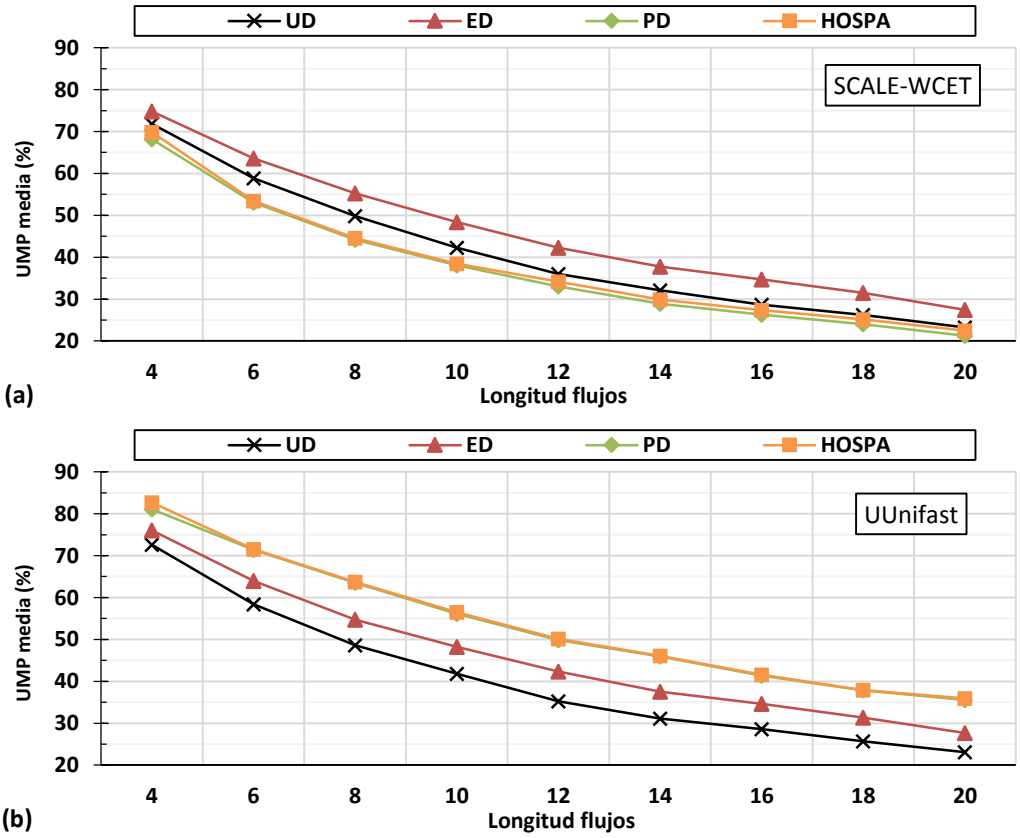


Figura 6-19 UMP media obtenida por los algoritmos de asignación para LC-EDF, para distintas longitudes de los flujos, y cargas generadas con **(a) SCALE-WCET y (b) UUnifast**

El efecto de la longitud de los flujos e2e se representa en la Figura 6-19, en la que se puede observar el mismo fenómeno inesperado. Para cargas generadas con *SCALE-WCET*, los algoritmos triviales UD y ED son los que mejores resultados obtienen. Con *UUnifast* sin embargo son HOSPA y PD los algoritmos con mejor rendimiento, aunque el resultado de HOSPA se obtiene principalmente en su asignación inicial PD, no pudiendo apenas mejorarla. Al contrario de lo observado con GC-EDF, aumentar la longitud de los flujos e2e tiene claramente un efecto negativo en la planificabilidad de los sistemas planificados por LC-EDF, con independencia del método utilizado para generar las cargas.

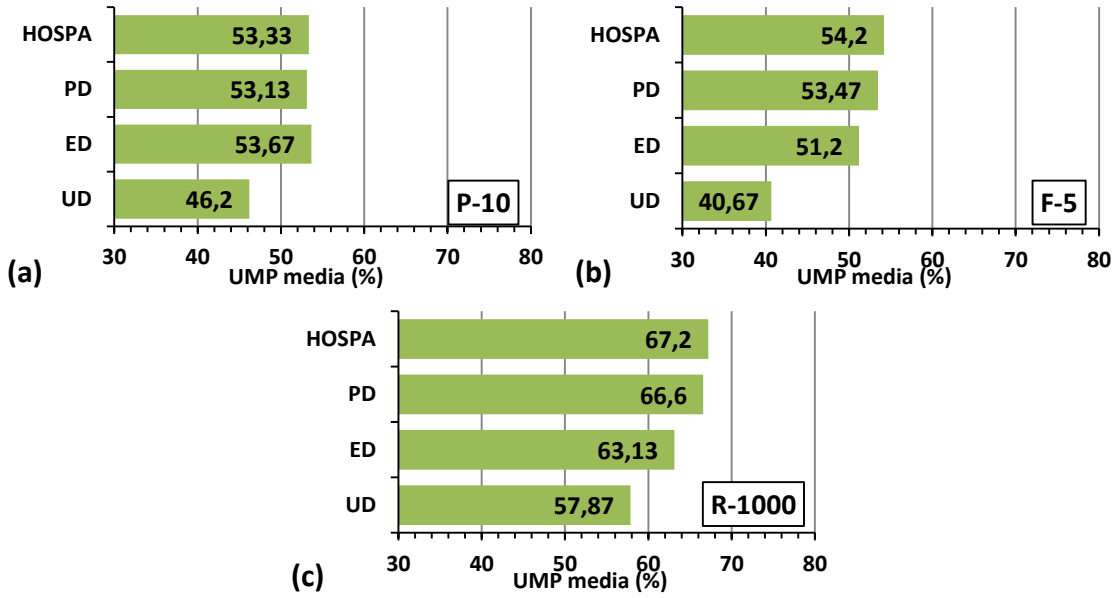


Figura 6-20 UMP media obtenida por los algoritmos de asignación para LC-EDF, para los grupos de sistemas **(a) P-10, (b) F-5, y (c) R-1000**

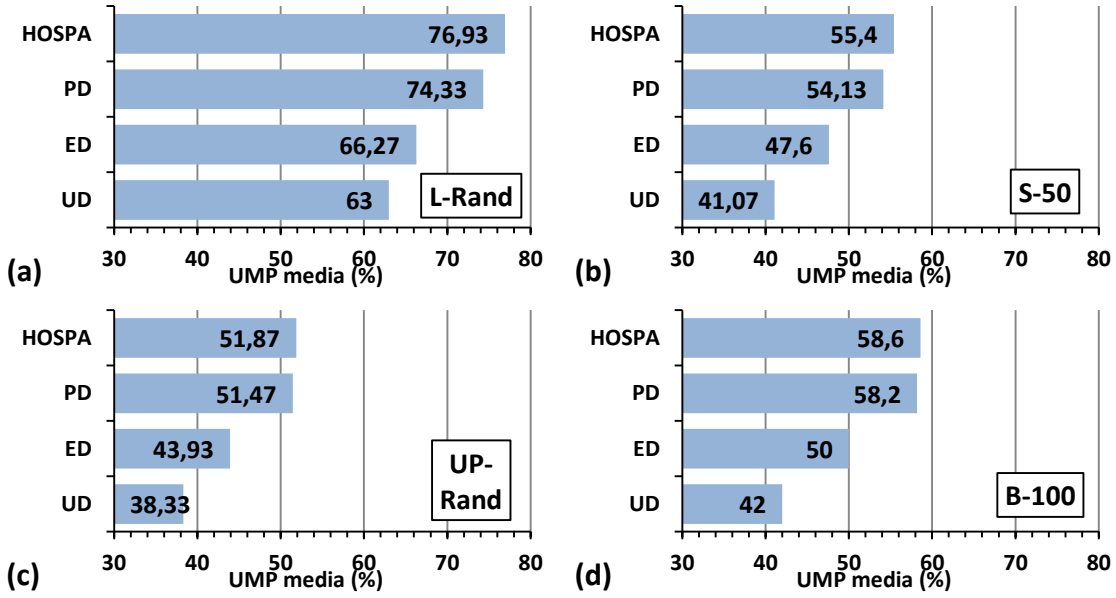


Figura 6-21 UMP media obtenida por los algoritmos de asignación para LC-EDF, para los grupos de sistemas **(a) L-Rand, (b) S-50, (c) UP-Rand, y (d) B-100**

En la Figura 6-20 y en la Figura 6-21 se representan las UMP medias obtenidas por los algoritmos de asignación de plazos locales de planificación para LC-EDF aplicados en los distintos grupos de sistemas con el resto de variaciones generadas. Es importante recordar que la carga en estos grupos se genera con el algoritmo *UUnifast*. Los resultados observados en estas figuras vuelven a representar la misma escena, HOSPA y PD poseen una reducida ventaja sobre los algoritmos más triviales UD y ED. Cabe reseñar los resultados del grupo P-10 (sistemas con 10 procesadores), para el que ED es ligeramente superior a HOSPA.

A la vista de los resultados aquí presentados, se puede concluir que el tipo de carga de los sistemas tiene una gran influencia en el rendimiento de los distintos algoritmos de asignación de plazos locales de planificación para LC-EDF.

Para sistemas con utilizaciones en las actividades (C_i/T_i) variadas, como las generadas con *UUnifast*, los algoritmos que mejores resultados obtienen en general son PD y HOSPA. El rendimiento de HOSPA proviene principalmente de su asignación inicial, ya que apenas es capaz de mejorarla.

Si todas las actividades en cada procesador poseen la misma utilización, como las cargas generadas con *SCALE-WCET*, el algoritmo con mejor rendimiento observado fue ED, seguido de UD. Hay que recordar la trivialidad de estos algoritmos, en especial UD, en el que todas las actividades de cada flujo poseen el mismo plazo de planificación.

El hallazgo de que existen situaciones en las que UD es mejor algoritmo que HOSPA o PD es un resultado totalmente inesperado y sorprendente. Esta anomalía, junto a las bajas utilizaciones máximas planificables alcanzadas en LC-EDF, nos motiva a profundizar en la búsqueda de las causas, con el objetivo de buscar también nuevas técnicas que puedan mejorar el rendimiento de la planificación LC-EDF. Éste tema será desarrollado en el Capítulo 7.

6.5 Comparativa entre políticas de planificación y conclusiones

En el presente capítulo se llevó a cabo un estudio comparativo de técnicas de asignación de parámetros de planificación, en el que se analizó el rendimiento de éstas sobre un dominio de estudio que abarcaba un conjunto amplio de sistemas, haciendo uso de la herramienta GEN4MAST. Este conjunto de sistemas se centraba en la variación de dos características que en el Capítulo 5 comprobamos que tienen gran influencia en la planificabilidad de los sistemas distribuidos: los plazos de principio a fin, y la longitud de los flujos e2e. Este dominio de estudio generado quedó compuesto por 792000 sistemas, ocupando un total de 68 *gigabytes* de ficheros descriptores de sistemas.

Sobre el dominio de estudio generado se aplicaron las diferentes técnicas de asignación de parámetros de planificación disponibles: UD, ED, PD, NPD, EQS, EQF y HOSPA. Para el algoritmo HOSPA se llevó a cabo un estudio previo en el que se buscó la combinación de sus parámetros de configuración óptima. En total, la ejecución del estudio (incluyendo el estudio previo para HOSPA) requirió aproximadamente 800 días de tiempo de CPU, que repartidos en 250 procesadores del supercomputador Tres Mares, se pudo completar en 5 días. Hay que tener en cuenta que durante todo el transcurso de la ejecución no se dispuso de los 250 procesadores en todo momento, de ahí que el tiempo

real de ejecución haya sido superior al teórico. Este estudio se llevó a cabo en una época de mayor utilización en el supercomputador Tres Mares, por lo que el número de procesadores utilizados se redujo de los 300 disponibles para llevar a cabo el estudio del Capítulo 5, a un máximo de 250.

Para cada política de planificación (FP, GC-EDF y LC-EDF) se determinó qué técnica de asignación de parámetros de planificación es la que mejores resultados obtiene en cada circunstancia probada. A modo de resumen, ahora utilizaremos estos resultados para presentar una comparación cruzada del rendimiento de las distintas políticas de planificación, utilizando en cada caso la técnica de asignación que mejores resultados obtuvo. Para FP identificamos que HOSPA obtiene un mejor rendimiento para plazos de principio a fin más bajos, mientras que EQF es la mejor técnica en promedio para el resto de situaciones. En GC-EDF se comprobó que HOSPA es la técnica que en promedio obtiene los mejores resultados en todo el rango de sistemas estudiado. Por último, en LC-EDF se observó que ED es la técnica con mejores resultados con sistemas con cargas calculadas con *SCALE-WCET*, mientras que para cargas calculadas con *UUnifast,* la mejor técnica es HOSPA.

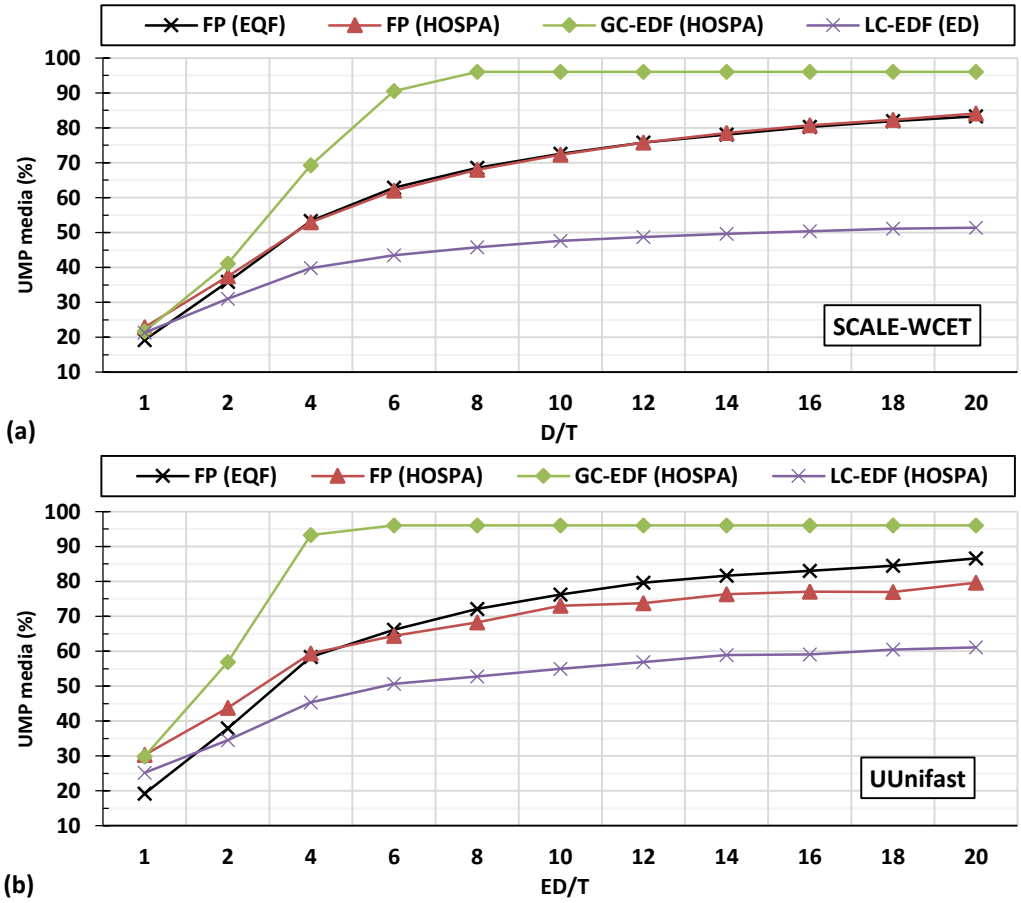


Figura 6-22 UMP media obtenida por las distintas políticas de planificación, para distintos plazos de principio a fin y cargas generadas con **(a) SCALE-WCET y (b) UUnifast**

En la Figura 6-22 se comparan las UMP medias obtenidas por las diferentes políticas de planificación en función de los plazos de principio a fin, utilizando en cada política la técnica de asignación que en promedio obtiene los mejores resultados. Para plazos de principio a fin pequeños, las diferencias entre las distintas políticas son reducidas. A medida que aumentan los plazos de principio a fin, GC-EDF mejora su capacidad de planificación de manera mucho más acelerada que las otras dos políticas de planificación.

El resultado de GC-EDF concuerda con el funcionamiento de EDF en sistemas monoprocesadores, en los que se demuestra que cuando las actividades son independientes y poseen plazos iguales a los periodos, el sistema es planificable hasta el 100% de utilización. LC-EDF posee un rendimiento marcadamente inferior a GC-EDF, con diferencias de hasta un 50% en la UMP media. FP se mantiene en un punto intermedio de rendimiento entre GC-EDF y LC-EDF.

En la Figura 6-23 se muestran las UMP medias obtenidas por las diferentes políticas de planificación para un rango de longitudes de los flujos e2e entre 4 y 20. Se repite la escena representada en la figura anterior, GC-EDF es la política que mayores utilizaciones alcanza, seguida por FP, y finalmente por LC-EDF. La ventaja de GC-EDF con el resto de políticas se acrecienta a medida que aumenta la longitud de los flujos e2e.

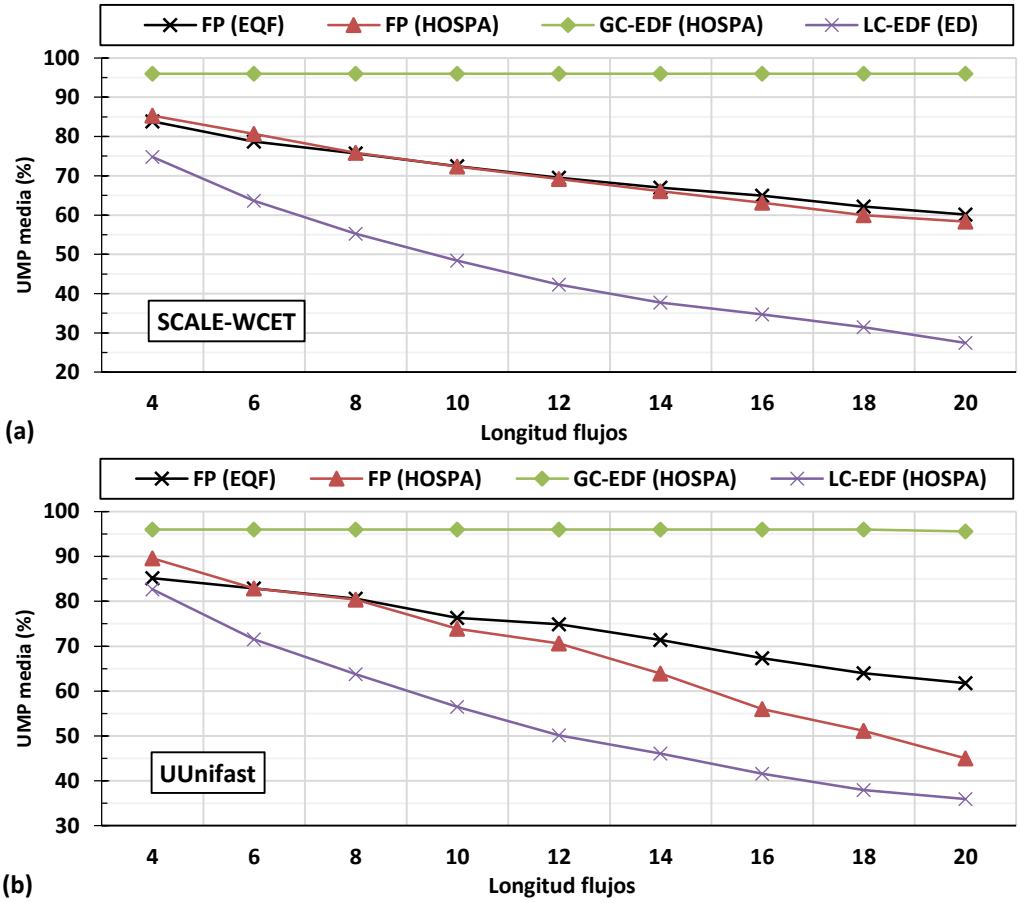


Figura 6-23 UMP media obtenida por las distintas políticas de planificación, para distintas longitudes de los flujos, y cargas generadas con **(a) SCALE-WCET y (b) UUnifast**

Cuando variamos el resto de características de los sistemas (Figura 6-24 y Figura 6-25) no se observan grandes diferencias en el orden de rendimiento ya establecido entre las diferentes políticas. Las distancias con GC-EDF se reducen en aquellas circunstancias menos adversas para la planificabilidad (flujos más cortos y rango de los periodos más amplio), pero esta reducción se debe principalmente a que GC-EDF no tiene margen posible a la mejora, ya que la UMP media se satura en el valor límite del estudio (el 96%).

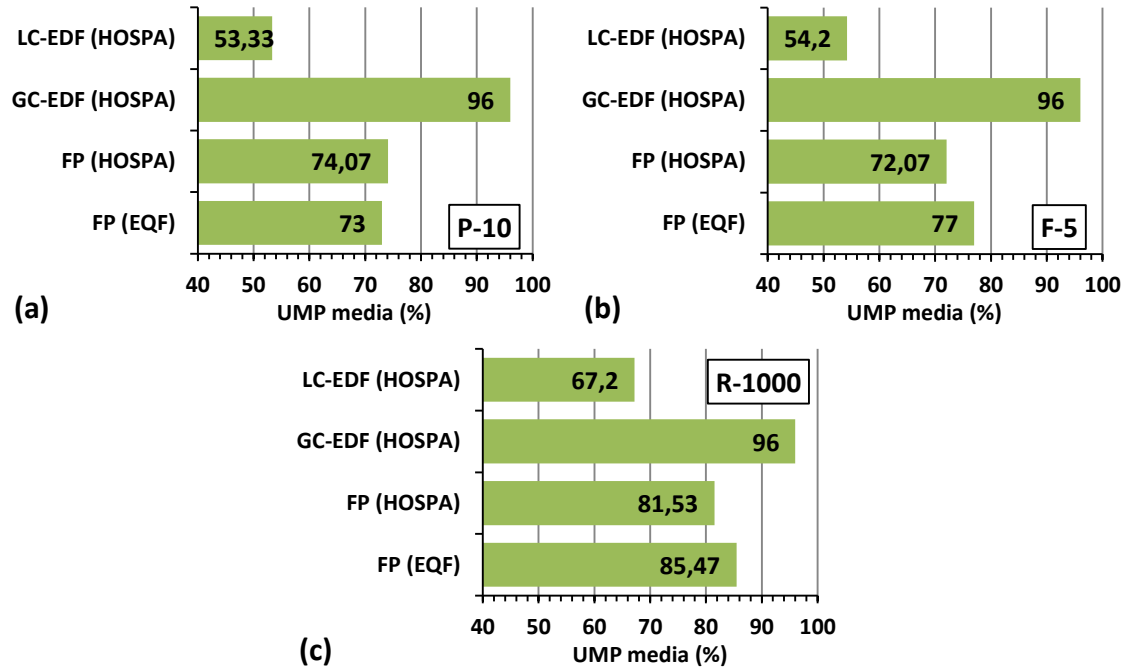


Figura 6-24 UMP media obtenida por las distintas políticas de planificación, para los grupos de sistemas **(a) P-10, (b) F-5, y (c) R-1000**

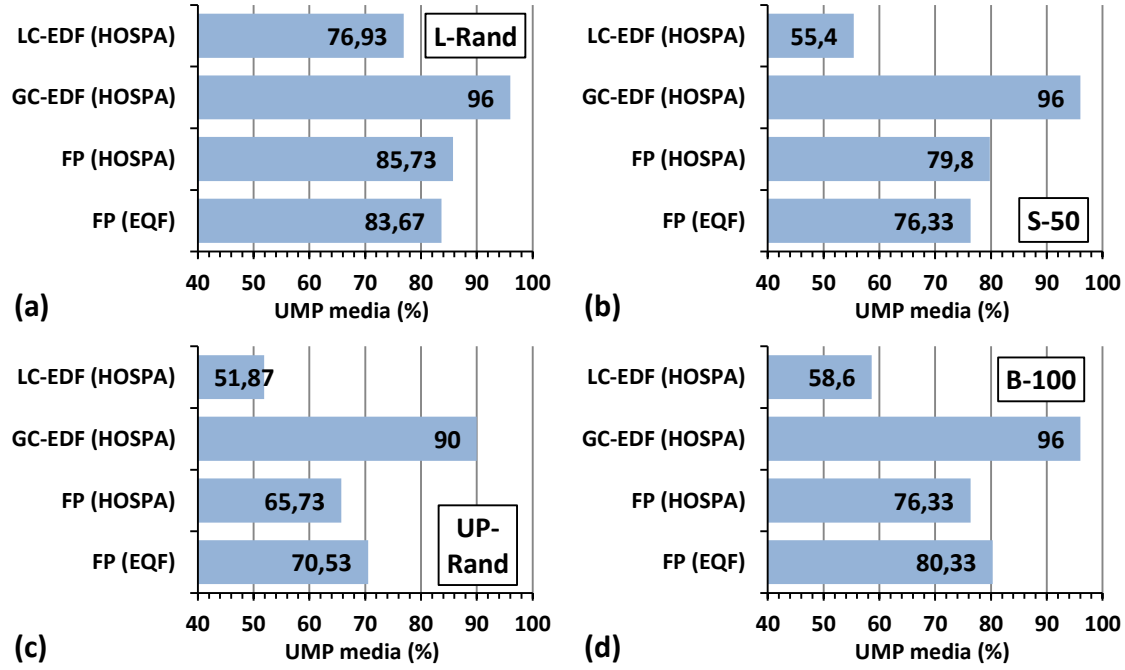


Figura 6-25 UMP media obtenida por las distintas políticas de planificación, para los grupos de sistemas **(a) L-Rand, (b) S-50, (c) UP-Rand, y (d) B-100**

7 Asignación de Plazos Locales de Planificación

En sistemas monoprocesadores con actividades independientes, la principal virtud de la planificación EDF es su capacidad de planificar sistemas con utilizaciones mayores que en FP. En sistemas distribuidos, esta virtud se ha observado en los Capítulos 5 y 6 con planificación GC-EDF, pero no así con LC-EDF, que utilizando las técnicas de asignación de plazos locales de planificación actuales tiene un rendimiento marcadamente inferior a GC-EDF, y aunque en menor medida, también inferior a FP. A la vista de los resultados que hemos expuesto, al diseñador de un sistema distribuido que quiera utilizar planificación EDF se le presentan dos opciones, (1) disponer de una solución para la sincronización de los relojes de los distintos recursos procesadores, asumiendo las posibles dificultades y sobrecargas añadidas al sistema, lo que le permitiría usar una planificación GC-EDF con los beneficios que conlleva, o bien, (2) utilizar los relojes locales de cada recurso procesador, haciendo que el sistema sea más sencillo, pero al tener que usar LC-EDF se puede perder hasta un 50% en la capacidad de cómputo tal como hemos mostrado en el capítulo anterior. Así pues, encontrar maneras de aumentar el rendimiento de LC-EDF sería de gran ayuda para tales situaciones en las que no se dispone de un mecanismo de sincronización de relojes, o no se quiere utilizar por simplificar la implementación.

En la comparativa de técnicas de asignación de plazos locales de planificación para LC-EDF llevada a cabo en el capítulo anterior, se mostraron unos resultados que parecen indicar que los algoritmos de asignación de plazos locales existentes tienen un funcionamiento deficiente y claramente mejorable. Este indicio se pudo observar en los resultados producidos por los algoritmos triviales UD y ED, que a pesar de no estar indicados para LC-EDF, son capaces de alcanzar utilizaciones planificables cercanas y en algunos casos superiores al algoritmo heurístico HOSPA, que tan buenos resultados obtiene en FP y GC-EDF. Resultó especialmente llamativo comprobar cómo UD obtuvo

mejores resultados que HOSPA para sistemas con utilizaciones de las actividades uniformes (carga generada con *SCALE-WCET*).

El hecho de que asignaciones de plazos que contravienen la lógica, al no respetar los plazos de principio a fin impuestos, obtengan unos resultados mejores que las asignaciones que sí los respetan como HOSPA, nos indica que existe algún tipo de problema en las suposiciones que realizamos al definir los plazos locales de planificación en un sistema distribuido. Este fenómeno inesperado nos lleva a indagar en las causas que lo producen con el objetivo de caracterizarlo y buscar posibles mejoras en las técnicas de asignación de plazos locales de planificación. Como primer paso, en la siguiente sección estudiaremos más detalladamente el comportamiento de LC-EDF.

7.1 Análisis del comportamiento de LC-EDF

La principal diferencia entre los plazos locales asignados por UD/ED y los asignados por PD/NPD/HOSPA es que los valores obtenidos por UD/ED son siempre mayores (mucho mayores en la mayoría de los casos). La intuición nos dice que un plazo de planificación más pequeño implica una prioridad mayor y debería producir un tiempo de respuesta de peor caso menor . Por otra parte, un aumento proporcional de los plazos de planificación de todas las actividades debería ser equivalente a una disminución proporcional de las prioridades, por lo que los tiempos de respuesta de peor caso no deberían cambiar. Sin embargo, tal y como se observó en el capítulo anterior, esto no ocurre así en LC-EDF, ya que una relajación de los plazos hace que los tiempos de respuesta de peor caso disminuyan. Este fenómeno contrario a la intuición se podría resumir en la expresión “vísteme despacio que tengo prisa”.

Otra característica que diferencia a UD/ED con respecto a PD/NPD/HOSPA es que dan lugar a una asignación de plazo de planificación cuya suma para un flujo e2e es mayor que el requisito temporal de principio a fin que tiene que cumplir, es decir, no cumplen con la Ecuación (2-31). La asignación de plazos locales de planificación conformes con los plazos de principio a fin surge de la idea intuitiva de repartir el requisito total del flujo e2e entre sus actividades, de forma que el cumplimiento de cada uno de los plazos locales implique el cumplimiento del plazo global del flujo. Esta idea intuitiva es utilizada habitualmente en los trabajos sobre flujos distribuidos [SER10]. Sin embargo, a la vista de los resultados observados en la sección anterior, restringir los plazos locales de planificación a valores conformes al plazo de principio a fin puede deteriorar el funcionamiento de la planificación LC-EDF.

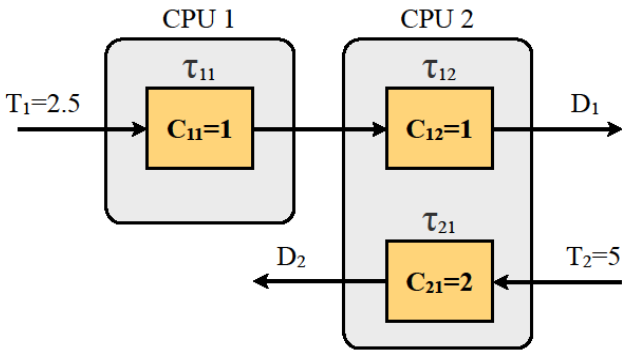


Figura 7-1 Ejemplo sencillo de 3 actividades para estudiar el comportamiento de LC-EDF

Para comprender las razones de este comportamiento anómalo, y con el objetivo final de aportar mejoras a las técnicas actuales de asignación de plazos locales de planificación, en primer lugar estudiaremos el comportamiento de LC-EDF sobre un ejemplo sencillo. Este ejemplo está compuesto por 3 actividades alojadas en dos procesadores y organizadas en 2 flujos e2e. El ejemplo queda ilustrado en la Figura 7-1, en la que se incluyen los valores de los periodos y los tiempos de ejecución de peor caso de las actividades. La actividad sobre la que nos centraremos es τ_{12}. Para este ejemplo estudiaremos el efecto de tener dos tipos distintos de plazos de planificación, unos mayores que los otros. Para ello creamos dos escenarios con plazos de principio a fin distintos, Escenario-1 y Escenario-2, de manera que los plazos del segundo son el doble de los del primero. Los valores concretos de estos plazos de principio a fin se muestran en la Tabla 7-1.

	Escenario-1	**Escenario-2**
D_1	5	10
D_2	5	10

Tabla 7-1 Plazos de principio a fin de los escenarios del ejemplo sencillo de 3 actividades

En el análisis de planificabilidad, los *jitters* dependen del tiempo de respuesta de peor caso de la actividad precedente en el flujo e2e, y los tiempos de respuesta dependen de los *jitters*. Para conseguir tener las mismas condiciones de *jitter* inicial en la actividad τ_{12} en ambos escenarios, se localizó su actividad precedente τ_{11} en un procesador sin ninguna otra interferencia de otras actividades.

Por simplicidad, aplicamos una asignación PD de plazos locales de planificación en ambos escenarios, y posteriormente utilizamos la técnica de análisis holística para el cálculo de los tiempos de respuesta de peor caso. Los resultados obtenidos se muestran en la Tabla 7-2 para ambos escenarios. Como puede comprobarse, unos plazos de planificación más grandes han repercutido en un tiempo de respuesta de peor caso más pequeño para la actividad τ_{12}.

	Escenario -1		**Escenario -2**	
	Sd_{ij}	R_{ij}	Sd_{ij}	R_{ij}
τ_{11}	2,5	1	5	1
τ_{12}	2.5	**2.5**	5	**2**
τ_{21}	5	4	10	4

Tabla 7-2 Tiempos de respuesta de peor caso para el ejemplo sencillo de 3 actividades en LC-EDF

Para una comprensión mejor de este problema, vamos a representar el diagrama temporal de la situación que conduce al tiempo de respuesta de peor caso en ambos

escenarios. Estos diagramas temporales quedan representadas en la Figura 7-2 y en la Figura 7-3 para el Escenario-1 y Escenario 2 respectivamente. En esta representación, t_c es el comienzo del periodo de ocupación, las flechas inferiores indican cada una de las activaciones del flujo e2e (cada una etiquetada con un valor p secuencial), y t_d es el plazo absoluto de cada activación.

La situación de peor caso en LC-EDF se construye de acuerdo al teorema 1 en [RIV10]: Cada actividad τ_{ij} diferente de la actividad bajo análisis τ_{ab} es activada en el comienzo del periodo de ocupación, o en un instante en el cual su plazo absoluto coincide con el de otra actividad.

La situación de peor caso de la actividad τ_{12} en el Escenario-1 se representa en la Figura 7-2, donde el *jitter* de la actividad τ_{12} (J_{12}) es 1. Como puede observarse, el tiempo de respuesta de peor caso se obtiene cuando su activación se sitúa en el comienzo del periodo de ocupación. En esta situación, su tiempo de respuesta de peor caso se encuentra en su segunda activación (p=2). Este tiempo de respuesta es 2.5, que es la diferencia entre el instante de activación (t=1,5) y el instante de finalización (t=4).

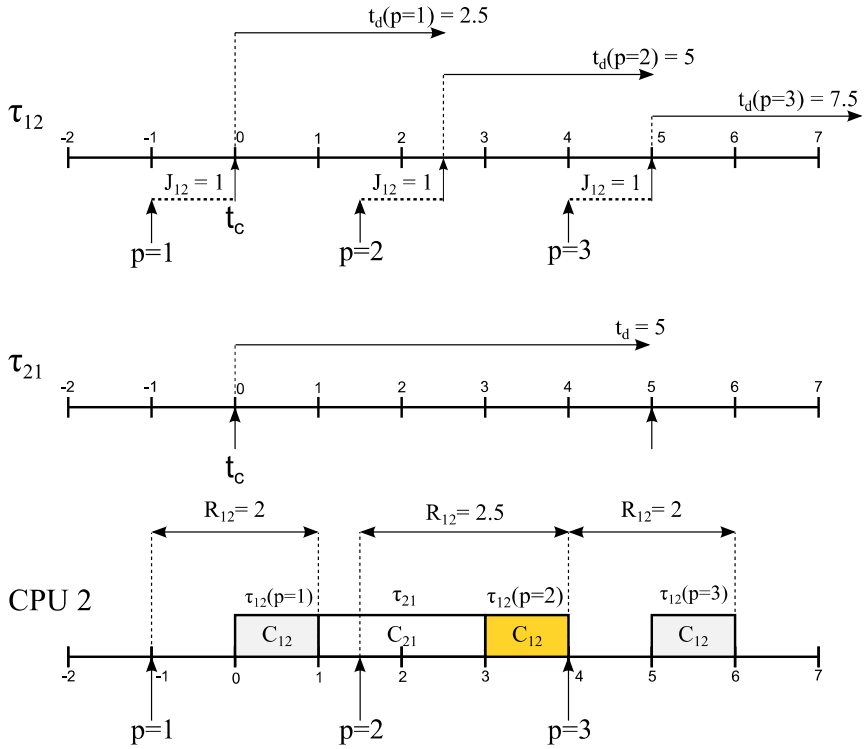


Figura 7-2 Línea de tiempos para la obtención de los tiempos de respuesta de peor caso de la actividad τ_{12} en el Escenario-1

En la Figura 7-3 se muestra la situación de peor caso de la actividad τ_{12} en el Escenario-2, con el mismo valor de *jitter* J_{12} igual a 1. Su tiempo de respuesta de peor caso también ocurre cuando se activa al comienzo del periodo de ocupación, pero en esta ocasión, la peor activación es la primera (p=1), con un tiempo de respuesta R_{12}=2.

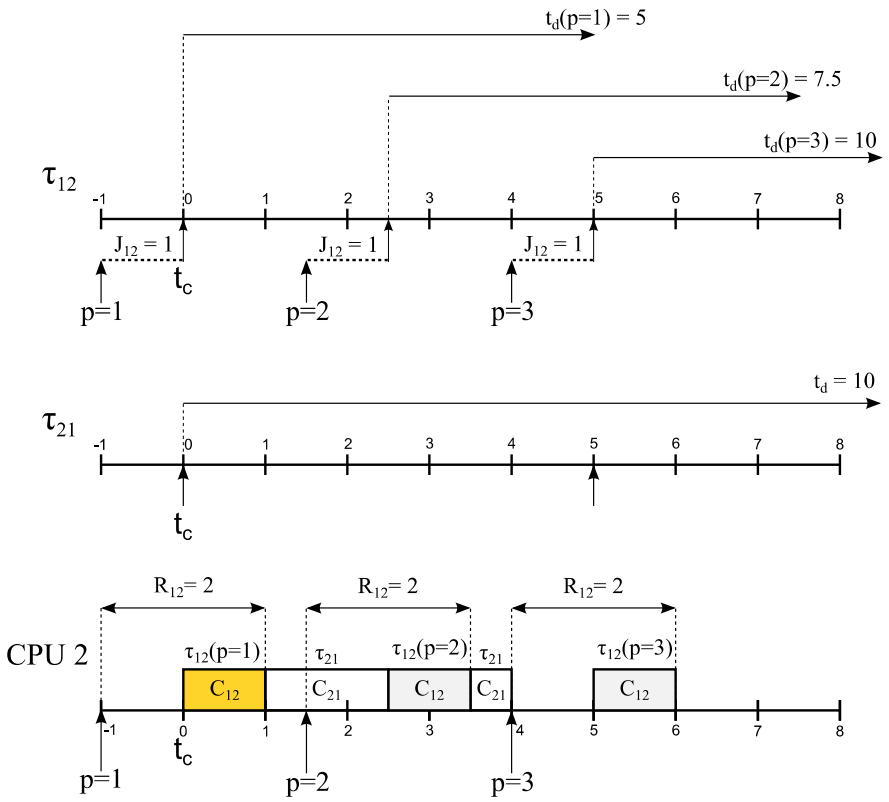


Figura 7-3 Línea de tiempos para la obtención de los tiempos de respuesta de peor caso de la actividad τ_{12} en el Escenario-2

Cuando el tiempo de respuesta de peor caso se encuentra en una activación posterior a la primera dentro del periodo de ocupación, aumentar los plazos de planificación puede mover esta situación de peor caso a la primera activación, con una posible reducción de los tiempos de respuesta. En el Escenario-1, el tiempo de respuesta de peor caso R_{12} ocurre en la segunda activación, con un plazo absoluto t_d=5, que coincide con el plazo absoluto de la actividad τ_{21}, por lo que esta actividad contribuye a aumentar el tiempo de respuesta de peor caso de la actividad bajo análisis τ_{12}. En el Escenario-2 los plazos se duplican, y por lo tanto Sd_{12}=5 y Sd_{21}=10. Sin embargo, el plazo absoluto de la segunda activación de τ_{12} se convierte en t_d=7.5, mientras que el plazo absoluto de τ_{21} es t_d=10, eliminando por lo tanto la interferencia de τ_{21} en el tiempo de respuesta de la segunda activación de τ_{12}.

Una explicación intuitiva de este comportamiento puede ser que distribuir los plazos de principio a fin entre sus actividades es prácticamente equivalente a imponer unos requisitos más estrictos. Mientras que esta asignación no es importante en sistemas con planificación FP, en los que estos plazos se transforman posteriormente en valores discretos con los que ordenar las actividades (prioridades fijas), en LC-EDF implica que el planificador posee menos tiempo para acomodar la ejecución de cada actividad. En LC-EDF no cumplir con los plazos locales no implica la no planificabilidad del sistema, puesto que son plazos para la planificación y no requisitos de tiempo, es decir, no son más que números para establecer un orden de ejecución, como las prioridades fijas en FP. Sin embargo, ajustar estos plazos en exceso puede acarrear problemas para el planificador a la hora de ordenar la ejecución de las actividades.

Para completar la discusión, en la Tabla 7-3 se muestran los tiempos de respuesta de peor caso de ambos escenarios cuando se utiliza planificación GC-EDF. Como se puede

observar, los tiempos de respuesta no se ven modificados al aumentar los plazos en el Escenario-2.

	Escenario-1		**Escenario-2**	
	SD_{ij}	R_{ij}	SD_{ij}	R_{ij}
τ_{11}	2.5	1	5	1
τ_{12}	5	**3**	10	**3**
τ_{21}	5	3	10	3

Tabla 7-3 Tiempos de respuesta de peor caso para el ejemplo sencillo de 3 actividades en GC-EDF

El ejemplo sencillo de 3 actividades sólo muestra un ligero cambio en los tiempos de respuesta de peor caso al cambiar a un escenario con plazos duplicados. Para mostrar que la mejora en los tiempos de respuesta puede ser más significativa, incluimos otro ejemplo en el que añadimos una actividad adicional τ_{22}, tal y como se describe en la Figura 7-4. Esta nueva actividad inducirá un aumento general del *jitter*, y consecuentemente, de los tiempos de respuesta de peor caso.

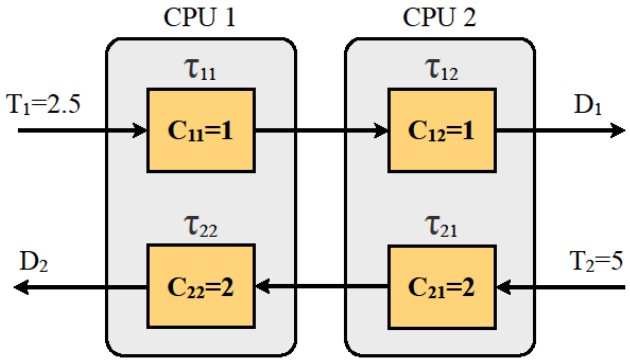


Figura 7-4 Ejemplo sencillo de 4 actividades para estudiar el comportamiento de LC-EDF

Como en el ejemplo anterior, creamos dos escenarios, de manera que los plazos de principio a fin del segundo escenario son el doble que los del primero. Los valores de los plazos se especifican en la Tabla 7-4.

	Escenario-1	**Escenario-2**
D_1	5	10
D_2	10	20

Tabla 7-4 Plazos de principio a fin de los escenarios del ejemplo sencillo de 4 actividades

Los resultados de aplicar una asignación local PD y el análisis holístico se resumen en la Tabla 7-5. En este ejemplo, la reducción de los tiempos de respuesta de peor caso se producen en todas las actividades, y de forma más patente que en el ejemplo con 3 actividades.

	Scn-1		**Scn-2**	
	Sdij	*Rij*	*Sdij*	*Rij*
τ_{11}	2.5	**3.5**	5	**1**
τ_{12}	2.5	**5**	5	**2**
τ_{21}	5	**5**	10	**4**
τ_{22}	5	**9**	10	**8**

Tabla 7-5 Tiempos de respuesta de peor caso para el ejemplo sencillo de 4 actividades en LC-EDF

Como se ha observado con estos ejemplos sencillos, aumentar los plazos de planificación puede conllevar una reducción de los tiempos de respuesta de peor caso, aumentando por lo tanto la planificabilidad del sistema. Este resultado concuerda con lo observado en el Capítulo 6 con las técnicas de asignación de parámetros de planificación UD y ED, que en ciertas circunstancias obtienen mejores resultados que PD y HOSPA. Aunque no se puede concluir que este comportamiento sea general en todos los casos, estos resultados nos motivan para estudiar nuevas técnicas de asignación de plazos locales que intenten aprovecharse de este fenómeno, rompiendo con la noción de que los plazos locales de planificación deban ser conformes al plazo de principio a fin.

7.2 Exploración de nuevas técnicas de asignación de plazos locales de planificación

En el punto anterior se comprobó que existen situaciones en las que aumentar los plazos locales de planificación produce una reducción generalizada de los tiempos de respuesta de peor caso. Aunque esta situación no puede afirmarse que se produzca en todos los casos, los resultados inesperados obtenidos con UD y ED nos podrían dar una indicación de que en promedio el aumento de los plazos induce una mejora en la planificabilidad de sistemas LC-EDF.

Con objeto de mejorar el rendimiento de las técnicas que producen asignaciones de plazos locales de planificación conformes a los plazos de principio a fin (PD/NPD/HOSPA), se van a proponer dos modificaciones que incumplen esta regla:

- Escalado de plazos (LC-EDF-DS, *Local-Clock EDF with Deadline Scaling*)
- LC-EDF con plazos globales de planificación (LC-EDF-GSD, *Local-Clock EDF with Global Scheduling Deadlines*)

Estas modificaciones se pueden aplicar sobre las técnicas existentes, e intentan mejorar el rendimiento en LC-EDF rompiendo con la idea de que los plazos locales de planificación deben ser conformes al plazo de principio a fin.

Al igual que en Capítulo 6, como figura de mérito para evaluar el rendimiento de las diferentes técnicas de asignación de parámetros de planificación usaremos la utilización máxima planificable (UMP) media. Para observar las diferencias con mayor claridad, también mostraremos las UMP medias tomando como referencia una técnica concreta. Así, si tomamos como referencia la técnica de asignación de parámetros de planificación X, definimos ΔUMP(X) como sigue:

$$\Delta UMP(\mathrm{X}) = \mathrm{UMP\ media\ (A)} - \mathrm{UMP\ media\ (X)} \qquad (7\text{-}1)$$

donde, A representa a cualquier técnica de asignación de parámetros de planificación, y X a la técnica de referencia.

7.2.1 Escalado de plazos (LC-EDF-DS)

LC-EDF-DS consiste en escalar los plazos de principio a fin de los flujos e2e por un factor *k*, sólo a efectos de la asignación de plazos de planificación. Como los plazos de planificación surgen del reparto del plazo de principio a fin, este escalado produce un aumento generalizado de los plazos de planificación. Es importante insistir en que LC-EDF-DS no cambia el plazo de principio a fin como requisito temporal.

Para los algoritmos PD/NPD/HOSPA, o cualquier otro conforme al plazo de principio a fin, aplicar LC-EDF-DS produce una nueva asignación de plazos que cumple con la siguiente ecuación:

$$\sum_{j=1}^{N_i} Sd_{ij} = k \cdot D_i \tag{7-2}$$

Para poder estudiar el efecto de aplicar LC-EDF-DS, lo añadimos como opción de ejecución en MAST y GEN4MAST. El parámetro de generación en GEN4MAST es "SCALE_FACTOR", incluido en la sección "EXECUTION" del fichero de configuración de GEN4MAST. Sus valores especifican los factores de escalado *k* a aplicar. La sintaxis se muestra en la Figura 7-5.

```
[EXECUTION]
SCALE_FACTOR k1 …
[/EXECUTION]
```

Figura 7-5 Sintaxis del parámetro SCALE_FACTOR en GEN4MAST para aplicar LC-EDF-DS

Utilizamos GEN4MAST para estudiar el efecto de aplicar LC-EDF-DS, ejecutando las técnicas de asignación para LC-EDF con distintos factores de escala *k*. Este estudio lo llevamos a cabo en el dominio generado en el Capítulo 6 (estudio comparativo de técnicas de asignación de parámetros de planificación). Como se comprobó en dicho capítulo, el tipo de carga del sistema tiene una gran influencia en los resultados en LC-EDF. Por esta razón estudiaremos el efecto de LC-EDF-DS por separado con cargas *SCALE-WCET* y *UUnifast*. Denotamos con el sufijo "-DS" (por ejemplo HOSPA-DS) a las técnicas de asignación utilizadas en conjunción con LC-EDF-DS.

En la Figura 7-6(a) se representan las UMP medias obtenidas al aplicar LC-EDF-DS a las distintas técnicas de asignación con diferentes factores de escala *k*, para los sistemas base (con $D_i=N_i*T_i$) y cargas *SCALE-WCET*. En la figura se comprueba el efecto beneficioso de aplicar LC-EDF-DS en PD y HOSPA, registrándose mejoras de hasta un 9% para k=20. Para ED y UD, aplicar LC-EDF-DS no produce cambios apreciables en las UMP medias. A pesar de las mejoras registradas con PD-DS y HOSPA-DS, ED (con o sin LC-EDF-DS) sigue siendo la técnica con mejor rendimiento en este tipo de sistemas.

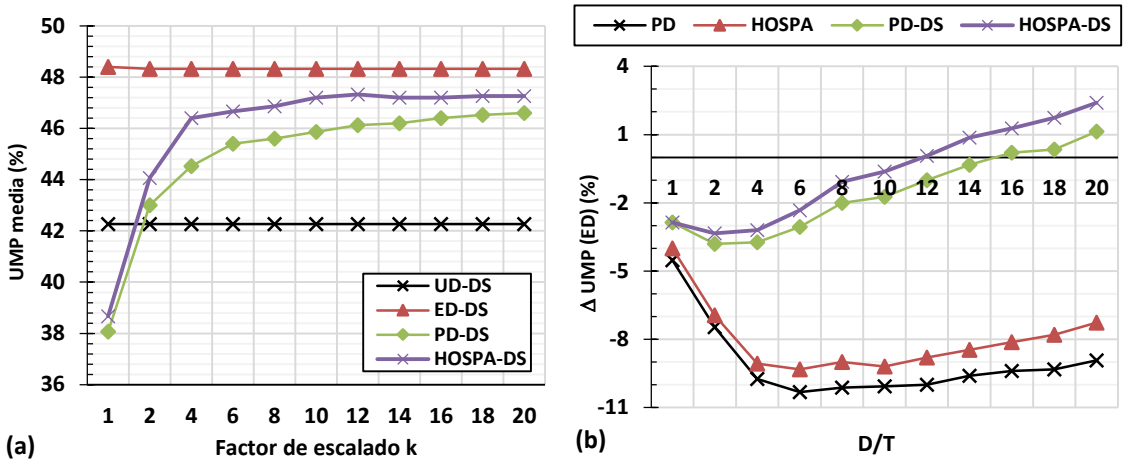


Figura 7-6 Rendimiento de LC-EDF-DS para sistemas con cargas generadas con *SCALE-WCET*. **(a) UMP media con distintos valores de *k*. (b) Para *k*=20, la ΔUMP(ED) de PD, HOSPA, PD-DS y HOSPA-DS**.

En la Figura 7-6(b) se representa la ΔUMP(ED) (definido en la Ecuación (7-4)) de los algoritmos PD, HOSPA, PD-DS y HOSPA-DS. Un valor de ΔUMP(ED) negativo indica que ED obtiene una UMP media superior. Se elige ED como referencia por ser la técnica que mejores resultados obtenía en LC-EDF para cargas *SCALE-WCET*. Los casos con LC-EDF-DS se ejecutan con un factor *k*=20. En primer lugar se comprueba que HOSPA-DS y PD-DS mejoran a la técnica ED en los sistemas con los plazos de principio a fin mayores ($ED_i/T_i>10$), aunque la mejora máxima es de sólo un 2.5% en la UMP media. La segunda conclusión relevante de esta figura es que aplicar LC-EDF-DS produce mejoras en PD y HOSPA para todo el rango de plazos de principio a fin estudiado.

A continuación seguimos el mismo procedimiento, pero para cargas generadas con *UUnifast*. En la Figura 7-7a se comprueba cómo aplicar LC-EDF-DS es también beneficioso para este tipo cargas y asignaciones PD y HOSPA. Las mejoras máximas se obtienen a partir de *k*=6 aproximadamente, estabilizándose para valores superiores.

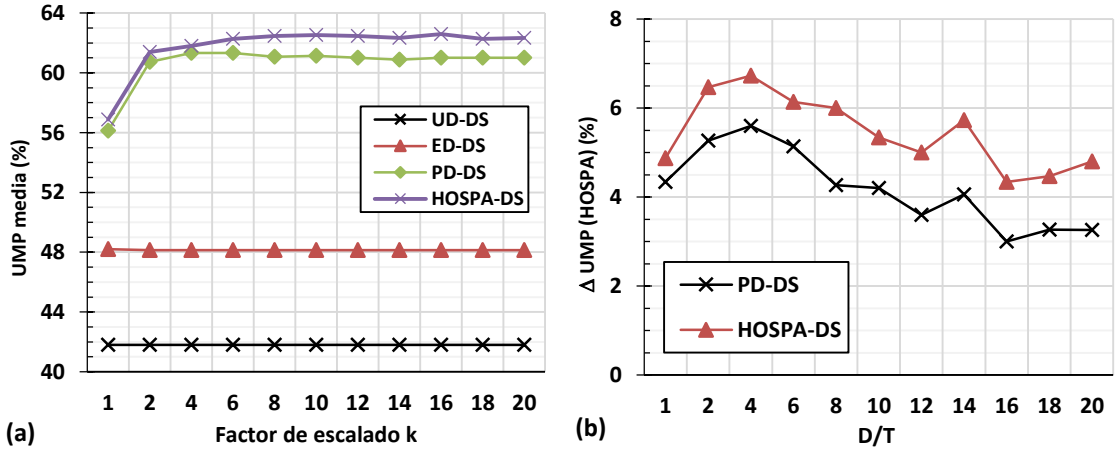


Figura 7-7 Rendimiento de LC-EDF-DS para sistemas con cargas generadas con *UUnifast*. **(a) UMP con distintos valores de *k*. (b) Para *k*=20, la ΔUMP(HOSPA) de PD-DS y HOSPA-DS**

En la Figura 7-7(b) se representa la ΔUMP(HOSPA) de las técnicas PD-DS y HOSPA-DS, utilizando un factor de escalado *k*=20. Se toma HOSPA sin DS como referencia, ya que es la técnica que previamente obtenía los mejores resultados en sistemas con carga *UUnifast*. En la figura se comprueba que HOSPA-DS mejora siempre los resultados obtenidos por HOSPA sin DS, superándolo hasta en un 7%. Incluso la

técnica no iterativa PD-DS es capaz de adelantar a HOSPA en todo el rango de plazos de principio a fin estudiado.

Hemos mostrado cómo la técnica del escalado de plazos LC-EDF-DS produce mejoras en los resultados de las técnicas de asignación de plazos locales de planificación PD y HOSPA. Esta mejora se produce en todo el rango de plazos de principio a fin estudiado. A pesar de este incremento en el rendimiento observado, para sistemas con cargas generadas con *SCALE-WCET* la técnica ED sigue siendo la que mejores resultados obtiene en la mayoría de los casos. Esto puede ser un indicativo de que aún se puede extraer más rendimiento en LC-EDF estudiando otras modificaciones.

7.2.2 LC-EDF con plazos globales (LC-EDF-GSD)

La segunda propuesta para mejorar las asignaciones de plazos locales es LC-EDF-GSD, y consiste en utilizar los valores de los plazos de planificación globales como plazos locales de planificación.

En PD/NPD/HOSPA, los plazos globales se calculan como la suma de los plazos locales precedentes (incluyéndose a sí mismo) en el flujo e2e. Por lo tanto es inmediato comprobar que utilizando los plazos globales estamos aumentando de forma general el valor de los plazos de planificación con respecto a una asignación LC-EDF tradicional. Con el fin de formalizar esta propuesta, en PD/NPD/HOSPA, u otra asignación local conforme al plazo de principio a fin, el criterio LC-EDF-GSD, o simplemente GSD, consiste en asignar los plazos de planificación según la siguiente ecuación:

$$Sd_{ij}^{GSD} = SD_{ij} = \sum_{k=1}^{j} Sd_{ik} \qquad (7\text{-}3)$$

Aparte de transformar asignaciones locales en globales y utilizarlas como plazos locales de planificación, también vamos a comprobar cómo se comportan en LC-EDF las técnicas que directamente producen asignaciones globales, como EQF o EQS.

Con el fin de estudiar el efecto de aplicar GSD, lo añadimos como opción de ejecución en MAST y GEN4MAST. El parámetro de generación en GEN4MAST es "FORCE_GLOBAL", que es un booleano que si se establece a valor True, indica que se deben asignar plazos de planificación con los valores globales en LC-EDF. La sintaxis se muestra en la Figura 7-10.

```
[EXECUTION]
FORCE_GLOBAL True/False …
[/EXECUTION]
```

Figura 7-8 Sintaxis del parámetro FORCE_GLOBAL en GEN4MAST para aplicar LC-EDF-GSD

Para estudiar el rendimiento de LC-EDF-GSD aplicamos PD y HOSPA con GSD en el dominio del estudio propuesto en la Sección 6. Denotamos con el sufijo "-GSD" (por ejemplo HOSPA-GSD) a las técnicas de asignación utilizadas en conjunción con LC-EDF-GSD.

En primer lugar, en la Figura 7-9(a) se representa la ΔUMP(ED) de las asignaciones PD, HOSPA, PD-GSD, y HOSPA-GSD, en sistemas con cargas *SCALE-WCET*. Se elige

ED como referencia ya que es la técnica que previamente obtenía los mejores resultados para sistemas con este tipo de cargas. Es fácil comprobar que PD-GSD y HOSPA-GSD introducen un gran beneficio en la planificabilidad de los sistemas, con mejoras de hasta un 36% en la UMP media para $D_i/T_i=20$. Con estas mejoras, HOSPA-GSD y PD-GSD superan ampliamente a ED en hasta un 29% y 27% en la UMP media respectivamente para $D_i/T_i=20$. Sin embargo, para los plazos más restrictivos ($D_i=T_i$), ED aún posee una ligera ventaja sobre HOSPA-GSD, de sólo un 1,3%. En la figura se concluye también que para cargas de tipo *SCALE-WCET*, LC-EDF-GSD obtiene mayores mejoras que LC-EDF-DS, en todo el rango de plazos de principio a fin.

En Figura 7-9(b) se representa la ΔUMP(HOSPA-GSD) de las asignaciones EQF y EQS, para comprobar si estas asignaciones mejoran al algoritmo HOSPA-GSD. En la figura se observa que para $D_i/T_i>4$, con EQS y EQF se obtienen unos resultados ligeramente superiores a HOSPA-GSD, siendo EQF la técnica con mayor rendimiento.

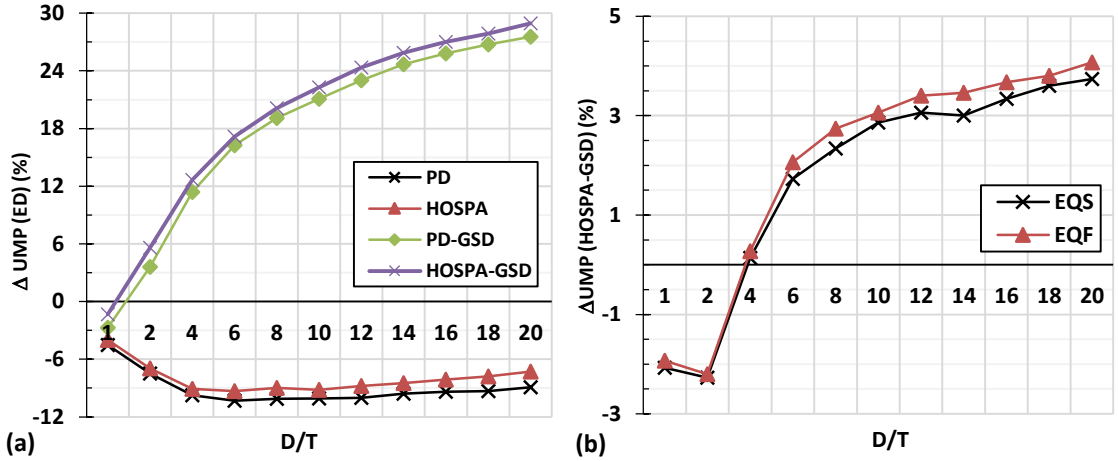


Figura 7-9 Rendimiento de LC-EDF-GSD para sistemas con cargas generadas con *SCALE-WCET*. **(a) ΔUMP(ED) de PD, HOSPA, PD-DS y HOSPA-GSD, y (b) ΔUMP(HOSPA-GSD) de EQS y EQF**.

A continuación repetimos el proceso para los sistemas con cargas *UUnifast*. En la Figura 7-10(a) se muestra la ΔUMP(HOSPA) de las técnicas PD-GSD y HOSPA-GSD. Se elige como referencia HOSPA ya que era la técnica que previamente obtenía los mejores resultados para este tipo de cargas. Se confirma que con GSD se obtiene una mejora clara de los resultados cuando los sistemas no poseen plazos de principio a fin restrictivos, con mejoras de hasta un 21% en la UMP media con HOSPA-GSD para $D_i/T_i=20$. Para $D_i=T_i$ sin embargo aplicar GSD introduce un claro deterioro en los resultados, de entre un 6% y 8% en la UMP media.

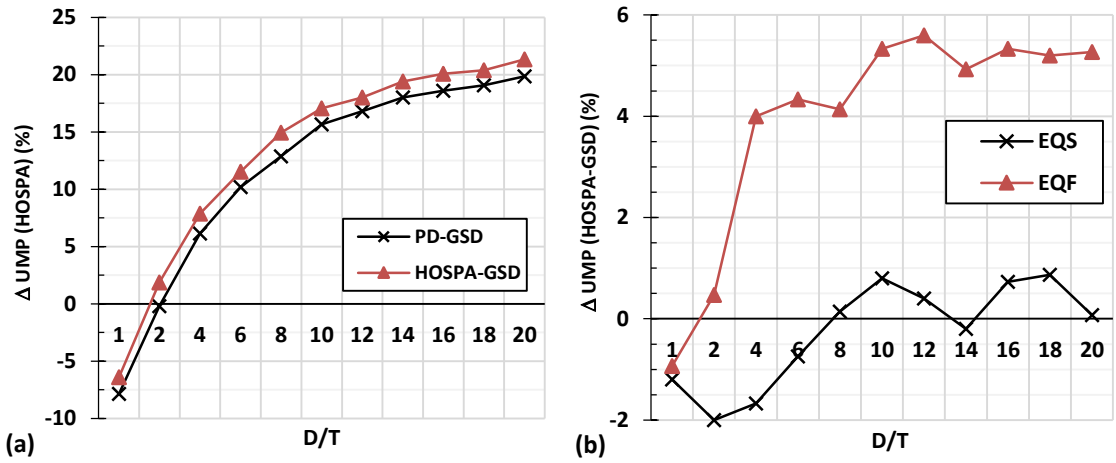


Figura 7-10 Rendimiento de LC-EDF-GSD para sistemas con cargas generadas con *UUnifast*. **(a) ΔUMP(HOSPA) de PD-GSD y HOSPA-GSD, y (b) ΔUMP(HOSPA-GSD) de EQS y EQF**.

En la Figura 7-10(b) se muestra la ΔUMP(HOSPA-GSD) de las técnicas EQS y EQF para cargas *UUnifast*. Se observa que EQS mantiene un rendimiento irregular en el rango de plazos de principio a fin, pero sin embargo, EQF posee una clara ventaja sobre HOSPA-GSD, con una diferencia máxima en torno al 5% en la UMP media, salvo para plazos $D_i=T_i$.

7.2.3 Combinación de LC-EDF-DS y LC-EDF-GSD

Se ha comprobado que con LC-EDF-GSD se pueden obtener mejoras superiores a las obtenidas con LC-EDF-DS. La aplicación de ambas modificaciones en las técnicas no es mutuamente excluyente. En esta sección se comprobará si aplicar LC-EDF-DS y LC-EDF-GSD simultáneamente produce algún beneficio adicional a aplicar sólo LC-EDF-GSD.

En los resultados mostraremos el incremento en la UMP media que se produce al aplicar la modificación LC-EDF-DS sobre PD-GSD, HOSPA-GSD, EQS y EQF. Así, análogamente a la definición de ΔUMP(X), definimos ΔUMP(DS) de la siguiente manera:

$$\Delta UMP(\mathrm{DS}) = \mathrm{UMP\ media\ (B)} - \mathrm{UMP\ media\ (B{-}DS)} \qquad (7\text{-}4)$$

donde B representa a cualquier técnica de asignación de parámetros de planificación (con o sin aplicación de modificación), y B-DS representa a la misma técnica a la que se aplica LC-EDF-DS.

En la Figura 7-11 se representa la ΔUMP(DS) de PD-GSD, HOSPA-GSD, EQS y EQF, para cargas *SCALE-WCET* y *UUnifast*. En ambos tipos de cargas se observa la misma tendencia. Así, para los plazos de principio a fin más pequeños, aplicar LC-EDF-DS sobre las asignaciones de plazos globales produce un reducido beneficio, situándose el máximo en torno al 1% en la UMP media. Sin embargo, en el resto del rango de plazos de principio a fin, combinar LC-EDF-DS con LC-EDF-GSD produce incluso un ligero empeoramiento de los resultados. Podemos concluir que añadir LC-EDF-DS a LC-EDF-GSD no produce mejoras significativas en general.

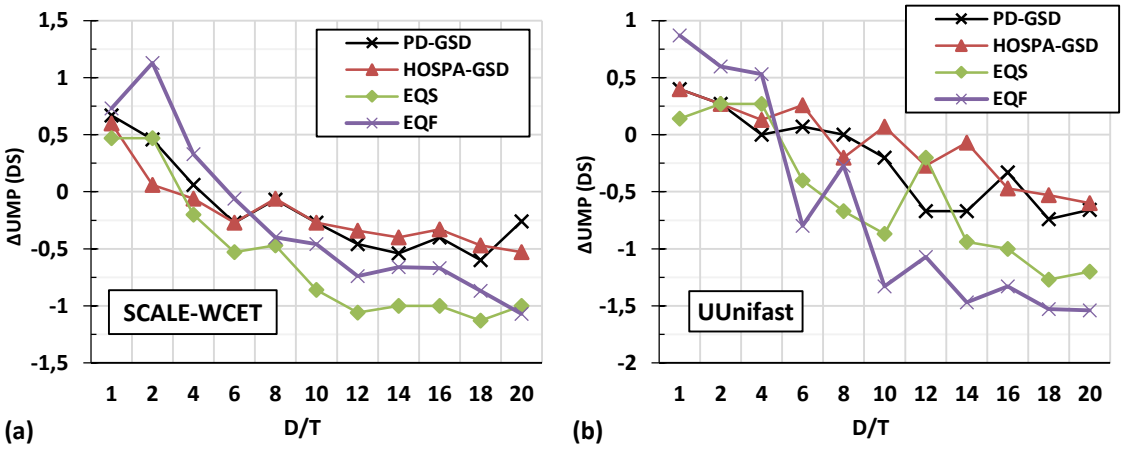


Figura 7-11 ΔUMP(DS+GSD, DS) de PD, HOSPA, EQS y EQF, para sistemas con cargas generadas **con (a) *SCALE-WCET* y (b) *UUnifast***.

7.3 Aplicación a un caso de estudio: sistema de control de vuelo

Se han mostrado los beneficios de aplicar LC-EDF-DS y LC-EDF-GSD sobre conjuntos sintéticos y masivos de ejemplos. Para validar estos resultados sobre un caso práctico, vamos a aplicar estas propuestas en un ejemplo real y representativo de un sistema de tiempo real distribuido, el sistema de control de vuelo presentado en la Sección 3.3 de esta tesis, que ha sido extraído de [JAY08]. A modo de recordatorio, el sistema está formado por 3 flujos e2e que recorren 8 recursos procesadores. Asumimos que el bus se modela como un procesador más.

Sobre este ejemplo aplicamos HOSPA bajo 4 situaciones distintas: GC-EDF, LC-EDF, LC-EDF-DS(k=20), y LC-EDF-GSD. En la Tabla 7-6 se muestran los tiempos de respuesta de peor caso de los tres flujos e2e en las 4 situaciones descritas. Como técnica de análisis de planificabilidad se utiliza el análisis holístico, ya que es el único disponible tanto para GC-EDF como LC-EDF. Se observa que los flujos e2e Γ_1 y Γ_3 cumplen su plazo de principio a fin en las 4 situaciones.

Por otro lado se comprueba que el flujo e2e Γ_1 es planificable con GC-EDF, pero no así con LC-EDF. Sin embargo, si aplicamos LC-EDF-DS o LC-EDF-GSD conseguimos que el sistema sea planificable. Observamos que con LC-EDF-DS, el más indicado como se ha visto para plazos tan restrictivos, se obtienen unos tiempos de respuesta de peor caso que son globalmente incluso mejores que los obtenidos por GC-EDF.

			Tiempos de respuesta de peor caso			
	T_i	D_i	**GC-EDF**	**LC-EDF**	**LC-EDF DS(k=20)**	**LC-EDF GSD**
Γ_1	100	100	70,3	80	59	74,2
Γ_2	250	200	191,1	230,7	195	191,2
Γ_3	500	450	285	222,4	270	285

Tabla 7-6 Resultados de la aplicación de LC-EDF-DS y LC-EDF-GSD sobre un sistema de control de vuelo.

7.4 Nueva comparativa entre políticas de planificación y conclusiones

Durante el transcurso del presente capítulo se ha demostrado que la idea de que los plazos locales de planificación deban ser conformes al plazo de principio a fin suponía una barrera a la planificación en LC-EDF. Mostramos que rompiendo con esta idea, y asignando plazos locales de planificación cuya suma pueda ser mayor que el plazo de principio a fin, se pueden conseguir mejoras en el rendimiento de LC-EDF. Basándonos en esta nueva noción, propusimos dos nuevas técnicas para asignar plazos locales de planificación que se pueden combinar con las ya existentes.

La primera técnica, llamada LC-EDF-DS (*Deadline Scaling*, escalado de plazos) consistía en multiplicar el plazo de principio a fin por un factor *k*, sólo a efectos de la asignación de los plazos de planificación. Se mostró que aplicar LC-EDF-DS en las técnicas PD y HOSPA da lugar a mejoras en la UMP media de hasta un 7%.

El segundo método de asignación propuesto fue el de la utilización de plazos globales de planificación en LC-EDF, al que llamamos LC-EDF-GSD (*Global Scheduling Deadlines*). Se comprobó que aplicar LC-EDF-GSD con PD o HOSPA produce mejoras de hasta un 36% en la UMP media observada. Utilizar la técnica global EQF de forma directa en LC-EDF produce mejoras adicionales sobre HOSPA con GSD de hasta un 5% en la UMP media.

En el capítulo también se comprobó que combinar LC-EDF-DS y LC-EDF-GSD no produce en general beneficios significativos sobre aplicar sólo LC-EDF-GSD.

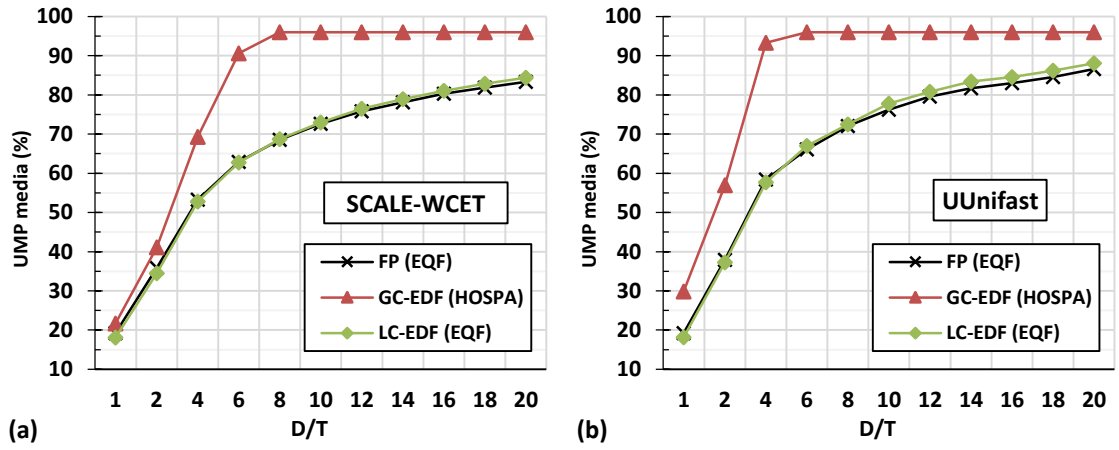


Figura 7-12 UMP media obtenida por las distintas políticas de planificación, aplicando las mejoras en LC-EDF, para distintos plazos de principio a fin y cargas generadas con **(a)** ***SCALE-WCET*** **y (b)** ***UUnifast***

Previo a estas modificaciones, LC-EDF era la política de planificación con menor rendimiento, a gran distancia de la segunda, que era FP. En la Figura 7-12 se muestra la UMP media de las tres políticas de planificación para diferentes plazos de principio a fin. Por simplicidad, para cada política se muestran los resultados de la técnica de asignación con la que en promedio se obtienen los mejores resultados, esto es, HOSPA para GC-EDF, y EQF para FP y LC-EDF. En esta figura se observa que LC-EDF, con los nuevos criterios para la asignación de plazos locales de planificación, alcanza el rendimiento de la política FP, e incluso la supera ligeramente para los plazos de principio a fin mayores. GC-EDF sigue teniendo una marcada ventaja sobre el resto de las técnicas, pero su ventaja sobre LC-EDF se ha reducido a prácticamente la mitad.

Estas conclusiones se mantienen si se estudia la influencia de la longitud de los flujos e2e, como se ilustra en la Figura 7-13. En este caso LC-EDF aumenta su ventaja sobre FP en aquellos sistemas con flujos e2e más cortos.

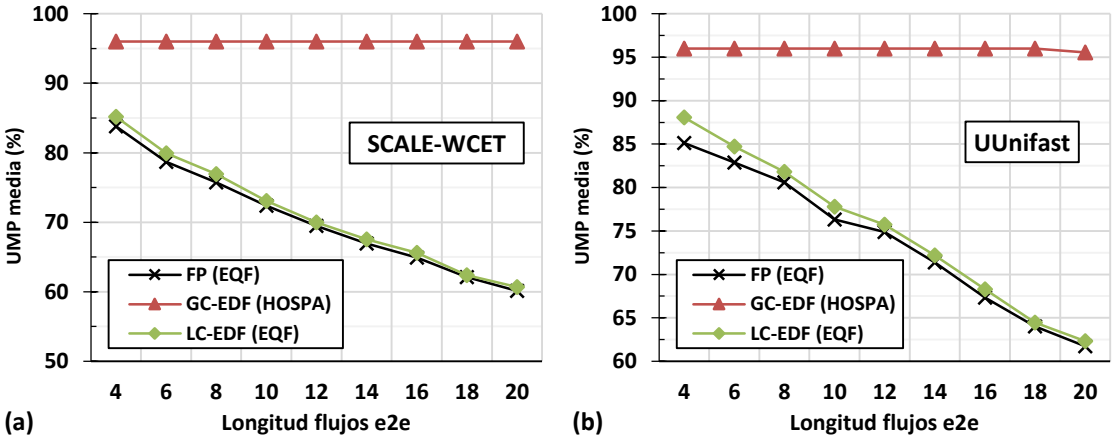


Figura 7-13 UMP media obtenida por las distintas políticas de planificación, aplicando las mejoras en LC-EDF, para flujos con distintas longitudes, y cargas generadas con **(a)** ***SCALE-WCET*** **y (b)** ***UUnifast***

Finalmente, destacar que las mejoras que se han propuesto para la asignación de plazos locales de planificación en LC-EDF son "gratis", es decir, sólo representan la modificación del valor de un parámetro de planificación que se asigna *off-line*, pero no requiere ninguna modificación en el resto del software del sistema.

8 Estudio de la Influencia de la Eliminación del *Jitter*

Como se ha ido observando a lo largo de este trabajo, el *jitter* de activación en las actividades induce un efecto negativo en la planificabilidad de los sistemas distribuidos. En el Capítulo 2 se describió cómo trabajan las técnicas de análisis de planificabilidad con la interdependencia entre el *jitter* y el tiempo de respuesta de peor caso. En los Capítulos 5 y 6 se comprobó mediante dos estudios masivos el efecto negativo que produce el *jitter* en la planificabilidad, observando que para una misma carga computacional total, dividir los flujos e2e en un número mayor de actividades, y por lo tanto aumentar el efecto de la propagación del *jitter*, produce una disminución en la planificabilidad de dichos sistemas. Adicionalmente, en el Capítulo 7 se mostró cómo el *jitter* juega un papel fundamental en los problemas de planificabilidad en LC-EDF. Por todo ello, el estudio del efecto del *jitter* en los sistemas distribuidos puede darnos una visión que nos ayude a evaluar si merece la pena utilizar mecanismos de reducción o eliminación del jitter, que aunque complican la implementación de los sistemas, potencialmente puede tener grandes beneficios en su planificabilidad.

Un método sencillo para reducir el *jitter,* y por lo tanto también los tiempos de respuesta de peor caso, consiste en estimar y utilizar en el análisis de planificabilidad los tiempos de ejecución de mejor caso de las actividades. Puesto que el *jitter* de activación se define como la diferencia entre los tiempos de respuesta de peor caso y mejor caso de la actividad precedente, este método produce una disminución del *jitter*, pero como se pudo comprobar en los Capítulos 5 y 6, este efecto beneficioso es limitado, y afecta especialmente a los flujos con mayor número de actividades.

En [GUT96] se estudia la influencia del *jitter* en sistemas distribuidos planificados por FP, y se propone el uso de servidores esporádicos [SPR89] para su eliminación. En dicho trabajo se observaron incrementos en la utilización máxima planificable de hasta un 50% para sistemas libres de *jitter*. En [SPU96b] se define el servidor esporádico para EDF, que

podría tener el mismo efecto, aunque la influencia del *jitter* en sistemas planificados por EDF no ha sido estudiado.

Otros métodos con los que se puede reducir o incluso eliminar el *jitter* en sistemas FP es mediante la utilización de protocolos especiales de activación de las actividades. Uno de estos protocolos es el *Phase-Modification Protocol* [BET94], que consiste en, para cada flujo e2e Γ_i, activar sus actividades a instantes fijos desde la llegada del evento que activa Γ_i. La utilización de este protocolo requiere sincronización entre los relojes del sistema, además de un conocimiento de los tiempos de respuesta de peor caso, para poder así mantener las relaciones de precedencia en las ejecuciones de las actividades dentro de un flujo e2e.

Una alternativa al *Phase-Modification Protocol* es el *Release Guard Protocol* (RGP) [SUN96]. Con RGP las activaciones se establecen de forma que el tiempo transcurrido entre dos activaciones de una actividad τ_{ij} nunca sea menor que el periodo del flujo e2e en el que reside (T_i). La extensión de este protocolo para sistemas EDF se conoce como *Distributed Deadline Synchronization Protocol* (DDSP) [SER10].

En esta sección nos planteamos como objetivo cuantificar el efecto de eliminar el *jitter* de activación en sistemas distribuidos planificados por FP, GC-EDF y LC-EDF. El estudio se va a llevar a cabo a dos niveles distintos, siguiendo la línea del trabajo realizado en los Capítulos 5 y 6.

En primer lugar se estudiará el rendimiento de las técnicas de asignación de parámetros de planificación en sistemas distribuidos libres de *jitter*. Se comprobará además si las propuestas para LC-EDF presentadas en el Capítulo 7 (LC-EDF-DS y LC-EDF-GSD) siguen siendo aplicables a sistemas en los que el *jitter* se ha eliminado.

En segundo lugar se examinará cómo afecta la eliminación del *jitter* a los tiempos de respuesta de peor caso obtenidos por las diferentes técnicas de análisis de planificabilidad. Aunque se espera una reducción generalizada de los tiempos de respuesta, en este estudio se intentará observar en qué situaciones esta reducción es mayor, y cómo afecta el *jitter* a las distintas técnicas de análisis.

A lo largo de este estudio nos centraremos únicamente en observar las consecuencias de poseer un sistema libre de *jitter*, con independencia del procedimiento utilizado para conseguirlo. Para ello, no es nuestro objetivo comparar los diferentes métodos de eliminación del *jitter* existentes, o proponer otros nuevos, sino que modificaremos las técnicas de análisis de planificabilidad incluidas en MAST para que emulen la aplicación de cualquiera de estos métodos (servidores esporádicos, RGP, DDSP, etc.). Como paso previo al estudio que hemos propuesto, en la siguiente sección exponemos las modificaciones realizadas en las técnicas de análisis para emular sistemas libres de *jitter*.

8.1 Emulación de la eliminación del *jitter* en MAST

En MAST, un método sencillo que puede utilizar el usuario para eliminar el *jitter* en alguna o todas las actividades de un sistema es el establecimiento de *offsets* en la descripción del sistema. El *offset* es un elemento del modelo MAST que puede poseer una actividad (ver Sección 1.4), y que establece la cantidad de tiempo mínimo que debe transcurrir desde la activación del evento externo que activa el flujo e2e hasta que se activa la actividad. Para que un *offset* elimine el *jitter* de una actividad, éste debe tener un

valor suficientemente alto para que la variabilidad en la ejecución de las actividades que le preceden en el flujo e2e no le afecte, es decir, que el *offset* debe ser al menos igual al tiempo de respuesta de peor caso de la actividad precedente en el flujo e2e. Este método de activación de las actividades es equivalente al *Phase-Modification Protocol*.

Salvo en sistemas triviales, los tiempos de respuesta de peor caso son valores que se desconocen a la hora de describir el sistema, por lo que típicamente se sigue un proceso de dos pasos: en primer lugar se calculan los tiempos de respuesta de peor caso sin haber especificado *offsets*, y a continuación se utilizan esos tiempos de respuesta de peor caso para calcular los *offsets*. Además de estar obligado a analizar el sistema en dos ocasiones, el problema de seguir éste método es que los *offsets* resultantes son innecesariamente grandes, ya que se calculan a partir de unos tiempos de respuesta obtenidos para una situación con *jitter*.

El método que proponemos para emular la eliminación del *jitter* se basa en el cálculo iterativo de los *offsets* de las actividades, haciendo uso de los conceptos utilizados en el análisis para sistemas heterogéneos (ver Sección 3.1). La modificación consiste en, tras el análisis de cada actividad, actualizar el *offset* de la siguiente actividad del flujo e2e con los valores del tiempo de respuesta de peor caso calculado. Éste método posee estas características fundamentales:

1. Las técnicas de análisis ya están adaptadas para interpretar los *offsets*, por lo que no requieren modificaciones.
2. Las modificaciones se añaden únicamente en la inicialización y actualización de los *jitters* y *offsets* heredados del análisis heterogéneo (J'_{ij} y ϕ'_{ij} respectivamente), siendo éste un elemento común a todas las técnicas de análisis. Un único cambio en estos elementos afecta a todas las técnicas de análisis.

A modo de recordatorio, en la siguiente figura se resume en forma de pseudocódigo el funcionamiento del análisis para sistemas heterogéneos presentado en el Capítulo 3, en el que se destacan en rojo los puntos en los que se realizan los cambios:

```
function analisis_heterogéneo
   inicialización de jitters y offsets heredados
   loop
      for each flujo e2e
      loop
         for each actividad en el flujo
         loop
            Análisis de actividad (nuevos tiempos de respuesta)
            actualización de jitters y offsets heredados
         end loop
      end loop
      exit when no hay cambios
   end loop
end analysis
```

Figura 8-1 Pseudocódigo análisis heterogéneo.

Para emular la eliminación del *jitter*, la inicialización de los *jitters* y *offsets* se realiza según las siguientes ecuaciones para una actividad τ_{ij}:

$$R_{ij} = \sum_{j=1}^{i} C_{ij} \qquad (8\text{-}1)$$

$$J'_{ij} = 0 \tag{8-2}$$

$$\begin{aligned} \Phi'_{ij} &= R_{ij-1} \qquad j > 1 \\ \Phi'_{ij} &= 0 \qquad\quad j = 1 \end{aligned} \tag{8-3}$$

Donde, para una actividad τ_{ij}, R_{ij} es su tiempo de respuesta de peor caso, J'_{ij} su *jtter* heredado, ϕ'_{ij} su *offset* heredado, y C_{ij} su tiempo de ejecución de peor caso.

Una vez llevado a cabo el análisis de cada actividad τ_{ij}, actualizamos el *offset* de la actividad posterior en el flujo (si la hubiera) según la siguiente ecuación.

$$\Phi'_{ij+1} = R_{ij} \qquad j < N_i \tag{8-4}$$

donde N_i es el número de actividades del flujo en el que reside τ_{ij}.

La interdependencia entre el *offset* y el tiempo de respuesta de peor caso es similar a la que ocurre entre el *jitter* y el tiempo de respuesta de peor caso en el análisis holístico estándar, y se resuelve de igual manera, de forma iterativa. La monotonicidad de esta interdependencia asegura la convergencia a unos valores de tiempos de respuesta de peor caso y *offsets* que representan una situación libre de *jitter*.

Aunque planteamos este método para obtener los tiempos de respuesta que se obtendrían si utilizáramos algún método de eliminación del *jitter*, los *offsets* a los que converge el análisis también resultan interesantes, ya que representan los *offsets* más pequeños que pueden ser asignados a las actividades de un sistema dado para que se elimine el *jitter* por completo. La implementación de dicho sistema con *offsets* requiere que los relojes de los recursos procesadores estén sincronizados.

Para poder hacer uso del método de emulación que planteamos, lo integramos dentro del módulo de análisis para sistemas heterogéneos de MAST. Para activar el modo de emulación de eliminación del jitter añadimos un nuevo parámetro de ejecución en línea de comandos de MAST, que denotamos con "*-jitter_avoidance*". Si este parámetro es suministrado en la ejecución de MAST, durante el análisis se aplica la inicialización y actualización de términos especificadas previamente con las ecuaciones (8-1) a (8-4).

En conjunción con la implementación en MAST, incorporamos a GEN4MAST la opción de poder analizar sistemas con emulación de la eliminación del *jitter*. Para ello, añadimos el parámetro booleano "JITTER_AVOIDANCE" a la sección "EXECUTION" del fichero de configuración de GEN4MAST. Si dicho parámetro tiene un valor *True*, se especifica una ejecución de MAST con el parámetro "*-jitter_avoidance*". En la Figura 8-2 se muestra la sintaxis para este parámetro.

```
[EXECUTION]
JITTER_AVOIDANCE True/False …
[/EXECUTION]
```

Figura 8-2 Sintaxis parámetro JITTER_AVOIDANCE en GEN4MAST.

Una vez definido el método para emular la aplicación de técnicas de eliminación del *jitter*, y que éste ha sido integrado en MAST y GEN4MAST, ya estamos en disposición de llevar a cabo los estudios que nos marcamos como objetivos en esta sección.

8.2 Asignación de parámetros de planificación en ausencia de *jitter*

En el Capítulo 6 se estudió el rendimiento de las diferentes técnicas de asignación de parámetros de planificación disponibles en MAST. Aquí planteamos una comparativa similar en la que estudiaremos el efecto de eliminar el *jitter* con dos objetivos: cuantificar la mejora que se obtiene en la asignación de parámetros de planificación al eliminar el *jitter*, y comparar el rendimiento de las distintas técnicas de asignación con sistemas libres de *jitter*. En primer lugar definimos el conjunto de sistemas que formarán parte del estudio.

8.2.1 Dominio y ejecución del estudio

Los resultados del estudio presentado en el Capítulo 6 para sistemas con *jitter* nos aportaron una idea general sobre qué parámetros de los sistemas son los más influyentes en los resultados. Para simplificar la ejecución y presentación de los resultados de la presente comparativa, definimos un nuevo dominio en el que nos centraremos en el estudio de las características que se observaron como más relevantes en los estudios previos:

- Plazos de principio a fin: T, T1, T2 y NT en la nomenclatura de GEN4MAST.
- Longitud de los flujos e2e: 4, 10 y 16 actividades por flujo e2e.
- Número de procesadores: 5 y 10.
- Método de generación de la carga: Cargas generadas con *SCALE-WCET* y *UUnifast*.

El resto de características se mantienen como en el sistema base utilizado en el Capítulo 6, esto es:

- Número de flujos e2e: 10.
- Número de actividades para todos los flujos e2e (longitud): 10
- Localización de las actividades: Si es posible, no se repite un recurso procesador en el flujo (pseudo-aleatoria).
- Periodos: Se seleccionan aleatoriamente en el rango [100, 1000] (Ratio=10), mediante una distribución de probabilidad logarítmico-uniforme (*Log-Uniform*).
- Tiempos de ejecución de mejor caso de las actividades igual a 0.
- Todos los procesadores poseen la misma utilización en todo momento.
- Se generan series de utilizaciones en el rango [10, 96] (en %).
- Número de recursos procesadores: 5
- Plazos de principio a fin: $D_i=N_i*T_i$
- Utilización de las actividades calculadas mediante el algoritmo *UUnifast*.

Para obtener resultados con relevancia estadística, generamos 30 sistemas con cada combinación de características (parámetro “POPULATION 30” en GEN4MAST).

Como técnica de análisis de planificación utilizamos en todos los casos el análisis holístico, por ser el único disponible para los tres tipos de planificación. En todos los

casos el análisis lo realizamos por duplicado: análisis normal con *jitter*, y el análisis sin *jitter* estableciendo el parámetro JITTER_AVOIDANCE de GEN4MAST a valor True.

Los resultados de la comparativa se presentan de manera separada para cada política de planificación.

8.2.2 Planificación FP

Aplicamos las técnicas de asignación de prioridades fijas (UD, ED, PD, EQS, EQF y HOSPA) sobre todo el dominio de sistemas generado. No incluimos la técnica NPD ya que no vamos a estudiar sistemas con recursos procesadores con distintas utilizaciones. Como criterio de comparación nos centraremos en observar la evolución de la utilización máxima planificable (UMP) media obtenida.

En primer lugar, en la Figura 8-3 observamos la evolución de la UMP media para los 4 tipos de plazos de principio a fin generados, para análisis con *jitter* y sin *jitter*. Utilizamos una gráfica de barras en la que cada barra representa la UMP media obtenida por una técnica de asignación dada con un tipo de plazo de principio a fin. La sección de color oscuro de la barra representa la UMP media obtenida para el análisis con *jitter*, y la parte más clara representa la UMP media obtenida para el análisis eliminando el *jitter*. Este tipo de representación de los resultados se puede llevar a cabo debido a que las UMP medias obtenidas con cancelación del *jitter* siempre son mayores o iguales que las

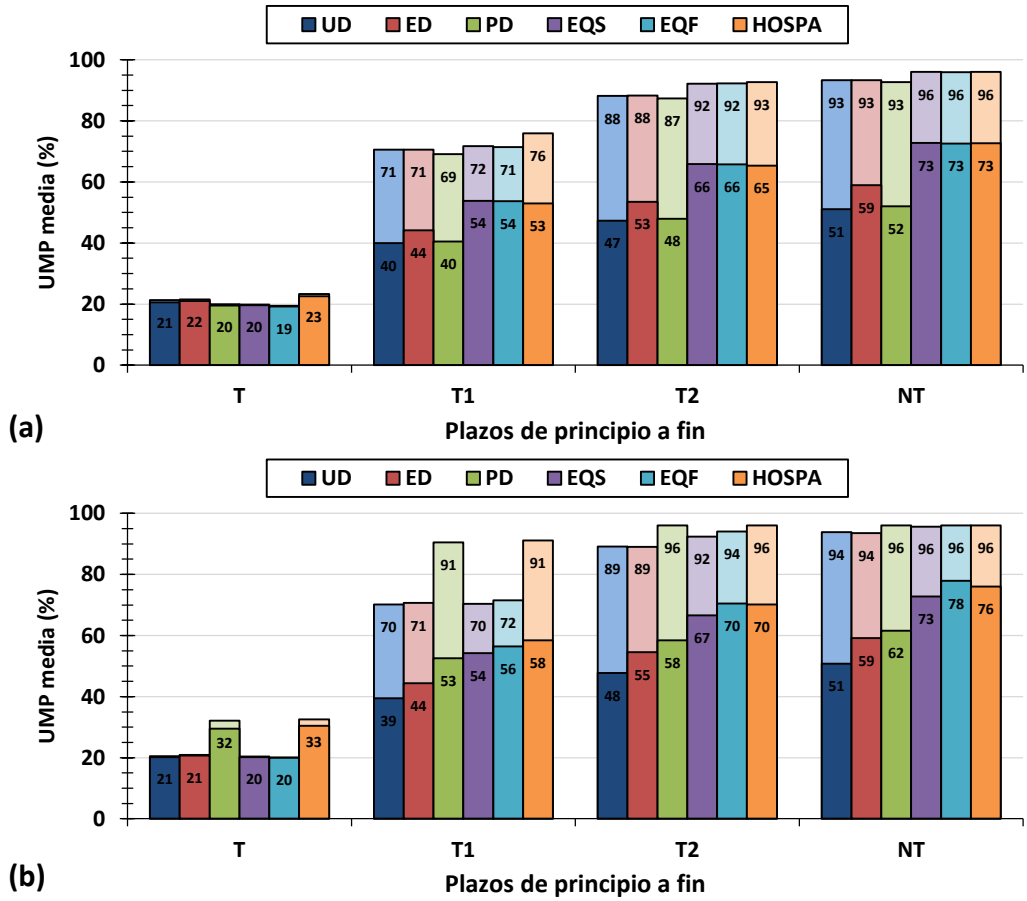


Figura 8-3 UMP media obtenida por los algoritmos de asignación en FP, para distintos plazos de principio a fin, con *jitter* (color oscuro) y sin *jitter* (color claro), y cargas generadas con **(a) *SCALE-WCET* y (b) *UUnifast***

obtenidas sin cancelarlo. También se muestran los valores de las UMP medias redondeados al entero más próximo.

Lo primero que se puede observar en dicha figura es que para los plazos de principio a fin más restrictivos, esto es $D_i=T_i$, los beneficios obtenidos al eliminar el *jitter* son muy limitados. Para $D_i>T_i$, las ventajas de eliminar el *jitter* resultan más evidentes, con mejoras que se mueven entre el 20% y el 40% en la UMP media aproximadamente. En el Capítulo 6 concluimos que para sistemas FP afectados por *jitter*, la técnica de asignación de prioridades fijas que mejores resultados obtenía en términos generales era EQF. Esta conclusión no se mantiene si utilizamos algún método de eliminación del *jitter*, siendo HOSPA la técnica de asignación de prioridades fijas que mejores resultados obtiene en este caso. El rendimiento de HOSPA, y en especial de su asignación inicial PD, queda especialmente retratado en sistemas con cargas generadas por *UUnifast* (Figura 8-3(b)), para los que estos algoritmos poseen una clara ventaja sobre EQF en $D_i=T1$ y en menor medida para plazos de principio a fin mayores. La disminución de las diferencias para los plazos de principio a fin mayores es debida a que éstas son situaciones en las que la UMP media llega a alcanzar el valor de la utilización máxima generada del 96%.

A continuación estudiaremos el efecto de la eliminación del *jitter* para sistemas con diferentes longitudes en sus flujos e2e. Para no obtener utilizaciones máximas planificables saturadas en valores próximos al 96%, y poder así observar con mayor claridad las diferencias entre las diferentes técnicas, mostraremos los resultados obtenidos para sistemas con plazos de principio a fin $D_i=T1$. Mostramos estos resultados en la

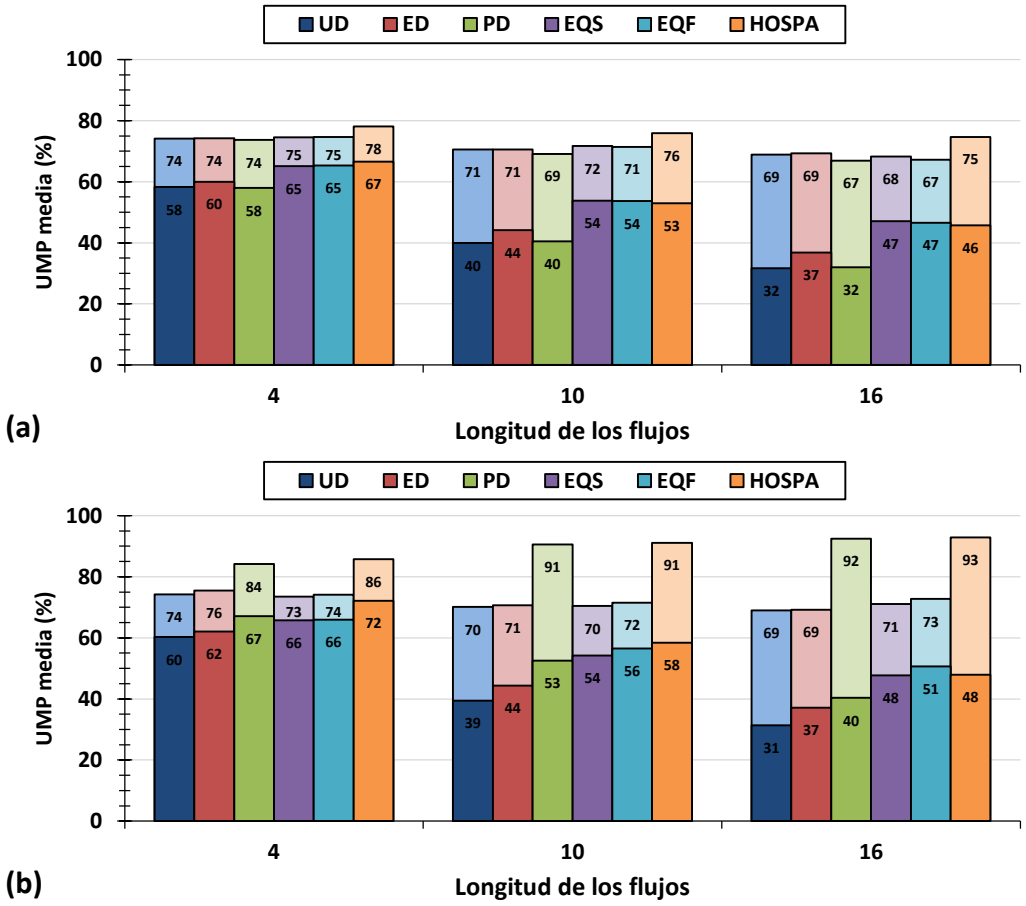


Figura 8-4 UMP media obtenida por los algoritmos de asignación en FP, para distintas longitudes de los flujos e2e, con *jitter* (color oscuro) y sin *jitter* (color claro), y cargas generadas con **(a) *SCALE-WCET* y (b) *UUnifast***

Figura 8-4, donde se aprecia que a medida que los flujos e2e poseen más actividades, aumenta también la propagación del *jitter* a lo largo del flujo, y por lo tanto también su efecto negativo, tal y como ya se observó en el Capítulo 6. Si eliminamos el *jitter* observamos que esta tendencia se amortigua enormemente, e incluso observamos que para cargas generadas con *UUnifast*, los algoritmos PD (y por extensión HOSPA) aumentan su rendimiento a medida que los flujos son más largos. Como era de esperar, los mayores beneficios al eliminar el *jitter* se obtienen cuando los flujos son más largos.

Por último estudiamos el efecto de la variación del número de recursos procesadores en el sistema, igualmente para plazos D_i=T1. Los resultados se resumen en la Figura 8-5. Para cargas generadas con *SCALE-WCET* (Figura 8-5(a)) se observa que en situaciones libres de *jitter,* todos los algoritmos de asignación obtienen menores UMP medias cuando se pasa de sistemas con 5 a 10 recursos procesadores. Este comportamiento se observa igualmente para sistemas con cargas de tipo *UUnifast* (Figura 8-5(b)). Independientemente del tipo de carga y del número de recursos procesadores estudiado, en las situaciones libres de *jitter* el algoritmo HOSPA es el que mayores UMP medias registra. Adicionalmente, se observa que para cargas *UUnifast*, al algoritmo HOSPA apenas mejora la UMP media obtenida por su asignación inicial PD, con independencia del número de recursos procesadores.

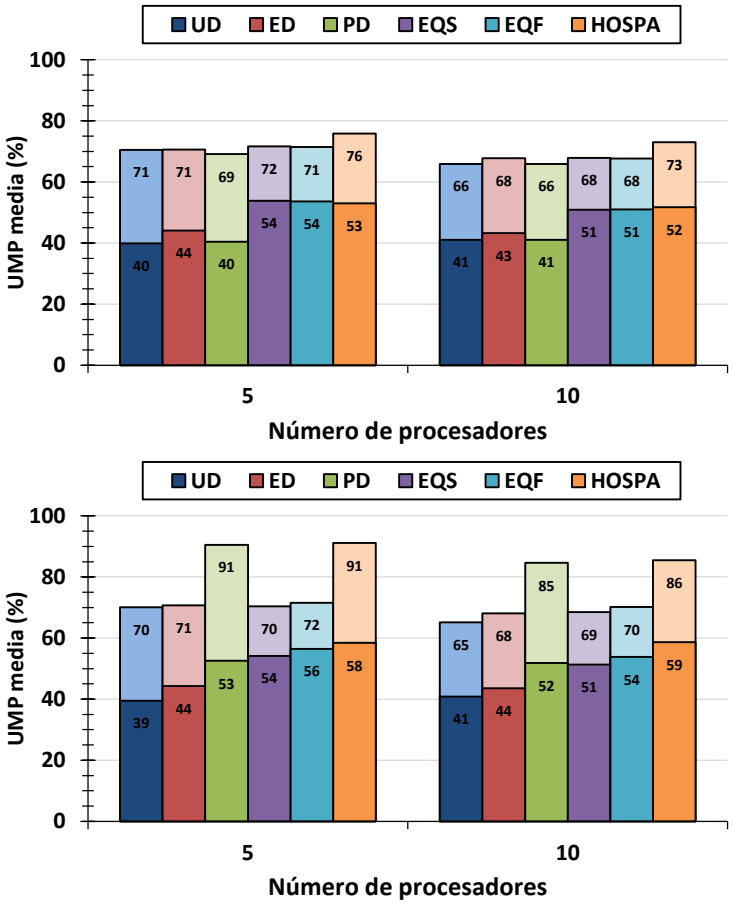


Figura 8-5 UMP media obtenida por los algoritmos de asignación en FP, para distintas números de recursos procesadores, con *jitter* (color oscuro) y sin *jitter* (color claro), y cargas generadas con **(a) *SCALE-WCET* y (b) *UUnifast***

8.2.3 Planificación GC-EDF

Repetimos el mismo proceso comparativo para los sistemas planificados por GC-EDF. Estudiando los resultados para distintos plazos de principio a fin (Figura 8-6) observamos que para D_i>T1 y el *jitter* eliminado, las técnicas con mejor rendimiento (PD y HOSPA) obtienen una UMP media cercana al valor límite del 96%. Es llamativo el caso con plazos D_i=T1 y cargas *SCALE-WCET* (Figura 8-6(a)), en el que se observa que eliminar el *jitter* apenas produce mejoras en las asignaciones PD y HOSPA. Por el contrario, las asignaciones UD y ED sí encuentran mejoras, adelantando a PD y HOSPA para este tipo de plazos de principio a fin. EQS y EQF presentan un rendimiento peor que los demás algoritmos tanto con *jitter* como sin él. Al igual que lo observado en FP, con planificación GC-EDF eliminar el *jitter* no produce un beneficio sustancial para sistemas con los plazos de principio a fin más restrictivos (D_i=T). Para cargas de tipo *UUnifast* (Figura 8-6(b)) se observa una tendencia similar, con la salvedad de que para plazos D_i=T1 y situaciones libres de *jitter*, los algoritmos UD y ED no mejoran los resultados obtenidos por PD y HOSPA.

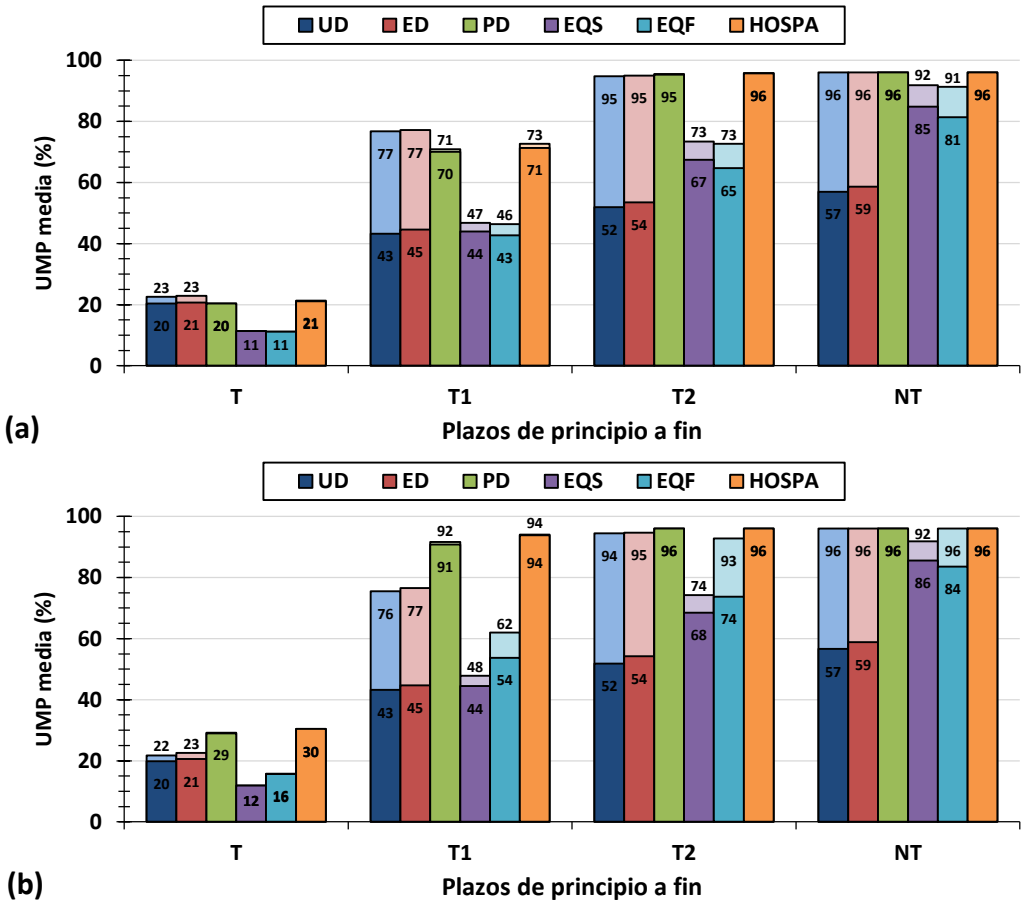


Figura 8-6 UMP media obtenida por los algoritmos de asignación en GC-EDF, para distintos plazos de principio a fin, con *jitter* (color oscuro) y sin *jitter* (color claro), y cargas generadas con **(a) *SCALE-WCET* y (b) *UUnifast***

Para observar la influencia de la longitud de los flujos e2e (Figura 8-7) nos centramos en los sistemas con plazos de principio a fin D_i=T1, puesto que con estos plazos las UMP medias no llegan al límite estudiado. En la figura se vuelve a comprobar que en sistemas sin *jitter* y cargas *UUnifast*, las asignaciones PD y HOSPA son las que mejores soluciones aportan, independientemente de la longitud de los flujos e2e. Incluso se aprecia un aumento de la planificabilidad cuando los flujos e2e son más largos. Para

cargas *SCALE-WCET* en cambio, las asignaciones UD y ED se mantienen como las mejores en todo el rango de longitudes en ausencia de *jitter*. A destacar que la mejora que se produce en estos algoritmos aumenta con la longitud del flujo e2e, aunque también se advierte que con este tipo de cargas, aumentar la longitud de los flujos e2e hace que los límites de planificabilidad de todos los algoritmos disminuyan para sistemas libres de *jitter*.

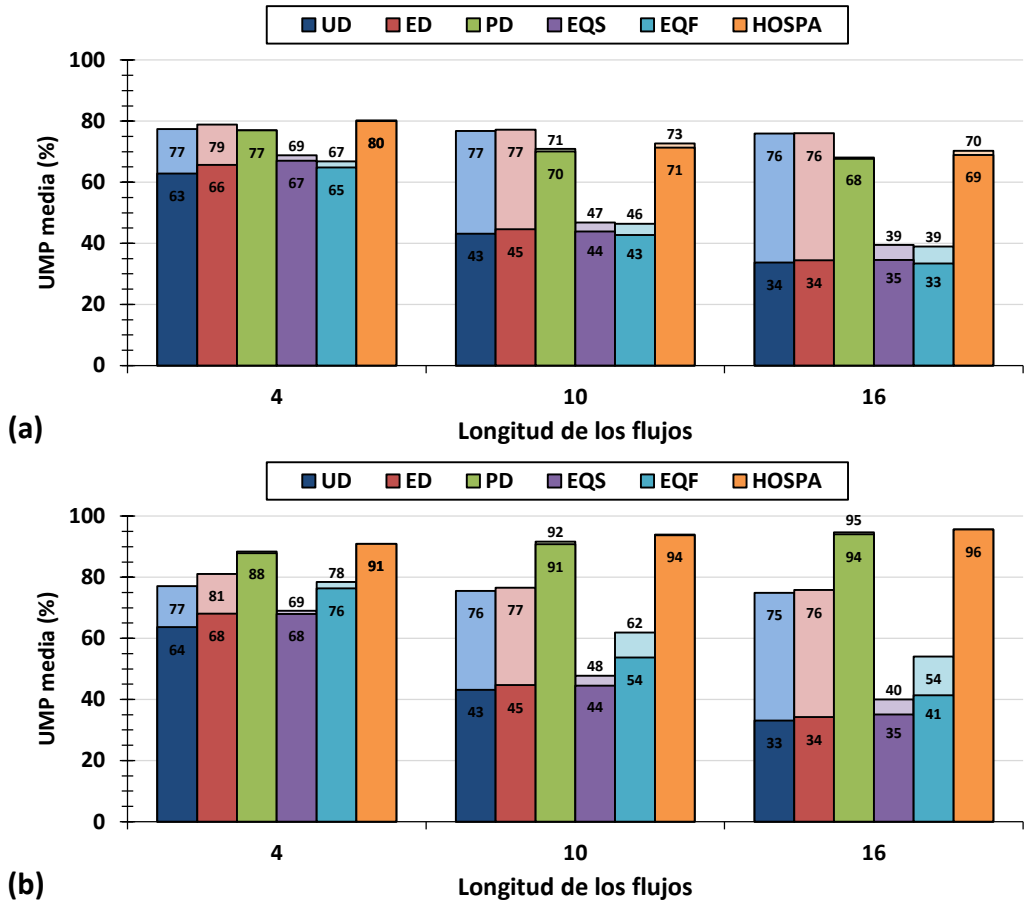


Figura 8-7 UMP media obtenida por los algoritmos de asignación en GC-EDF, para distintas longitudes de los flujos e2e, con *jitter* (color oscuro) y sin *jitter* (color claro), y cargas generadas con **(a) *SCALE-WCET* y (b) *UUnifast***

Finalmente, variar el número de procesadores (Figura 8-8) produce un ligero descenso en la planificabilidad de los sistemas, tanto con *jitter* como sin él. El resto de conclusiones se mantienen, así para cargas *SCALE-WCET* las asignaciones UD y ED tienen una ligera ventaja, mientras que con cargas *UUnifast* los algoritmos PD y HOSPA son los que mejores resultados obtienen.

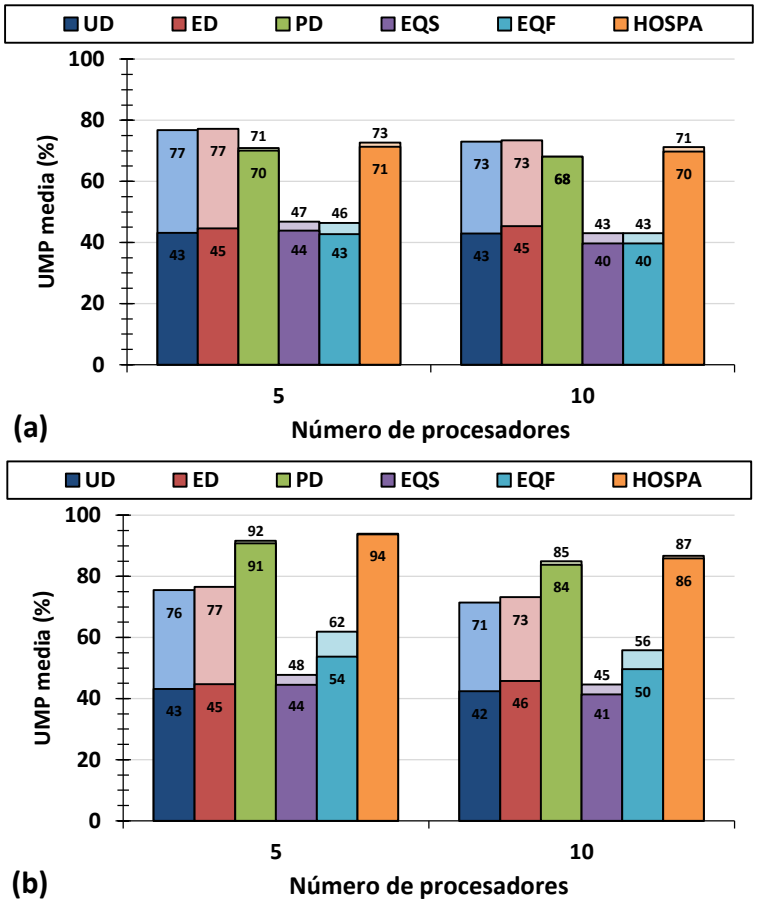


Figura 8-8 UMP media obtenida por los algoritmos de asignación en GC-EDF, para distintas números de procesadores, con *jitter* (color oscuro) y sin *jitter* (color claro), y cargas generadas con **(a) *SCALE-WCET* y (b) *UUnifast***

8.2.4 Planificación LC-EDF

Para los sistemas LC-EDF repetimos el mismo proceso de estudio seguido anteriormente en FP y GC-EDF. En primer lugar aplicamos las técnicas de asignación de parámetros de planificación disponibles para LC-EDF sin las modificaciones propuestas en el Capítulo 7 (LC-EDF-DS y LC-EDF-GSD).

Probando distintos tipos de plazos de principio a fin (Figura 8-9) observamos una situación que se asemeja a lo observado en los sistemas FP. Para los plazos de principio a fin más restrictivos (D_i=T), los beneficios de eliminar el *jitter* son prácticamente inexistentes. Los beneficios sólo se manifiestan para plazos de principio a fin mayores. Para cargas generadas con *SCALE-WCET* las diferentes técnicas obtienen resultados similares una vez que se elimina el *jitter*. Sin embargo, para cargas *UUnifast* se observa que, en ausencia de *jitter*, las asignaciones PD y HOSPA se distancian del resto.

Otro aspecto importante a destacar de estos resultados es que el mayor rendimiento que obtienen EQS y EQF para sistemas con *jitter* se pierde si se elimina el *jitter*. Esto es un indicativo de que la principal ventaja de utilizar plazos globales de planificación en LC-EDF es que aportan soluciones que atenúan el efecto del *jitter*. Esta conclusión se refuerza cuando observamos el drástico incremento en el rendimiento de PD y HOSPA

cuando el *jitter* se elimina. El aumento registrado en la UMP media es de hasta un 58% cuando las cargas son generadas por *SCALE-WCET*.

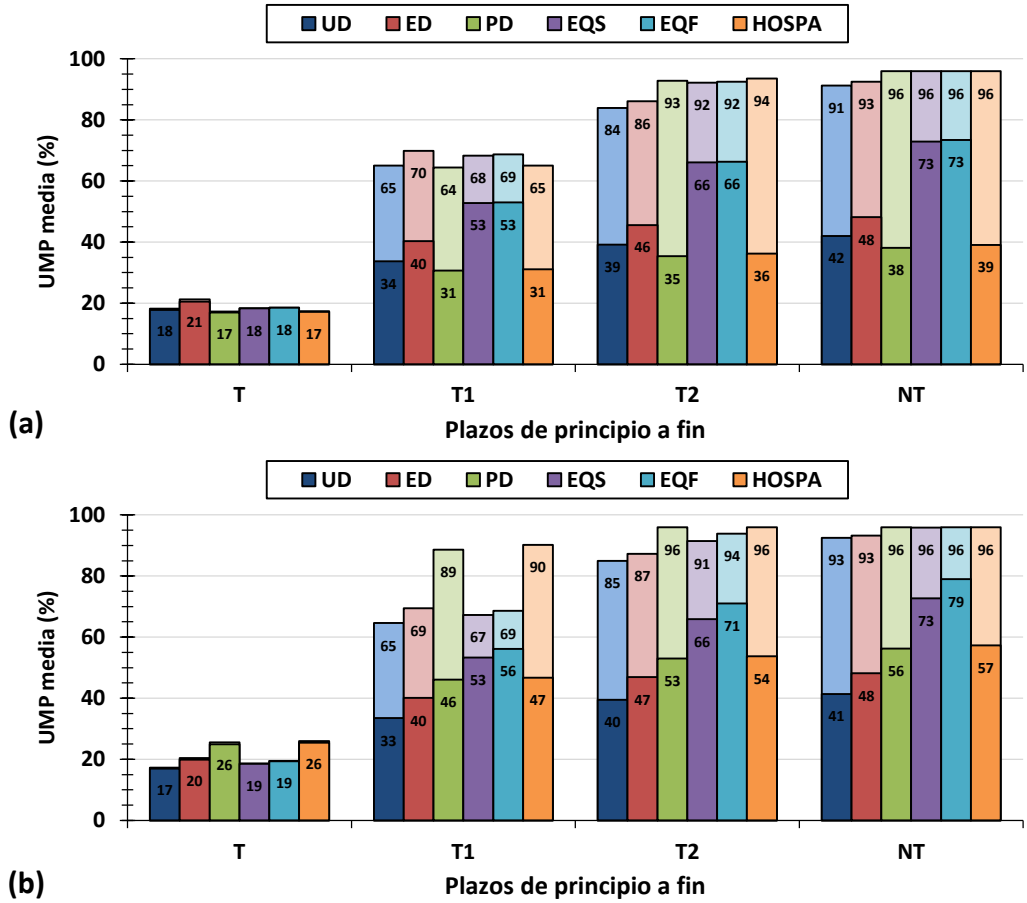


Figura 8-9 UMP media obtenida por los algoritmos de asignación en LC-EDF, para distintos plazos de principio a fin, con *jitter* (color oscuro) y sin *jitter* (color claro), y cargas generadas con **(a) *SCALE-WCET* y (b) *UUnifast***

Para estudiar el efecto de modificar la longitud de los flujos e2e nos centraremos en los sistemas con plazos de principio a fin D_i=T1, puesto que con plazos de principio a fin mayores las UMP obtenidas empiezan a alcanzar el valor máximo del 96%, dificultando así apreciar las diferencias entre las distintas técnicas. Como era de esperar, en los resultados, que se presentan en la Figura 8-10, se observa que los beneficios de eliminar el *jitter* aumentan a medida que los flujos e2e poseen más actividades. Cuando la carga se genera con *SCALE-WCET*, aumentar la longitud en los sistemas sin *jitter* produce un deterioro general del rendimiento en todos los algoritmos de asignación. Sin embargo, para las cargas *UUnifast* se observa que PD y HOSPA aumentan su UMP media a medida que los flujos e2e poseen mayores longitudes, con incrementos de hasta un 58% con flujos e2e de longitud 16.

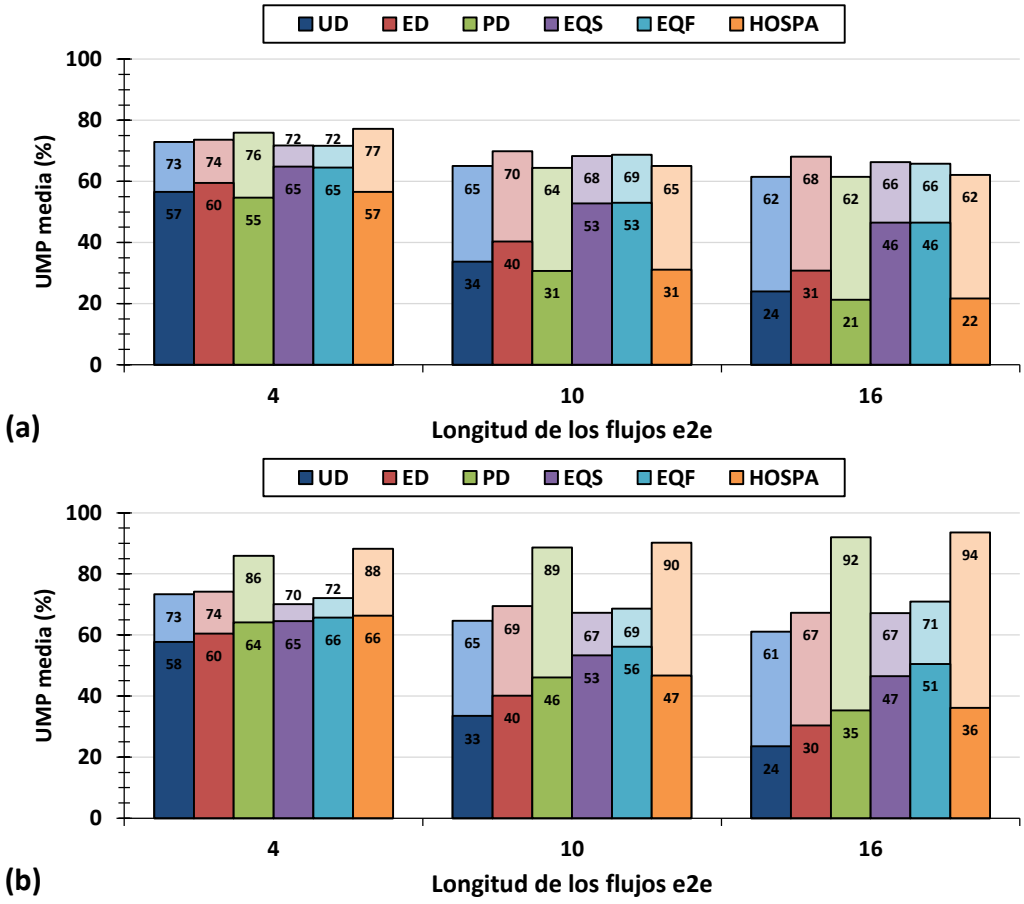


Figura 8-10 UMP media obtenida por los algoritmos de asignación en LC-EDF, para distintas longitudes de los flujos e2e, con *jitter* (color oscuro) y sin *jitter* (color claro), y cargas generadas con **(a) *SCALE-WCET* y (b) *UUnifast***

Variando el número de procesadores (Figura 8-11) se observan que para cargas *SCALE-WCET*, aumentar el número de procesadores en situaciones libres de *jitter* hace que los algoritmos ED, EQS y EQF ven disminuido su rendimiento, mientras que en UD, PD y HOSPA se aprecia un aumento de la UMP media, convirtiéndose ésta último en la técnica que mejores resultados obtiene. Para cargas *UUnifast*, aumentar el número de procesadores en sistemas libres de *jitter* produce un empeoramiento de la UMP media en todos los algoritmos de asignación estudiados, aunque HOSPA y su asignación inicial PD siguen siendo los algoritmos que mejores resultados obtienen.

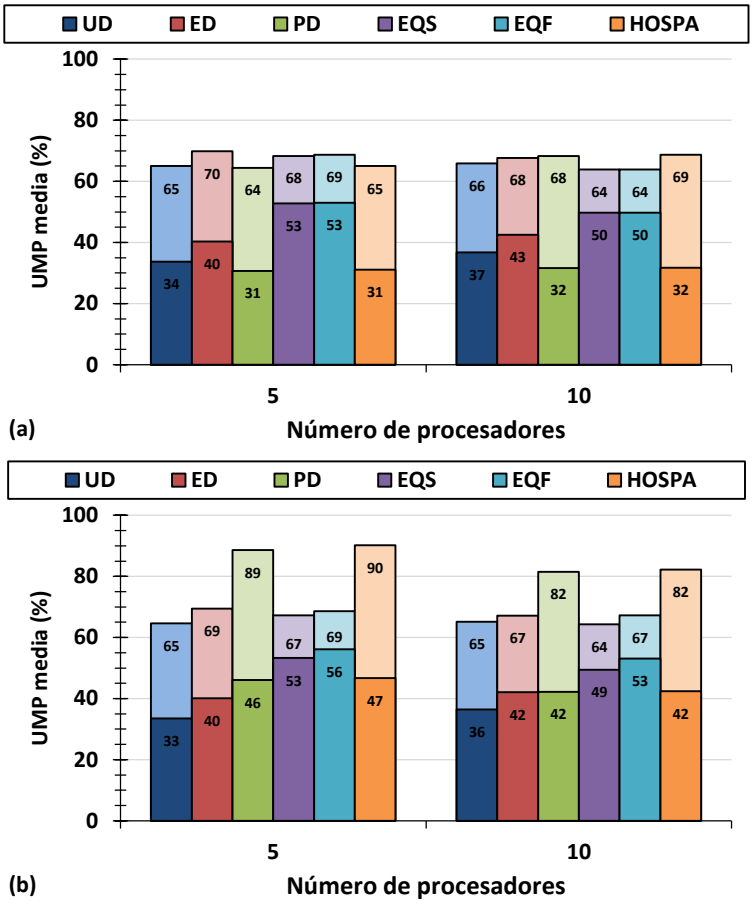


Figura 8-11 UMP media obtenida por los algoritmos de asignación en LC-EDF, para distintas números de procesadores, con *jitter* (color oscuro) y sin *jitter* (color claro), y cargas generadas con **(a) *SCALE-WCET* y (b) *UUnifast***

Con objeto de completar el estudio, vamos a comprobar si las modificaciones propuestas en el Capítulo 7 para mejorar el rendimiento de LC-EDF (LC-EDF-DS y LC-EDF-GSD) también producen beneficios en sistemas en los que el *jitter* ha sido eliminado. Para ello volvemos a ejecutar PD y HOSPA sobre el mismo dominio de estudio, pero esta vez aplicando LC-EDF-DS con k=20, LC-EDF-GSD, y la combinación de ambas. Elegimos k=20 ya que es el valor estudiado que mayores beneficios producía.

Los resultados para cargas *SCALE-WCET* y *UUnifast* se muestran en la Figura 8-12. Nos centramos en las variaciones de los plazos de principio a fin, puesto que en ellas se pueden observar las principales diferencias entre los algoritmos. En la figura observamos que los métodos de carga tienen una gran influencia en el rendimiento de LC-EDF-DS y LC-EDF-GSD en sistemas sin *jitter*. Para cargas *SCALE-WCET* (Figura 8-12(a)) y plazos D_i=T1 se comprueba que las modificaciones propuestas producen un ligero aumento en el rendimiento, siendo éste inferior al registrado para sistemas con *jitter*. Sin embargo, con plazos D_i=T2, ambas modificaciones hacen que los algoritmos funcionen peor. Si observamos los resultados para cargas *UUnifast* (Figura 8-12(b)), el efecto de aplicar LC-EDF-DS o LC-EDF-GSD en sistemas sin *jitter* es claramente negativo.

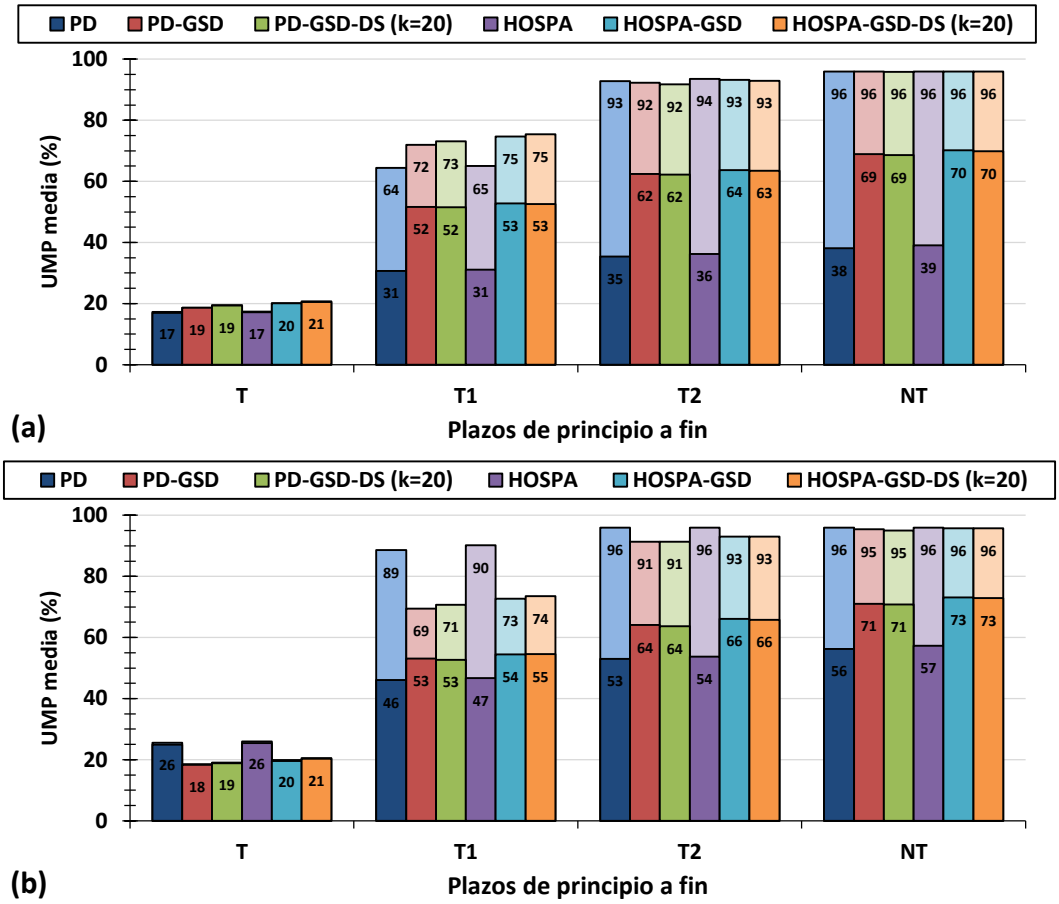


Figura 8-12 UMP media obtenida por los algoritmos PD y HOSPA para LC-EDF, con las modificaciones DS y GSD, para distintos plazos de principio a fin, con *jitter* (color oscuro) y sin *jitter* (color claro), y cargas generadas con **(a)** ***SCALE-WCET*** **y (b)** ***UUnifast***

8.2.5 Comparativa entre políticas de planificación en sistemas sin *jitter*

Hemos estudiado el rendimiento de las diferentes técnicas de asignación de parámetros de planificación en sistemas en los que el *jitter* ha sido eliminado. Se han determinado cuáles son las técnicas con mejor rendimiento en estas situaciones: HOSPA en FP, LC-EDF y GC-EDF (con cargas *UUnifast*); y ED en GC-EDF con cargas *SCALE-WCET*. En la Figura 8-13 recopilamos estos resultados, comparando el rendimiento de las diferentes políticas de planificación, en la que además se incluyen como referencia las UMP medias obtenidas para los sistemas con *jitter*. Observamos que las grandes diferencias entre las políticas que aparecían en los sistemas con *jitter* se ven drásticamente reducidas al eliminarlo. GC-EDF sigue siendo la política que en general posee la mayor capacidad para planificar, pero FP se sitúa cerca, e incluso le supera con plazos D_i=T y carga *UUnifast*. LC-EDF se manifiesta como la tercera política en términos de rendimiento, pero la diferencia con respecto GC-EDF la sitúa a menos de un 10% en la UMP media alcanzada.

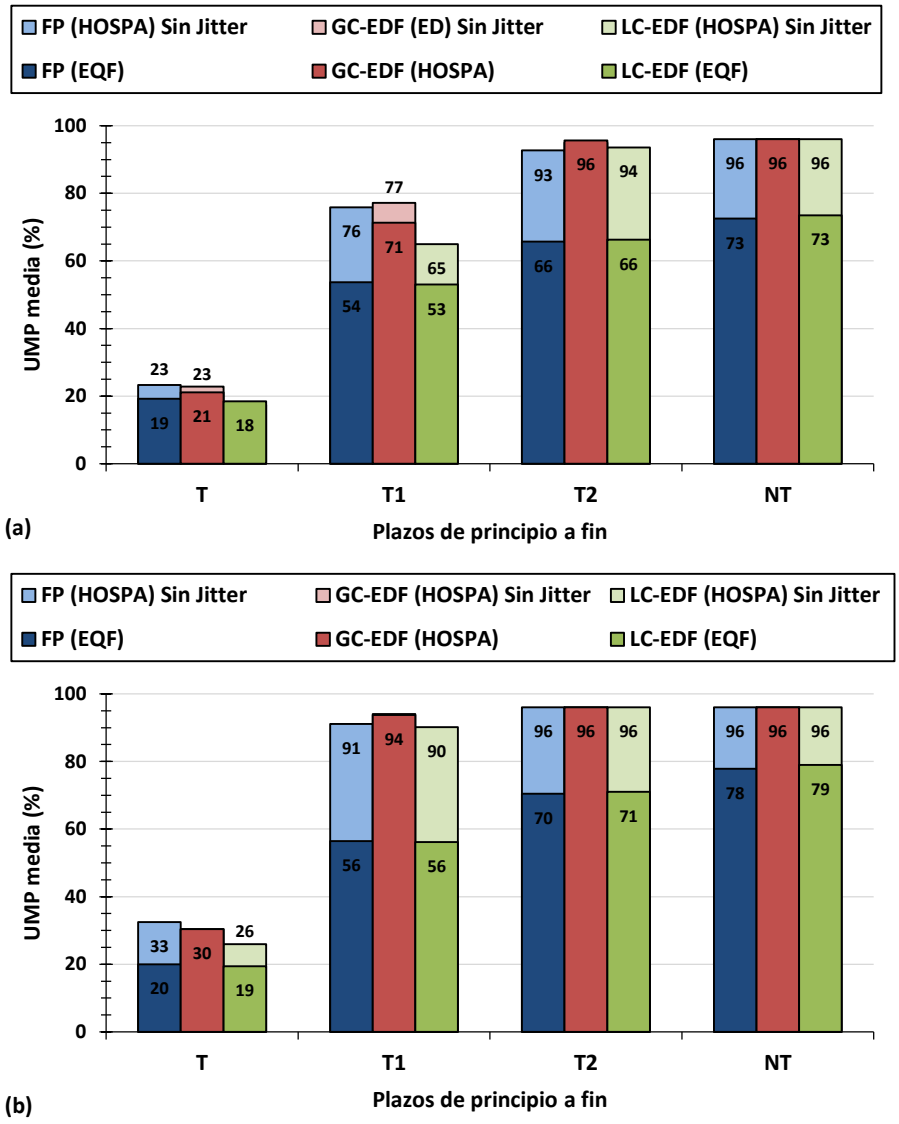


Figura 8-13 UMP media obtenida por las tres políticas de planificación (FP, GC-EDF y LC-EDF) en sistemas con *jitter* (color oscuro), y sin *jitter* (color claro), utilizando los algoritmos de asignación que en promedio obtienen los mejores resultados, y cargas generadas con **(a) *SCALE-WCET* y (b) *UUnifast***

8.3 Comparativa de técnicas de análisis de planificabilidad en ausencia de *jitter*

En el Capítulo 5 se estudió el rendimiento de las diferentes técnicas de análisis de planificabilidad disponibles en MAST. En esta sección estudiaremos el comportamiento de estas técnicas bajo situaciones libres de *jitter*. En primer lugar definimos el conjunto de sistemas que formarán parte del estudio.

8.3.1 Dominio y ejecución del estudio

Para llevar a cabo el estudio nos aprovecharemos del dominio generado en el Capítulo 5 para la comparativa de técnicas de análisis en sistemas con *jitter*, ejecutando esta vez todas las técnicas de análisis con la opción de eliminación del *jitter* activada.

A modo de recordatorio, el sistema base que forma el eje central del dominio del estudio posee las siguientes características:

- Número de flujos e2e: 10.
- Número de actividades por flujo e2e (longitud): 10.
- Número de recursos procesadores: 5.
- Localización de las actividades: Si es posible, no se repite ningún recurso procesador dentro de cada flujo (pseudo-aleatoria).
- Plazos de principio a fin: $D_i=N_i*T_i$
- Periodos: Se seleccionan aleatoriamente en el rango [100, 1000] (Ratio=10), mediante una distribución de probabilidad logarítmico-uniforme (*Log-Uniform*)
- Utilización de las actividades calculadas mediante el algoritmo *UUnifast*.
- Tiempos de ejecución de mejor caso de las actividades igual a 0.
- Todos los procesadores poseen la misma utilización en todo momento.

El *jitter* es un efecto que se propaga de manera longitudinal a lo largo del flujo e2e, por lo tanto su influencia tiene una dependencia directa con el número de actividades que lo componen. Este hecho fue confirmado en el estudio de las técnicas de asignación de parámetros de planificación en ausencia de *jitter* (ver Sección 8.2), donde se comprobó que los beneficios de eliminar el *jitter* se hacen más visibles a medida que los flujos e2e son más largos. Por esta razón en el presente estudio nos concentraremos en observar los resultados obtenidos en las variaciones del número de actividades (longitud) de los flujos e2e.

Otra conclusión del estudio previo fue que las UMP que se obtienen una vez eliminado el *jitter* aumentan significativamente, llegando a utilizaciones planificables cercanas al 100%, dificultando así la visualización de las diferencias entre las diferentes técnicas si hacemos uso de la UMP como criterio de comparación. Por este motivo, para analizar el comportamiento de las técnicas de análisis en ausencia de *jitter* nos fijaremos en los tiempos de respuesta de peor caso.

Para la asignación de parámetros de planificación utilizamos en todos los casos la técnica PD, al ser una técnica sencilla con resultado predecible en todos los casos. Adicionalmente, PD fue la técnica utilizada en la comparativa de técnicas de análisis con *jitter*, por lo que esos resultados pueden volver a utilizarse para completar el presente

estudio. Para obtener resultados con relevancia estadística, en el dominio de estudio se generaron 30 series de utilizaciones para cada combinación de características.

En este estudio añadimos una nueva figura de mérito, con la que cuantificar la reducción de los tiempos de respuesta de peor caso de los flujos e2e una vez que el *jitter* ha sido eliminado. Para ello mostraremos la ratio de los promedios de los tiempos de respuesta de peor caso obtenidos sin *jitter*, entre los obtenidos con *jitter*. A esta ratio la llamamos *Ratio*(*J*), y la formalizamos en la siguiente ecuación:

$$Ratio(J) = \frac{R(X)_{\text{sin } jitter}}{R(X)_{con\ jitter}} \tag{8-5}$$

donde *R(X)* es el promedio de los tiempos de respuesta de peor caso de todos los flujos e2e obtenidos con la técnica de análisis *X* (definido en la Sección 5.2).

Otra comparación que realizaremos será la de mostrar la diferencia en la reducción de los tiempos de respuesta de peor caso que se obtiene al utilizar dos métodos distintos: mediante la eliminación del *jitter*, y mediante la utilización de una estimación de los tiempos de ejecución de mejor caso, que contribuyen a reducir el *jitter* contabilizado en el análisis. Para ello, mostraremos la ratio de los promedios de los tiempos de respuesta de peor caso obtenidos sin *jitter* entre los obtenidos con *jitter* y *Cb=C*. A esta ratio la llamamos *Ratio*(*Cb*), y se formaliza en la siguiente ecuación:

$$Ratio(Cb) = \frac{R(X)_{\text{sin } jitter}}{R(X)_{con\ jitter\ y\ Cb=C}} \tag{8-6}$$

donde *R(X)* es el promedio de los tiempos de respuesta de peor caso de todos los flujos e2e obtenidos con la técnica de análisis *X* (definido en la Sección 5.2).

8.3.2 Planificación FP

Estudiamos el comportamiento de las técnicas de análisis de planificabilidad para sistemas FP con la opción de emulación de eliminación del *jitter* activada. Las técnicas bajo estudio son el análisis *holistic, offset based, offset based slanted* y *offset based w/pr*. No incluimos el análisis basado en *offsets* de fuerza bruta (*offset based brute force)*, ya que sería computacionalmente intratable para el que conjunto de sistemas generado, y por otra parte, ya hemos determinado que para sistemas con *jitter*, su rendimiento se sitúa en un punto intermedio entre *offset based slanted* y *offset based w/pr.*

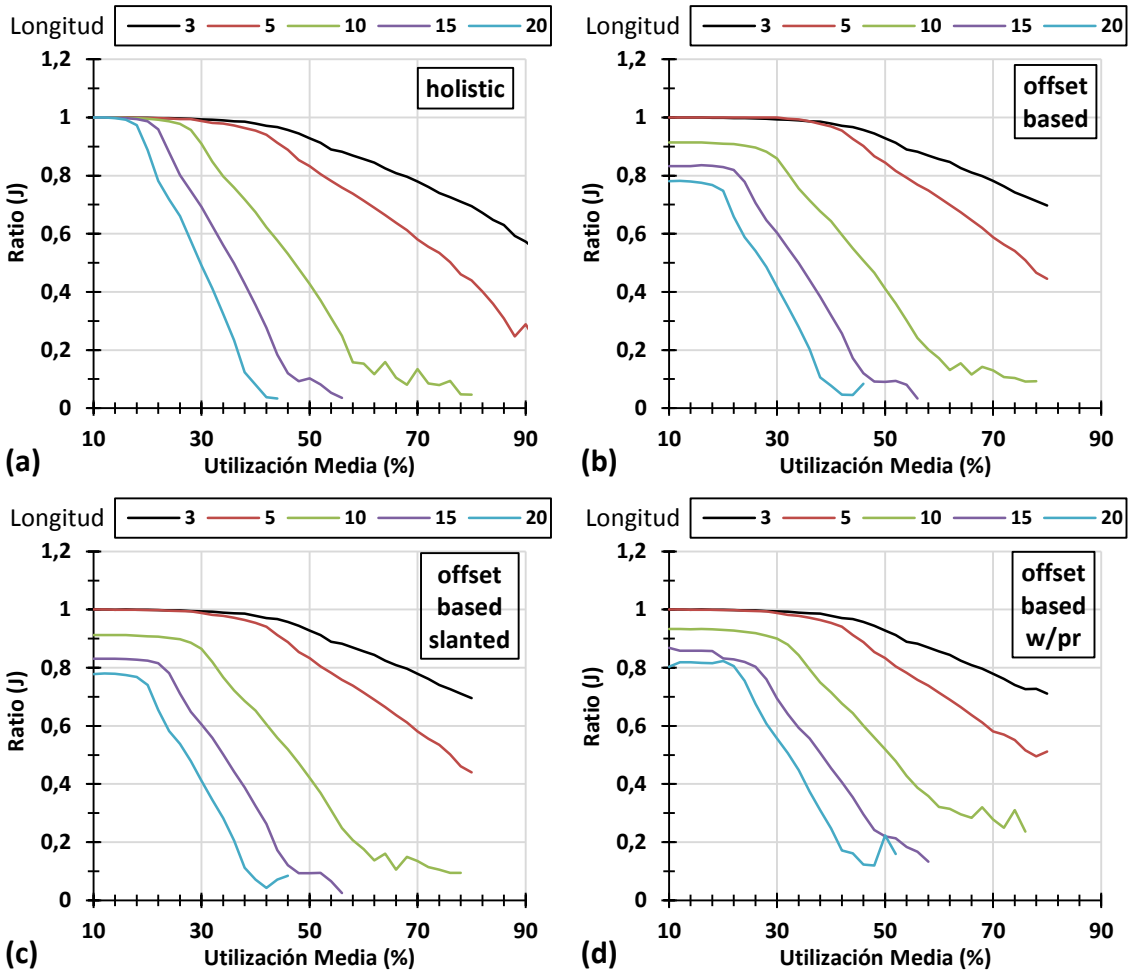


Figura 8-14 *Ratio*(*J*), para distintas longitudes de los flujos e2e en sistemas FP, Cb_{ij}=0, y las técnicas de análisis **(a) *holistic*, (b) *offset based*, (c) *offset based slanted*, y (d) *offset based w/pr.***

En la Figura 8-14 mostramos la *Ratio*(*J*) (definida en la Ecuación (8-5)), para cada técnica de análisis y sistemas con 5 longitudes distintas en los flujos e2e. Un valor de la ratio menor que 1 indica una reducción en el promedio de los tiempos de respuesta de peor caso al eliminar el *jitter*.

En esta misma figura observamos un comportamiento distinto entre la técnica *holistic* y las basadas en *offsets*. En el análisis *holistic* comprobamos que los beneficios de eliminar el *jitter* sólo se manifiestan a partir de una cierta utilización media en el sistema. Esta utilización depende de la longitud de los flujos, siendo más alta a medida que los flujos son más cortos. En los análisis basados en *offsets,* se observan reducciones en los tiempos de respuesta de peor caso para todo el rango de utilizaciones analizado en los flujos e2e con longitud 10 y mayores, es decir, cuando el flujo e2e tiene varias actividades en el mismo procesador. Cuando no se repite ningún procesador, la reducción en los tiempos de respuesta de peor caso de los análisis *holistic* y los basados en *offsets* son similares. Adicionalmente, en la Figura 8-14 observamos que, independientemente de la técnica de análisis, a partir una cierta utilización, el tiempo de respuesta de peor caso medio disminuye drásticamente, y de forma más acusada para sistemas con flujos e2e más largos.

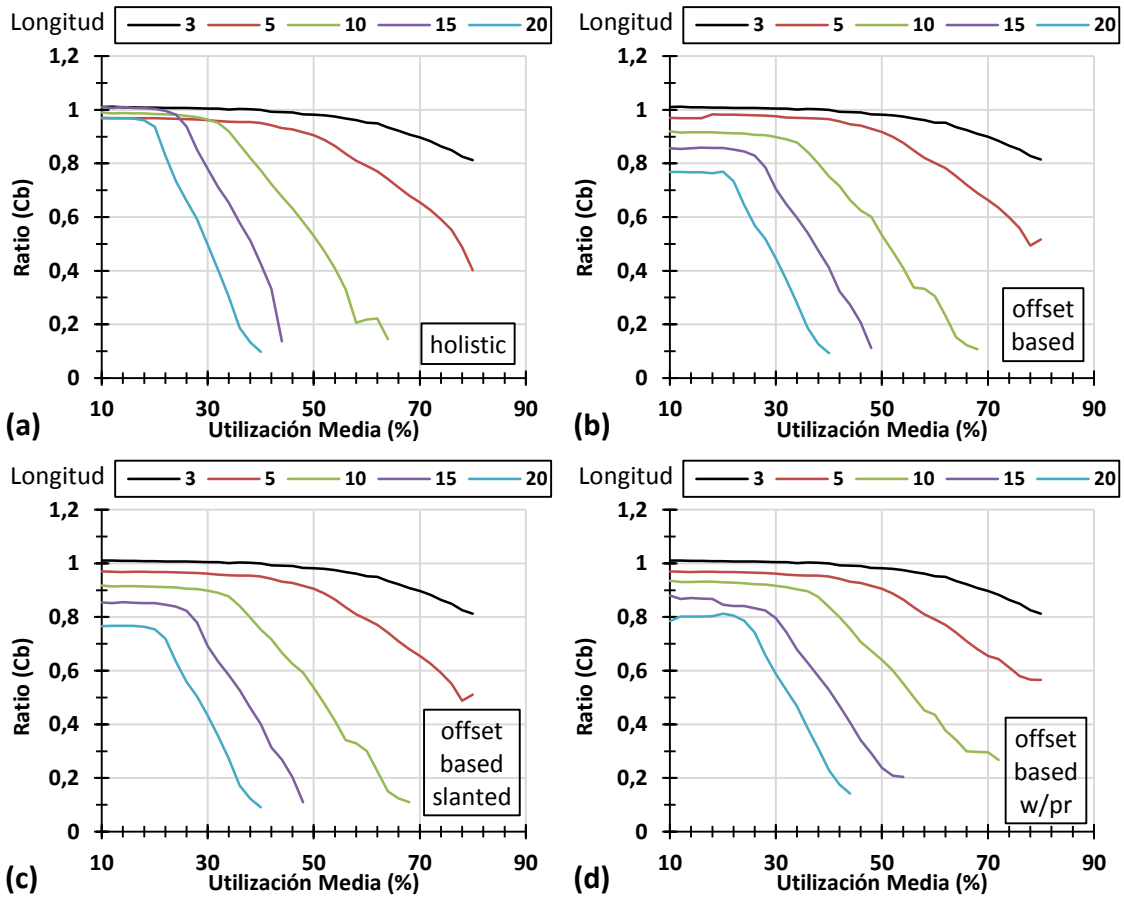


Figura 8-15 *Ratio* (*Cb*) para distintas longitudes de los flujos e2e en sistemas FP, y las técnicas de análisis **(a) holistic, (b) *offset based*, (c) offset based slanted, y (d) *offset based w/pr***

Completamos el estudio comparando la disminución en los tiempos de respuesta de peor caso utilizando dos métodos distintos: eliminando el *jitter*, o utilizando una estimación de los tiempos de ejecución de mejor caso. Para ello, en la Figura 8-15 mostramos la *Ratio*(*Cb*) (definida en la Ecuación (8-6)) de las diferentes técnicas de análisis utilizadas. Observamos que para los flujos e2e más cortos (de longitud 3), y utilizaciones bajas, utilizar una estimación de los tiempos de ejecución de mejor caso produce tiempos de respuesta de peor caso ligeramente inferiores a los obtenidos al eliminar el *jitter* por completo. Este resultado ocurre en las cuatro técnicas de análisis verificadas.

Finalizamos la comparativa cuantificando la reducción de los tiempos de respuesta de peor caso de los análisis basados en *offsets* con respecto a *holistic* para sistemas sin *jitter*. Para ello, mostraremos los promedios de los tiempos de respuesta de peor caso normalizados a los obtenidos por *holistic* (*R_norm (holistic),* definido en la Sección 5.2). Estos resultados los mostramos en la Figura 8-16, para diferentes longitudes en los flujos e2e, en la que se comprueba que, cuando no se producen repeticiones en los flujos e2e, esto es, con longitudes 3 y 5, todos los análisis obtienen los mismos resultados, y por tanto la ratio se mantiene constante a 1 en todo el rango de utilizaciones. Sin embargo, para flujos más largos se observa que los tiempos de respuesta de peor caso obtenidos por las técnicas basadas en *offsets* son menores. Los tiempos de respuesta de peor caso obtenidos con *offset based* son entre 0,88 y 0,75 veces los obtenidos por *holistic*. Ratios similares se obtienen con *offset based slanted*, mientras que con *offset based w/pr* se obtienen ratios entre 0,83 y 0,65. Las mayores reducciones con respecto *holistic* en los

tiempos de respuesta de peor caso obtenidas por los análisis basados en *offsets* se obtienen con los flujos más largos y utilizaciones más bajas.

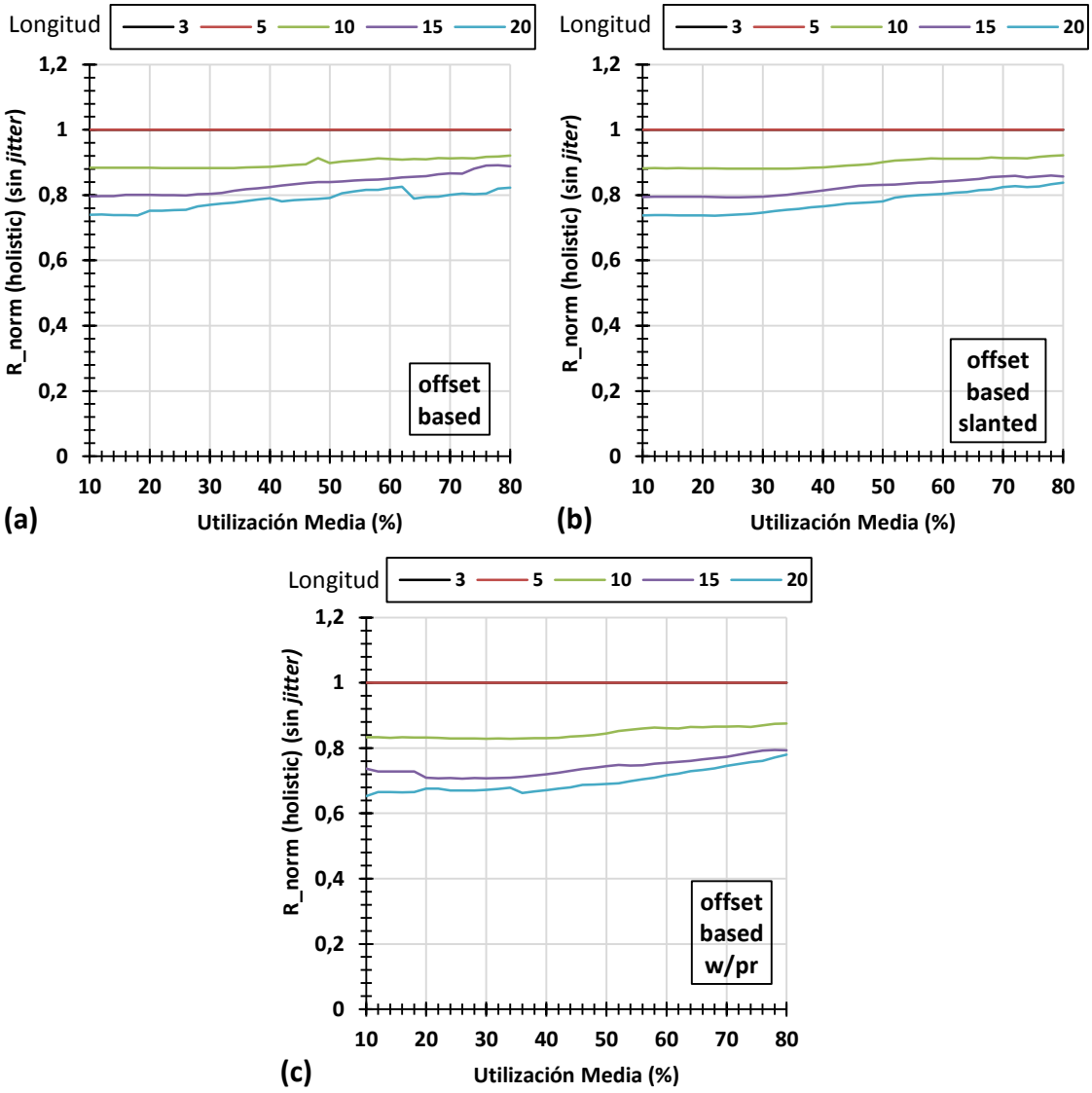


Figura 8-16 Tiempos de respuesta de peor caso medios obtenidos anulando el *jitter*, normalizados a *holistic*, para distintas longitudes de los flujos e2e en sistemas FP, y las técnicas de análisis **(a) *offset based*, (b) *offset based slanted*, y (c) *offset based w/pr***

8.3.3 Planificación GC-EDF

Repetimos el mismo proceso seguido previamente para FP, pero esta vez para sistemas planificados con GC-EDF. Puesto que para GC-EDF sólo tenemos disponibles dos técnicas de análisis *holistic* y *offset based*, el presente estudio resulta comparativamente más sencillo.

En la Figura 8-17 se representa la *Ratio*(*J*) (definida en la Ecuación (8-5)) de las dos técnicas de análisis, y flujos e2e con diferentes longitudes. En la figura observamos que esta ratio disminuye con la utilización, excepto para las utilizaciones más bajas de los flujos e2e más largos.

Para flujos e2e cortos (longitudes 3 y 5), la ratio se mantiene en 1 para utilizaciones inferiores al 30%. A partir de esta utilización las ratios disminuyen de forma gradual hasta un mínimo registrado de un 0,9 y 0,8 para longitudes 3 y 5 respectivamente, y ambas técnicas. Con flujos más largos (10, 15 y 20) las ratios son menores a 1 en todo el rango de utilizaciones, aunque se mantiene el mismo comportamiento gradual. En términos aproximados, a partir de una longitud de 5, por cada 5 actividades añadidas a los flujos, las ratios disminuyen aproximadamente en 0,1. Aunque la pauta general observada con ambas técnicas es similar, las ratios reportadas con *offset based* son aproximadamente 0,05 puntos inferiores a las registradas por *holistic*, esto es, la reducción de los tiempos de respuesta de peor caso al eliminar el *jitter* es mayor con el análisis *offset based*.

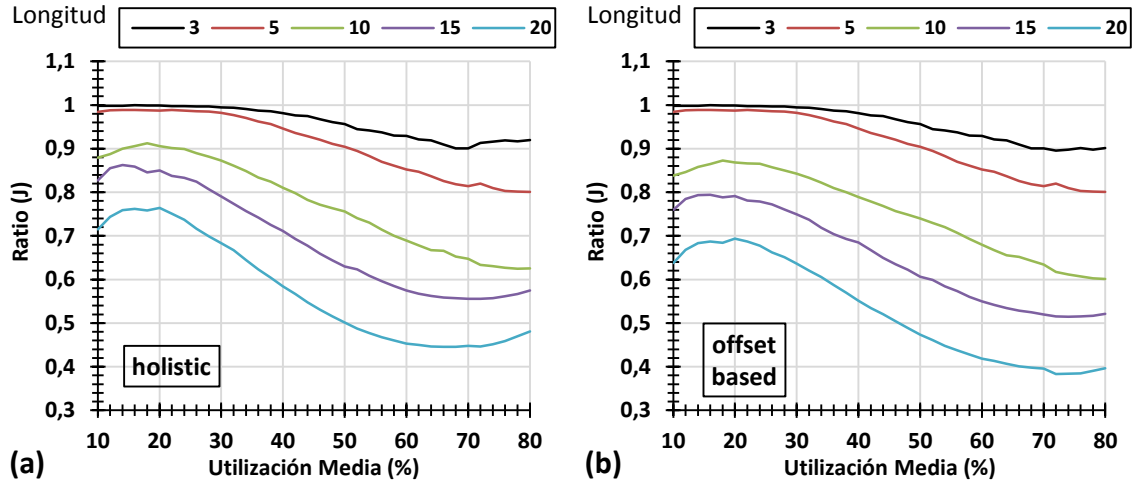


Figura 8-17 *Ratio*(*J*), para distintas longitudes de los flujos e2e en sistemas GC-EDF, Cb_{ij}=0, y las técnicas de análisis **(a) *holistic*, y (b) *offset based*.**

En la Figura 8-18 mostramos la *Ratio*(*Cb*) (definida en la Ecuación (8-6)) de las diferentes técnicas de análisis utilizadas. Al igual que lo observado en sistemas FP, para los flujos e2e más cortos y utilizaciones bajas, se obtienen unas ratios mayores que 1. Esto indica que utilizar una estimación de los tiempos de ejecución de mejor caso produce tiempos de respuesta de peor caso ligeramente inferiores a los obtenidos al eliminar el *jitter* por completo para los flujos e2e más cortos y utilizaciones más bajas.

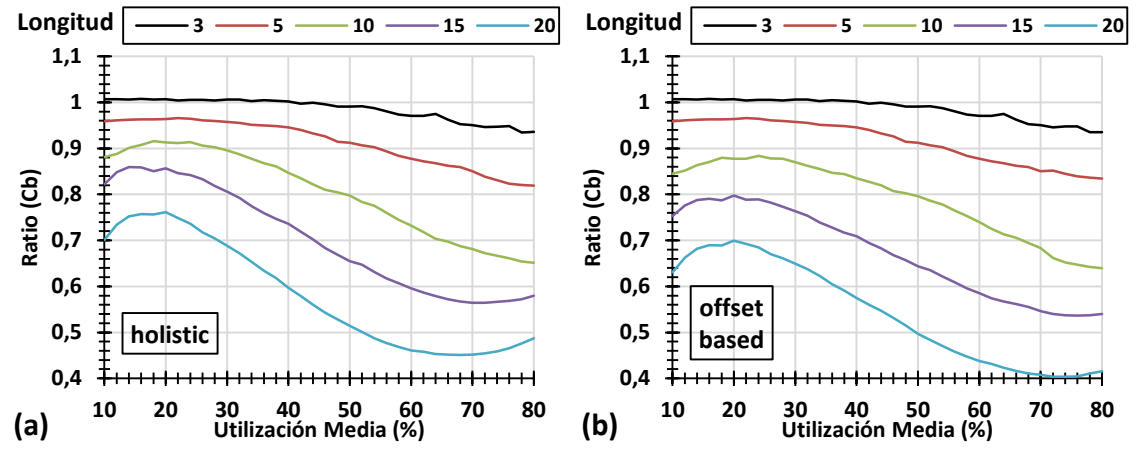


Figura 8-18 *Ratio* (*Cb*), para distintas longitudes de los flujos e2e en sistemas GC-EDF, y las técnicas de análisis **(a) holistic y (b) *offset based***

Para concluir el estudio en GC-EDF mostramos, para sistemas sin *jitter*, el promedio de los tiempos de respuesta de peor caso normalizados a *holistic* (*R_norm(holistic)*). Una

ratio inferior a 1 implica que *offset based* obtuvo tiempos de respuesta de peor caso inferiores a *holistic*. En la Figura 8-19 mostramos esta comparación. Para flujos e2e de longitud 3 y 5 se observa una ratio constante de 1 en el rango de utilizaciones estudiado, por lo tanto ambas técnicas son equivalentes en esta situación. Mayores longitudes de los flujos e2e muestran cómo el análisis *offset based* obtiene tiempos de respuesta de peor caso con ratios de 0,86 y 0,77 para longitudes en el rango de 10 a 20.

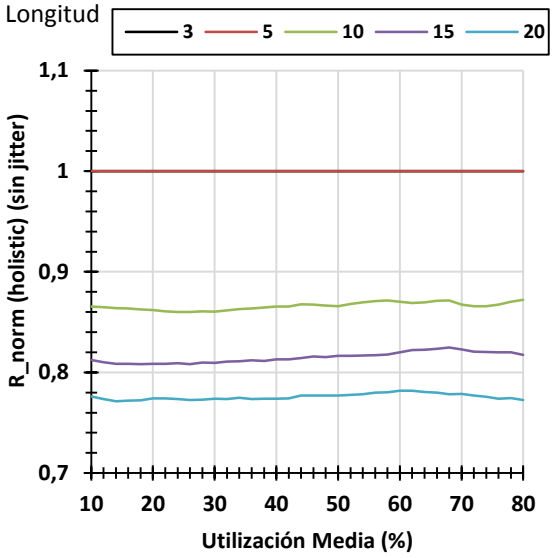


Figura 8-19 Tiempos de respuesta de peor caso medios obtenidos por la técnica *offset based* anulando el *jitter*, normalizados a *holistic*, para distintas longitudes de los flujos e2e en sistemas FP

8.3.4 Planificación LC-EDF

La planificación LC-EDF sólo tiene disponible una técnica de análisis, *holistic*. Por lo tanto, el estudio se limitará a cuantificar la reducción en los tiempos de respuesta de peor caso al eliminar el *jitter*, en esta única técnica de análisis.

En la Figura 8-20 se representa la *Ratio*(*J*) (definida en la Ecuación (8-5)) para las distintas longitudes de los flujos e2e generadas. Para flujos e2e cortos (longitudes 3 y 5) y utilizaciones inferiores al 30%., la ratio se sitúa en un valor próximo a 1. A partir del 30% de utilización media, las ratios disminuyen de forma gradual hasta un mínimo registrado de un 0,65 y 0,48 para longitudes 3 y 5 respectivamente. Con flujos más largos (10, 15 y 20) las ratios son menores a 1 en todo el rango de utilizaciones, alcanzando una ratio mínima de 0,12% para flujos e2e de longitud 20.

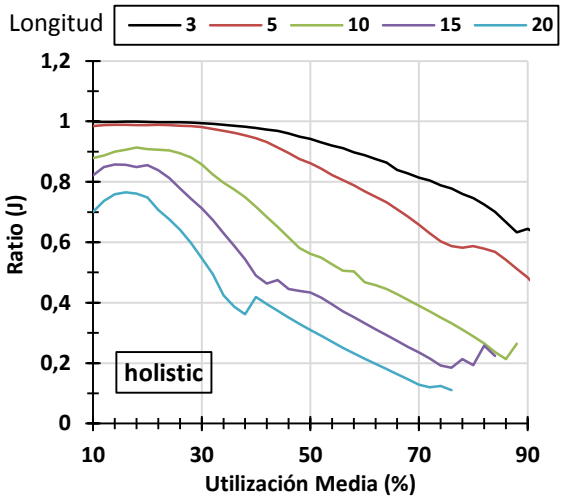


Figura 8-20 *Ratio*(*J*) de la técnica de análisis *holistic*, para distintas longitudes de los flujos e2e en sistemas LC-EDF, y Cb_{ij}=0

En la Figura 8-21 mostramos la *Ratio*(*Cb*) (definida en la Ecuación (8-6)) del análisis *holistic* en sistemas LC-EDF. Al igual que lo observado previamente en sistemas GC-EDF y FP, para los flujos e2e más cortos y utilizaciones bajas, las ratios observadas son mayores que 1. Esto es un indicativo de que estimar los tiempos de ejecución de mejor caso produce tiempos de respuesta de peor caso ligeramente inferiores a los obtenidos al eliminar el *jitter* por completo, en sistemas con utilizaciones bajas y flujos e2e más cortos.

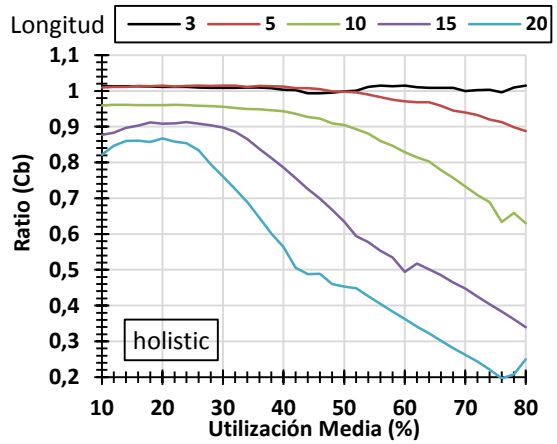


Figura 8-21 Ratio (Cb), para distintas longitudes de los flujos e2e en sistemas GC-EDF, y la técnica de análisis *holistic*

Los resultados presentados aquí para LC-EDF son un claro indicativo de que esta política de planificación se ve más afectada por el *jitter* que GC-EDF. Esta mayor dependencia con el *jitter* es en parte explicada en el hecho que los plazos locales de planificación para LC-EDF toman como referencia la activación retrasada por el *jitter* de la propia actividad, mientras que en GC-EDF el plazo global de planificación es independiente de este retraso en la activación.

8.4 Conclusiones

El *jitter* es un fenómeno que afecta negativamente a la planificabilidad de los sistemas distribuidos. En este capítulo planteamos un estudio para observar cómo responden las diferentes técnicas estudiadas a lo largo de la tesis, en situaciones en las que el *jitter* se ha eliminado.

En primer lugar, en la Sección 8.1 propusimos un método que modifica el funcionamiento de las técnicas de análisis, de forma que emulen sistemas libres de *jitter*. Esta modificación consiste en asignar a las actividades *offsets* heredados cuyos valores son los tiempos de respuesta de peor caso de las actividades precedentes. Este cálculo se realiza de manera iterativa durante el proceso de análisis.

Para el estudio de las técnicas de asignación de parámetros de planificación se eligió un conjunto que simplifica el utilizado en el Capítulo 6 para sistemas con *jitter*. Combinando los sistemas generados para FP, GC-EDF y LC-EDF, el dominio queda compuesto por 211200 sistemas. Este estudio en particular requirió 35 días de tiempo de CPU, que ejecutados en el *cluster* de computación, se pudo completar en unas 5 horas. Los resultados se pueden resumir en los siguientes puntos, para cada política de planificación:

- FP: mientras que en sistemas con *jitter*, las técnicas que alcanzan mayor UMP media son EQS, EQF y HOSPA, cuando el *jitter* se elimina, son ahora las técnicas HOSPA y su asignación inicial PD las técnicas que mejores resultados obtienen, especialmente cuando las cargas son calculadas con *UUnifast.*
- GC-EDF: cuando los sistemas poseen *jitter*, HOSPA y PD son las técnicas que mayores UMP medias obtienen. Cuando se elimina el *jitter*, HOSPA y PD obtienen los mejores resultados con cargas de tipo *UUnifast*, mientras que ED y UD obtienen la mayores UMP medias con cargas calculadas con *SCALE-WCET.*
- LC-EDF: En sistemas libres de *jitter*, observamos que los algoritmos PD y HOSPA son los que mejores resultados obtienen cuando las cargas son generadas por *UUnifast*. Para las cargas de tipo *SCALE-WCET*, las diferencias entre las diferentes técnicas se reducen, y las tendencias no están muy definidas. También se comprobó que la aplicación de las modificaciones propuestas en esta tesis, que llamamos LC-EDF-DS y LC-EDF-GSD, empeoran los resultados en los sistemas libres de *jitter* y cargas *UUnifast*, mientras que con cargas *SCALE-WCET* se observan mejoras en las UMP medias para plazos de principio a fin D_i=T1.

En la comparación de los resultados obtenidos entre las políticas de planificación se comprueba que, al igual que con sistemas con *jitter*, si eliminamos el *jitter* la política con la que se alcanzan mayores UMP medias es GC-EDF, salvo para los plazos de principio a fin más pequeños, en los que FP posee una limitada ventaja.

En la segunda parte del capítulo (Sección 8.3), se estudió el comportamiento de las diferentes técnicas de cálculo de tiempos de respuesta en situaciones libres de *jitter*. Para este estudio se utilizó el conjunto de ejemplos generado en el Capítulo 5. Combinando los sistemas generados para FP, GC-EDF y LC-EDF, el dominio quedaba compuesto por un total de aproximadamente un millón de sistemas. En total, la ejecución de este estudio necesitó aproximadamente 400 días de tiempo de CPU, que se convirtieron en algo menos de 2 días de ejecución en 250 procesadores del *cluster* Tres Mares.

En los resultados obtenidos se cuantificó la acusada reducción de los tiempos de respuesta de peor caso al eliminar el *jitter*, para las tres políticas de planificación. Se observan tiempos de respuesta de peor caso que son hasta 0,2 veces los obtenidos con *jitter*. Una situación especial que se ha comprobado es que para sistemas con flujos e2e más cortos, y utilizaciones medias en el sistema bajas, utilizar una estimación del tiempo de mejor caso de las actividades puede producir mayores reducciones en los tiempos de respuesta de peor caso, que eliminar el *jitter* por completo.

9 Conclusiones y Trabajo Futuro

En este capítulo se presentan las conclusiones obtenidas tras la realización de esta tesis. En primer lugar se revisarán los objetivos planteados en el Capítulo 1, y se detallará el modo en el que fueron abordados. A continuación se destacarán las principales contribuciones de este trabajo, y se aventurarán los posibles desarrollos futuros por los que se pueden continuar los trabajos realizados en esta tesis.

9.1 Revisión de los objetivos

En el capítulo introductorio de esta tesis se presentaron los conceptos básicos que rigen los sistemas de tiempo real, con un especial énfasis en los sistemas distribuidos. Se presentaron dos de los problemas fundamentales que se deben tratar en este tipo de sistemas: el análisis de planificabilidad, y la asignación de parámetros de planificación. En torno a estos dos problemas nos planteamos cuatro objetivos que fueron desarrollados a lo largo de esta tesis:

1. Soporte de sistemas heterogéneos FP+EDF.
2. Uso de un supercomputador para la evaluación de técnicas de análisis de planificabilidad y asignación de parámetros de planificación.
3. Estudio de la influencia del *jitter* en sistemas distribuidos de tiempo real.
4. Optimización en la asignación de plazos locales de planificación en sistemas LC-EDF.

En primer lugar, en el Capítulo 3 abordamos el primero de los objetivos, planteando dar soporte a sistemas distribuidos heterogéneos, tanto para el análisis de planificabilidad,

como para la asignación de parámetros de planificación. En cuanto al análisis de planificabilidad, propusimos una metodología que adaptaba las técnicas existentes para sistemas homogéneos. Por otro lado, para la asignación de parámetros de planificación formalizamos un método para transformar repartos de plazo de principio a fin en plazos de planificación, y utilizando estos principios, propusimos además la técnica HOSPA, para la asignación y optimización de parámetros de planificación en sistemas heterogéneos. Todas las propuestas de este capítulo se implementaron dentro del entorno de herramientas de análisis y optimización MAST.

El segundo de los objetivos se desarrolló en varios capítulos. En primer lugar, en el Capítulo 4 se definió la herramienta GEN4MAST, con la que es posible automatizar la generación y ejecución de test para la evaluación de técnicas de análisis y optimización de sistemas de tiempo real, haciendo uso de un supercomputador. En Capítulo 5 se utilizó la herramienta GEN4MAST para llevar a cabo un estudio comparativo de las técnicas de análisis de planificabilidad para sistemas distribuidos aplicables a las tres políticas de planificación disponibles en MAST: FP, GC-EDF y LC-EDF. En el Capítulo 6 se realizó un estudio comparativo de las técnicas de asignación de parámetros de planificación para las diferentes políticas de planificación, también gracias a la disponibilidad de GEN4MAST.

Para la consecución del tercero de los objetivos (estudio de la influencia del *jitter*), desarrollado en el Capítulo 8, en primer lugar propusimos un método para que las técnicas de análisis de planificabilidad emularan situaciones libres de *jitter*, asignando *offsets* heredados a las actividades. Haciendo uso de este método, y de la herramienta GEN4MAST, llevamos a cabo un estudio comparativo de las técnicas de análisis y de asignación de parámetros de planificación, para las tres políticas de planificación, y en situaciones libres de *jitter*.

El último de los objetivos, la optimización de la asignación de parámetros de planificación en LC-EDF, surge de la observación del pobre rendimiento observado para esta política de planificación. En el Capítulo 7 estudiamos el origen de este bajo rendimiento de LC-EDF, y a partir de estos resultados propusimos dos modificaciones que se pueden aplicar a las técnicas existentes para asignar parámetros de planificación en LC-EDF: LC-EDF-DS (LC-EDF *Deadline Scaling*) y LC-EDF-GSD (LC-EDF *with Global Scheduling Deadlines*) Ambas modificaciones consiguen mejorar los resultados de las técnicas sobre las que se aplican.

9.2 Contribuciones de esta tesis

El punto de partida de esta tesis era la disponibilidad de un conjunto de técnicas de análisis de planificabilidad y de asignación de parámetros de planificación para sistemas distribuidos aplicables a sistemas planificados por prioridades fijas (FP) o dinámicas (EDF). En los capítulos introductorios se desarrollaron los principios en los que se basa el funcionamiento de estas técnicas y las características que las diferencian. Además, la implementación de estas técnicas se presentó en el contexto del entorno de modelado, análisis y optimización MAST, que constituye un instrumento fundamental en el presente trabajo.

Para sistemas distribuidos planificados por FP y sistemas multiprocesadores, los conceptos relativos a los planificadores están claramente definidos en la bibliografía. Sin

embargo, para sistemas distribuidos con planificadores EDF, este cuerpo teórico no estaba definido con la misma extensión. Así pues, la primera aportación de esta tesis ha sido identificar dos tipos distintos de planificación EDF en sistemas distribuidos, en función de la disponibilidad de sincronización en los relojes: la planificación GC-EDF cuando existe tal sincronización, y LC-EDF cuando no se dispone de ella. Adicionalmente, se han redefinido los algoritmos de asignación de plazos de planificación en función del tipo de plazo que generan: plazo local de planificación para LC-EDF, o plazo global de planificación para GC-EDF.

Una característica de las técnicas existentes para el análisis y optimización de sistemas distribuidos de tiempo real es que son aplicables a sistemas homogéneos, en los que todos los recursos procesadores utilizan el mismo algoritmo de planificación, sólo FP o sólo EDF. Los sistemas distribuidos actuales tienden hacia la heterogeneidad, de manera que cada vez es más frecuente encontrar sistemas distribuidos en los que tengan que convivir subsistemas, formados por diferentes procesadores o redes de comunicaciones, en los que se deban utilizar políticas de planificación diferentes. Aunque las técnicas existentes, y que se encontraban integradas en MAST, eran sólo aplicables a sistemas homogéneos, su visión holística del sistema hacía pensar que se podría definir un método con el que integrarse para poder operar sobre sistemas heterogéneos, en los que los diferentes recursos procesadores pudieran combinar planificadores FP y EDF. Así, presentamos un método composicional con el que adaptar las técnicas de análisis integradas en MAST para el soporte de sistemas heterogéneos (ver Capítulo 3). Éste método se basa en la definición del análisis de cada actividad de un flujo e2e por separado, y de los mecanismos mediante los cuales se propagan los resultados. Para la asignación de parámetros de planificación en sistemas heterogéneos, se definió un proceso para transformar el reparto de plazos de principio a fin en distintos tipos de plazos de planificación o de prioridades fijas. Sobre esta base se fusionaron los algoritmos HOPA y HOSDA, aplicables a sistemas homogéneos FP y EDF respectivamente, para crear el nuevo algoritmo heurístico HOSPA. Esta nueva metodología fue integrada en la herramienta MAST.

Otro de los objetivos planteados en esta tesis consistía en llevar a cabo un estudio comparativo lo más extenso y representativo posible de todas las técnicas de análisis de planificabilidad y asignación de parámetros de planificación integradas en MAST. El carácter masivo del estudio requería idear una nueva herramienta con la que poder llevarlo a cabo de manera eficiente. Para ello presentamos la herramienta GEN4MAST (Capítulo 4) para la generación de sistemas de test y su posterior ejecución y procesado de resultados utilizando MAST. La principal ventaja de GEN4MAST frente a otras herramientas similares anteriores, es que puede hacer uso de un *cluster* de computación. Esta característica abre la puerta a llevar a cabo estudios más exhaustivos en espacios de tiempo razonables. Aunque GEN4MAST se ideó inicialmente para llevar a cabo los estudios propuestos en este trabajo, ha sido utilizado también con éxito por otros investigadores para llevar a cabo estudios masivos similares sobre nuevos algoritmos. [DIA14]. Tras presentar GEN4MAST, realizamos un estudio de los diferentes métodos de generación de cargas que implementa, y se comprobó la influencia que éstos tienen en los resultados, verificando que para sistemas con la misma utilización, los métodos con mayor sesgo producen sistemas con utilizaciones máximas planificables inferiores.

Utilizando GEN4MAST, se llevó a cabo un estudio comparativo masivo de las técnicas de análisis y optimización. Este estudio aporta un marco global con el que comparar el rendimiento de éstas, en el que el conjunto de situaciones tratadas intentan abarcar un abanico suficientemente amplio como para que un diseñador pueda usar los

resultados como guía para seleccionar la técnica que mejor se adapta al sistema concreto con el que está trabajando.

En el Capítulo 5 se llevó a cabo el estudio comparativo de las técnicas de análisis de planificabilidad para las distintas políticas de planificación: FP, GC-EDF y LC-EDF. Este estudio se llevó a cabo en un conjunto de sistemas que exploraban diferentes combinaciones de las características que definen su arquitectura, siendo éstas principalmente: el número de recursos procesadores, el número de flujos e2e, el número de actividades por flujo e2e, y los rangos de periodos. En todas las políticas de planificación se observó que aumentar el número de actividades de los flujos e2e, para una misma utilización en el sistema, induce un claro detrimento en la planificabilidad de los sistemas. Este es un indicativo del efecto negativo producido por el *jitter* de activación de las actividades en los flujos e2e.

Para los sistemas planificados por FP (Sección 5.3), se comprobó que la ventaja de las técnicas de análisis basadas en *offsets* sobre la técnica holística sólo se manifiesta cuando los flujos e2e repiten algún recurso procesador durante su recorrido. En términos generales se comprobó que el análisis basado en *offsets* con optimizaciones de relaciones de precedencia es el que obtiene un rendimiento superior al resto, incluso por encima del análisis de fuerza bruta. El análisis *offset based slanted* se mostró ligeramente superior a *offset based*.

Con planificación GC-EDF (Sección 5.4), y al igual que para FP, se comprobó que las mejoras en la cota de los tiempos de respuesta de peor caso de las técnicas basadas en *offsets* sólo se producen cuando los flujos e2e repiten algún recurso procesador, y son mayores para un mayor número de repeticiones de recursos. Adicionalmente se observó que GC-EDF obtiene utilizaciones máximas planificables próximas al 100% cuando los plazos de principio a fin de los flujos e2e tienen como valores sus periodos multiplicados por el número de actividades que lo componen. Este comportamiento de GC-EDF en sistemas distribuidos es similar al demostrado por EDF en sistemas monoprocesadores con actividades independientes, en los que para plazos iguales a los periodos, la condición suficiente de planificabilidad es que la utilización del sistema no sea superior al 100%.

El estudio de las técnicas de análisis para LC-EDF (Sección 5.5) sólo incluía la técnica holística, que era la única disponible, para la que se comprobó que al contrario que en el resto de políticas de planificación, el rendimiento del análisis holístico en LC-EDF se ve influenciado por el número de recursos procesadores en el sistema, observando un empeoramiento para sistemas con más recursos procesadores.

En el Capítulo 6 se llevó a cabo la comparativa del rendimiento de las diferentes técnicas de asignación de parámetros de planificación. Para ello se creó un nuevo conjunto de sistemas que abarcaban escenarios a los que típicamente pueden enfrentarse estas técnicas. En la primera parte del estudio se buscaron los valores más apropiados para los parámetros de configuración del algoritmo HOSPA. Con estos valores, se efectuó el estudio comparativo de las diferentes técnicas en cada política de planificación. Para la política FP se constató que las técnicas EQF y HOSPA son las que mejores resultados obtienen en términos generales. En GC-EDF es reseñable el rendimiento del algoritmo PD, que reparte los plazos de principio a fin de forma proporcional a los tiempos de ejecución de peor caso. Este algoritmo sólo se vio superado ligeramente por el algoritmo HOSPA, que utiliza PD como su asignación inicial. En cuanto a la planificación LC-EDF se observó un resultado inesperado, comprobando que bajo ciertas circunstancias, el algoritmo trivial UD, que asigna todos los plazos locales de planificación con el valor del plazo de principio a fin, obtenía mejores resultados que el algoritmo heurístico HOSPA.

Este resultado inesperado observado en LC-EDF, unido al pobre rendimiento presentado por esta política de planificación en comparación con FP y GC-EDF, nos motivó a estudiar con mayor detalle la asignación de plazos locales de planificación (Capítulo 7). A través del análisis de un ejemplo sencillo, concluimos que, al contrario de lo que podría dictar la intuición, en LC-EDF asignar plazos locales de planificación cuya suma supere al plazo de principio a fin del flujo e2e, puede conllevar una reducción general de los tiempos de respuesta de peor caso. Este hecho rompe con la regla básica utilizada para la asignación de plazos locales de planificación. Así, tomando como base esta observación, propusimos las modificaciones LC-EDF-DS (LC-EDF *Deadline Scaling*) y LC-EDF-GSD (LC-EDF *with Global Scheduling Deadlines*) que producen mejoras de hasta un 7% y un 36% respectivamente en la UMP (Utilización Máxima Planificable) media alcanzada por los algoritmos PD y HOSPA. De la misma manera, esta nueva posibilidad de asignar plazos locales de planificación abría la puerta a la utilización en LC-EDF de técnicas de asignación que en principio sólo eran aplicables a GC-EDF. Así por ejemplo, la técnica EQF para LC-EDF obtenía un 5% más en la UMP media que LC-EDF-GSD aplicado a HOSPA. Mediante las propuestas presentadas en este capítulo, la ventaja en el rendimiento que obtenía la política de planificación GC-EDF sobre LC-EDF se redujo a la mitad.

Para concluir la tesis, se cuantificaron los beneficios de eliminar el *jitter* en sistemas distribuidos FP, GC-EDF y LC-EDF (Capítulo 8). El primer paso fue proponer un método basado en el cálculo de *offsets* heredados para las actividades, que permite emular el efecto de la eliminación del *jitter* en las técnicas de cálculo de tiempos de respuesta. La integración de este método en MAST se vio facilitada por los fundamentos composicionales de las técnicas de análisis para sistemas heterogéneos presentadas en el Capítulo 3. Utilizando este método, se cuantificó el aumento en el rendimiento de las diferentes técnicas de análisis de planificabilidad y asignación de parámetros de planificación en situaciones libres de *jitter*. Una primera observación extraída de estos estudios es que para sistemas sin *jitter*, aumentar el número de actividades en los flujos e2e no conduce a un empeoramiento en la planificabilidad de los sistemas, llegándose incluso a producir un ligero aumento en las utilizaciones máximas planificables alcanzadas.

En el estudio de las técnicas de asignación de parámetros de planificación en sistemas libres de *jitter* (Sección 8.2) se observó un cambio de tendencia con respecto a los sistemas con *jitter*. En sistemas con planificación FP y en presencia de *jitter*, la técnica EQF obtenía buenos resultados, pero no sucede lo mismo si se elimina el *jitter*. En esta situación fue HOSPA el algoritmo claramente superior, situación similar a lo observado en LC-EDF. Para esta última política de planificación se comprobó que las propuestas del Capítulo 7 (modificaciones LC-EDF-DS y LC-EDF-GSD) aplicadas en sistemas sin *jitter* sólo producen mejoras con cargas generadas con *SCALE-WCET*. Por último, en GC-EDF observamos que, al igual que para sistemas con *jitter*, HOSPA y PD son en general los mejores algoritmos para situaciones libres de *jitter*, aunque se encuentran situaciones en las que UD y ED obtienen cierta ventaja (en sistemas con cargas de tipo *SCALE-WCET* y plazos de principio a fin pequeños).

En el estudio comparativo de las técnicas de análisis en sistemas libres de *jitter* (Sección 8.3) se muestran descensos acusados en los tiempos de respuesta de peor caso al eliminar el *jitter,* en todas las políticas de planificación. En los sistemas con planificación FP se observa una utilización media en el sistema a partir de la cual los tiempos de respuesta de peor caso descienden drásticamente en comparación a los obtenidos con *jitter*. En los sistemas planificados por GC-EDF y LC-EDF no se observa este punto de

inflexión. Independientemente de la política de planificación, la ventaja de las técnicas basadas en *offsets* se sigue manifestando cuando los flujos e2e repiten algún recurso procesador.

Para terminar, damos una idea cuantitativa de la magnitud de los experimentos mostrados en esta tesis. En total, todos los estudios que aparecen en esta memoria necesitaron aproximadamente 4,5 años de CPU para ejecutarse. Si contabilizamos además los estudios previos realizados y otros más que no aparecen en esta memoria, el tiempo total de ejecución se multiplicaría por cuatro. Resulta evidente por lo tanto que el trabajo realizado en esta tesis no podría haberse llevado a cabo, al menos no con la misma extensión, si no se hubiese dispuesto de un *cluster* de computación y de la herramienta con la que explotar este recurso, GEN4MAST, que se ha desarrollado en esta tesis.

9.3 Trabajo futuro

Durante la finalización de esta tesis se han presentado dos técnicas de análisis de planificabilidad basadas para LC-EDF [DIA14]: una basada en *offsets* y la otra basada en *offsets* con relaciones de precedencia. Una primera extensión del trabajo realizado sería la inclusión de estas nuevas técnicas en el estudio comparativo de técnicas de análisis de planificabilidad realizado en el Capítulo 5.

La inclusión de estas técnicas de análisis de planificabilidad también significa que el análisis holístico ya no es la única técnica disponible en las tres políticas de planificación. Por lo tanto, el estudio comparativo de técnicas de asignación de parámetros de planificación llevado a cabo en el Capítulo 6 podría actualizarse utilizando como técnicas de análisis de planificabilidad las basadas en *offsets* para las tres políticas de planificación FP, GC-EDF y LC-EDF.

En el campo de la asignación de parámetros de planificación se pueden seguir explorando nuevos métodos al margen de los convencionalismos establecidos, que como hemos demostrado en el caso de LC-EDF, limitan la capacidad de optimización. Así por ejemplo, se podrían explorar nuevas interpretaciones de los algoritmos EQS y EQF, u otros similares.

Adicionalmente, en el futuro se puede considerar aumentar el espectro de sistemas que se pueden generar con GEN4MAST, añadiendo algunas de las capacidades de modelado que están presentes en MAST, como por ejemplo el soporte de flujos e2e con múltiples caminos (*fork, join,* etc.), la capacidad de tratar sistemas heterogéneos, o la posibilidad de usar nuevos métodos de generación de las cargas.

No todos los investigadores tienen acceso a un sistema de supercomputación con el que poder explotar todas las capacidades de GEN4MAST. Por ello, podría ser interesante añadir en el futuro soporte a GEN4MAST para ser ejecutado en plataformas comerciales de *cloud computing* en las que el usuario puede alquilar servidores virtuales durante un espacio de tiempo para realizar sus experimentos. Ejemplos de tales servicios son Amazon EC2 [AMEC2] o Microsoft Azure [AZURE].

Bibliografía

[1QBB] IEEE: The Institute of Electrical and Electronics Engineers, “802.1Qbb - Priority-based Flow Control”, http://www.ieee802.org/1/pages/802.1bb.html.

[ABE04] L. Abeni and G.C. Buttazzo, "Resource reservation in dynamic real-time systems," Real-Time Systems, vol. 27, no. 2, págs. 123-167, 2004.

[AFDX] Airlines Electronic Engineering Committee, Aeronautical Radio INC., “ARINC Specification 664P7: Aircraft Data Network, Part 7 - Avionics Full Duplex Switched Ethernet (AFDX) Network,” September 2009.

[ALD01] M. Aldea and M. González Harbour, "MaRTE OS: An Ada kernel for real-time embedded applications," Reliable SoftwareTechnologies - Ada-Europe 2001. Springer Berlin Heidelberg, págs. 305-316, 2001.

[ALU94] R. Alur and D.L. Dill, "A theory of timed automata," Theoretical computer science, vol. 126, no. 2, págs. 183-235, 1994.

[AMEC2] Amazon EC2, página web http://aws.amazon.com/es/ec2/

[ARI629] ARINC 629 Communication System Description Document, D227W003, Rev. B, 1994.

[AUD91] N.C. Audsley, “Optimal priority assignment and feasibility of static priority tasks with arbitrary start times,” Department of Computer Science, University of York, Technical Report YCS-164, December 1991.

[AUD91b] N.C. Audsley, A. Burns, M.F. Richardson, A.J. Wellings, "Hard real-time scheduling: the deadline-monotonic approach, " Proc. of the 8th IEEE Workshop on Real-Time Operating Systems and Software, Atlanta (Georgia), 1991.

[AUD93] N.C. Audsley, A. Burns, M. Richardson, K. Tindell and A.J. Wellings, "Applying new scheduling theory to static priority pre-emptive scheduling," Software Engineering Journal, vol. 8, no. 5, págs. 284-292, 1993.

[AZK11] E. Azketa, J. P. Uribe, M. Marcos, L. Almeida and J. J. Gutiérrez, “Permutational genetic algorithm for the optimized assignment of priorities to tasks and messages in distributed real-time systems,” Proc. of the 8th IEEE Intl. Conference on Embedded Software and Systems (ICESS), Changsha (China), págs. 958-965, 2011

[AZURE] Microsoft Azure, página web http://azure.microsoft.com/

[BAK91] T.P. Baker, "Stack-based scheduling of realtime processes," Real-Time Systems, vol. 3, no. 1, págs. 67-99, 1991.

[BAL08] P. Balbastre, I. Ripoll and A. Crespo, "Minimum deadline calculation for periodic real-time tasks in dynamic priority systems," IEEE Transactions on Computers, vol. 57, no. 1, págs 96-109, 2008.

[BAR90] S. Baruah, L.E. Rosier and R.R. Howell, "Algorithms and complexity concerning the preemptive scheduling of periodic, real-time tasks on one processor," Real-Time Systems, vol. 2, no. 4, págs. 301-324, 1990.

[BAR90b] S. Baruah, A.K. Mok and L.E. Rosier, "Preemptively scheduling hard-real-time sporadic tasks on one processor," Proc. of the 11th Real-Time Systems Symposium (RTSS), págs. 182-190, 1990.

[BAR99] S. Baruah, G. Buttazzo, S. Gorinsky and G. Lipari, "Scheduling periodic task systems to minimize output jitter," Proc. of the Sixth International Conference on Real-Time Computing Systems and Applications (RTCSA), págs. 62-69, 1999.

[BAR08] S. Baruah and N. Fisher, “Non-migratory feasibility and migratory schedulability analysis of multiprocessor real-time systems,” *Real-Time Systems*, vol 39, no. 1-3, págs. 97-122, 2008.

[BER09] M. Bertogna, M. Cirinei and G. Lipari, “Schedulability analysis of global scheduling algorithms on multiprocessor platforms,” IEEE Transactions on Parallel and Distributed Systems, vol 20, no. 4, págs. 553-566, 2009.

[BET94] R. Bettati, "End-to-end scheduling to meet deadlines in distributed systems," Tesis Doctoral, University of Illinois at Urbana-Champaign, 1994.

[BIN03] E. Bini, G.C. Buttazzo and G.M. Buttazzo, "Rate monotonic analysis: the hyperbolic bound," IEEE Transactions on Computers, vol. 52, no. 7, págs. 933-942, 2003.

[BIN05] E. Bini and G.C. Buttazzo, "Measuring the performance of schedulability tests," Real-Time Systems, vol 30, no. 1-2, págs. 129-154, 2005.

[BOU01] J.Y Le Boudec and P. Thiran, “Network calculus: a theory of deterministic queuing systems for the internet,” Springer, 2001.

[BUR91] A. Burns, ‘‘Scheduling Hard Real-Time Systems: A Review,’’ Software Engineering Journal vol. 6, no. 3, págs. 116-128, 1991.

[BUR06] A. Burns and G. Baxter, "Time bands in systems structure," In Structure for Dependability: Computer-Based Systems from an Interdisciplinary Perspective, Springer London, págs. 74-88, 2006.

[BUR09] A. Burns, A.J. Wellings and F. Zhang, “Combining EDF and FP Scheduling: Analysis and Implementation in Ada 2005,” Proc. of the 14th Ada-Europe International Conference on Reliable Software Technologies, 2009.

[BUR09b] A. Burns, and A.J. Wellings, “Real-Time Systems and Programming Languages: Ada, Real-Time Java and C/Real-Time POSIX,”,. Addison-Wesley Educational Publishers Inc, 2009.

[BUR14] A. Burns, M. Gutiérrez, M. Aldea and M. González Harbour, “A Deadline Floor Inheritance Protocol for EDF Scheduled Embedded Real-Time Systems with Resource Sharing,” In Press, in IEEE Transaction on Computers , 2014.

[BUT05] G.C. Buttazzo, "Rate monotonic vs. EDF: judgment day," Real-Time Systems, vol 29, no. 1, págs. 5-26, 2005.

[BUT11] G.C. Buttazzo, “Hard real-time computing systems: predictable scheduling algorithms and applications,” Springer Science & Business Media, 2011.

[CAN] Robert Bosch GmbH , “CAN Specification - Version 2.0”, Stuttgart, 1991.

[CHA12] Y. Chandarli, F. Fauberteau, D. Masson, S. Midonnet and M. Qamhieh, "Yartiss: A tool to visualize, test, compare and evaluate real-time scheduling algorithms," Proc. of the 3rd International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems (WATERS), Pisa (Italy), págs. 21-26, 2012.

[CHEDD] Cheddar, página web http://beru.univ-brest.fr/~singhoff/cheddar/

[CHR06] H. Charara, J. Scharbarg, J. Ermont and C. Fraboul, "Methods for bounding end-to-end delays on an AFDX network," Proc. of the 18th Euromicro Conference on Real-Time Systems (ECRTS), Dresde (Alemania), pp. 192-202, 2006.

[COU11] P. Courbin and L. George, "FORTAS: Framework for real-time analysis and simulation," Proc. of the 2nd International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems (WATERS), Porto (Portugal), págs. 21-26, 2011.

[CUE13] C. Cuevas, L. Barros, P.L. Martínez, J.M. Drake, "Beneficios que aporta la metodología MDE a los entornos de desarrollo de sistemas de tiempo real," Revista Iberoamericana de Automática e Informática Industrial RIAI, vol 10, no 2, págs. 216-227, 2013.

[DAV11] R.I. Davis and A. Burns, “A Survey of Hard Real-Time Scheduling for Multiprocessor Systems,” ACM Computing Surveys, vol 43, no. 4, article 35, 2011.

[DER74] M.L. Dertouzos, "Control Robotics: the Procedural Control of Physical Processes," Information Processing, p. 74, 1974.

[DEU11] M. Deubzer, "Robust Scheduling of Real-Time Applications on Efficient Embedded Multicore Systems," Tesis Doctoral, Technische Universität München, 2011.

[DHA78] S.K. Dhall and C.L. Liu, "On a real-time scheduling problem," Operations research, vol. 26, no. 1, págs. 127-140, 1978.

[DIA14] U. Díaz-de-Cerio, J.P. Uribe, M. González Harbour and J.C. Palencia, "Adding precedence relations to the response-time analysis of edf distributed real-time systems," Proc. of the 22nd International Conference on Real-Time Networks and Systems, p. 129, 2014.

[DIE08] C. Diederichs, U. Margull, F. Slomka and G. Wirrer, "An application-based EDF Scheduler for OSEK/VDX," Design, Automation and Test in Europe (DATE), Munich (Germany), págs. 1045-1050, 2008.

[DIN01] M. Di Natale and A. Meschi, "Scheduling Messages with Earliest Deadline Techniques," Real-Time Systems, vol 20, Kluwer Academic Publishers, págs. 255-285, 2001.

[EAD08] EADS Innovation Works, "Case Study on Distributed Heterogeneous Communication Systems (HCS)", http://combest.imag.fr/home/?link=Application1, 2008.

[EID06] J.C. Eidson, "Measurement, control, and communication using IEEE 1588," Springer Science & Business Media, 2006.

[EMB10] P. Emberson, R. Stafford and R.I. Davis, "Techniques for the synthesis of multiprocessor tasksets," Proc. of the 1st International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems (WATERS), Brussels, págs. 6-11, 2010.

[ERIKA] ERIKA (Embedded Real-time Kernel Architecture), http://erika.sssup.it

[FAG09] D. Faggioli, F. Checconi, M. Trimarchi and C. Scordino, "An EDF scheduling class for the Linux kernel," Proc of the 11th Real-Time Linux Workshop (RTLWS), Dresden (Germany), 2009.

[FEI06] P.H. Feiler, D.P. Gluch and J.J. Hudak, "The architecture analysis & design language (AADL): An introduction," No. CMU/SEI-2006-TN-011. Carnegie-Mellon Univ Pittsburgh PA Software Engineering Inst, 2006.

[FLEXR] FlexRay Consortium, "FlexRay communications systems protocol specification version 2.1 rev. a", 2005. http://www.flexray.com

[FOLAH] Folding@home, página web https://folding.stanford.edu/home

[FRTOS] FreeRTOS, página web http://www.freertos.org/

[GON03] M. González Harbour and J.C. Palencia, "Response time analysis for tasks scheduled under EDF within fixed priorities," Proc of the 24th IEEE Real-Time Systems Symposium (RTSS), págs. 200-209, 2003.

[GUT95] J.J. Gutiérrez and M. González Harbour, "Optimized priority assignment for tasks and messages in distributed hard real-time systems," Proc of the 3rd Third Workshop on Parallel and Distributed Real-Time Systems (WPDRTS), Santa Barbara (California, USA), págs. 124-132, 1995.

[GUT95b] J.J. Gutiérrez and M. González Harbour (Director), "Planificación, análisis y optimización de sistemas distribuidos de tiempo real estricto," Tesis Doctoral, Universidad de Cantabria, 1995.

[GUT96] J.J. Gutiérrez and M. González Harbour, "Minimizing the effects of jitter in distributed hard real-time systems," Journal of systems architecture, vol 42, no. 6, págs. 431-447, 1996.

[GUT14] J.J. Gutiérrez, J.C. Palencia and M. González Harbour, "Holistic schedulability analysis for multipacket messages in AFDX networks," Real-Time Systems, vol. 50, no. 2, págs. 230-269, 2014.

[HAM04] A. Hamann, M. Jersak, K. Richter and R. Ernst, "Design space exploration and system optimization with SymTA/S-symbolic timing analysis for systems," Proc. of the 25th IEEE International Real-Time Systems Symposium (RTSS), págs. 469-478, 2004.

[HAM06] A. Hamann, M. Jersak, K. Richter and R. Ernst, "A framework for modular analysis and exploration of heterogeneous embedded systems," Real-Time Systems, vol 33, no. 1-3, págs. 101-137, 2006.

[HAN12] A. Hangan and G. Sebestyen, "RTMultiSim: A Versatile Simulator for Multiprocessor Real-Time Systems," Proc. of the 3rd International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems (WATERS), Pisa (Italy), págs. 15, 2012.

[HAR87] P.K. Harter, "Response times in level-structured systems," ACM Transactions on Computer Systems (TOCS), no. 5 vol. 3, págs. 232-248, 1987.

[HEN05] R. Henia, A. Hamann, M. Jersak, R. Racu, K. Richter and R. Ernst, "System level performance analysis–the SymTA/S approach," IEE Proceedings-Computers and Digital Techniques, vol.152, no. 2, págs. 148-166, 2005.

[HEN15] R. Henia and L. Rioux, "Formal Methods for Timing Verification The 2015 FMTV Challenge", Reto propuesto en el Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems (WATERS), https://waters2015.inria.fr/files/2014/11/FMTV-2015-Challenge.pdf, 2015.

[HOL92] J. Holland, "Genetic algorithms," Scientific American, vol. 267, no. 1, págs. 66-72, 1992.

[JAMVM] Aicas. JamaicaVM real-time java technology, página web http://www.aicas.com

[JAVRTS] E.J. Bruno and G. Bollella, "Real-Time Java Programming: With Java RTS," Pearson Education, 2009.

[JAY08] P. Jayachandran and T. Abdelzaher, "Transforming distributed acyclic systems into equivalent uniprocessors under preemptive and non-preemptive scheduling," Proc. of the Euromicro Conference on Real-Time Systems (ECRTS), págs. 233-242, 2008.

[JOS86] M. Joseph and P. Pandya. "Finding response times in a real-time system," The Computer Journal, vol 29, no. 5, págs. 390-395, 1986.

[KAO93] B. Kao and H. García Molina, "Deadline assignment in a distributed soft real-time system," Proc. of the 13th International Conference on Distributed Computing Systems, págs. 428-437, 1993.

[KAO97] B. Kao and H. Garcia-Molina, "Deadline Assignment in a Distributed Soft Real-Time System," IEEE Transactions On Parallel And Distributed Systems (TPDS), vol. 8, no. 12 1997.

[KIR83] S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi**, "**Optimization by Simulated Annealing," Science, New Series, vol. 220, no. 4598, págs. 671-680, 1983.

[KLE93] M. Klein, T. Ralya, B. Pollak, R. Obenza and M. González Harbour, "A practitioner's handbook for Real-Time Analysis," Springer, 1993.

[KOP87] H. Kopetz and W. Ochsenreiter, "Clock synchronization in distributed real-time systems," IEEE Transactions on Computers, vol. 100, no. 8, págs. 933-940, 1987.

[KOP98] H. Kopetz, "The time-triggered model of computation, " Proc. of the 19th IEEE Real-Time Systems Symposium (RTSS), Madrid (Spain), págs. 168-177, 1998.

[KOP04] H. Kopetz, A. Ademaj and A. Hanzlik, "Integration of internal and external clock synchronization by the combination of clock-state and clock-rate correction in fault-tolerant distributed systems." Proc. of the 25th Real-Time Systems Symposium (RTSS), Lisbon (Portugal), págs. 415-425, 2004.

[KOP11] H. Kopetz, "Real-time systems: design principles for distributed embedded applications," Springer Science & Business Media, 2011.

[LAM80] B.W. Lampson and D.D. Redell, "Experience with Processes and Monitors in Mesa," Communications of the ACM, vol. 23, no. 2, págs. 105-107, 1980.

[LEH89] J. Lehoczky, L. Sha and Y. Ding, "The rate monotonic scheduling algorithm: Exact characterization and average case behavior," In Real Time Systems Symposium (RTSS), págs. 166-171, 1989.

[LEH90] J. Lehoczky, "Fixed Priority Scheduling of Periodic Task Sets with Arbitrary Deadlines," In Real Time Systems Symposium (RTSS), págs. 201-209, 1990.

[LEU82] J.Y.T. Leung and J. Whitehead, "On the complexity of fixed-priority scheduling of periodic, real-time tasks," Performance evaluation, vol. 2, no. 4, págs. 237-250, 1982.

[LIJ00] J.W.S. Liu, "Real-time systems," Prentice Hall, 2000.

[LIU73] C.L. Liu and J.W. Layland, “Scheduling Algorithms for Multi-Programming in a Hard Real-Time Environment,” Journal of the Association for Computing Machinery, vol 20, no 1, págs. 46-61, 1973.

[MAK08] J. Mäki-Turja and M. Nolin, “Efficient implementation of tight response-times for tasks with offsets,” Real-Time Systems Journal, vol. 40, no. 1, págs. 77-116, 2008.

[MAR05] J.M. Martínez and M. González Harbour, “RT-EP : A Fixed-Priority Real Time Communication Protocol over Standard Ethernet, ” Proc. of the 4th International Workshop on Real-Time Networks (RTN), Palma de Mallorca (Spain), 2005.

[MARTE] Object Management Group, “UML Profile for MARTE: Modeling and Analysis of Real-Time Embedded systems,” 2009 OMG Document, v1.0 formal/2009-11-02.

[MAST] M. González Harbour, J.J. Gutiérrez, J.C. Palencia and J.M. Drake, “MAST: Modeling and Analysis Suite for Real Time Applications,” Proc. of the 13th Euromicro Conference on Real-Time Systems, Delft (The Netherlands), págs. 125-134, 2001.

[MAST2] M. González Harbour, J.J. Gutiérrez, J.M. Drake, P. López and J.C. Palencia, “Modeling distributed real-time systems with MAST 2,” Journal of Systems Architecture, vol 56, no. 6, Elsevier, págs. 331-340, 2013.

[MASTh] MAST, página web http://www.mast.unican.es

[MED01] J.L. Medina, M. Gonzalez Harbour and J.M. Drake, "Mast real-time view: A graphic UML tool for modeling object-oriented real-time systems," Proc of the 22nd Real-Time Systems Symposium (RTSS), London, págs. 245-256, 2001.

[MOK83] A.K. Mok, "Fundamental Design Problems of Distributed Systems for the Hard Real-Time Environment," Tesis Doctoral, MIT, Cambridge (Massachusetts), 1983.

[MUB12] S. Mubeen, M. Sjödin and J. Mäki-Turja, "Supporting early modeling and end-to-end timing analysis of vehicular distributed real-time applications," REACTION 2012 co-located with IEEE RTSS, San Juan (Puerto Rico), pp. 57-62, 2012.

[PAL98] J.C. Palencia and M. González Harbour, “Schedulability Analysis for Tasks with Static and Dynamic Offsets,” Proc. of the 19th Real-Time Systems Symposium (RTSS), págs. 26-37, 1998.

[PAL99] J.C. Palencia and M. González Harbour, “Exploiting Precedence Relations in the Schedulability Analysis of Distributed Real-Time Systems,” Proc. of the 20th Real-Time Systems Symposium (RTSS), págs. 328-339, 1999.

[PAL03] J.C. Palencia and M. González Harbour, “Offset-Based Response Time Analysis of Distributed Systems Scheduled under EDF,” Proc. of the 15th Euromicro Conference on Real-Time Systems (ECRTS), Porto (Portugal), págs. 3-12, 2003.

[PAP08] L. Palopoli, T. Cucinotta, L. Marzario and G. Lipari, “AQuoSA-adaptive quality of service architecture,” Software-Practice and Experience, Wiley InterScience, vol 39, pp-1-31, 2008.

[PED02] P. Pedreiras and L. Almeida, “EDF Message Scheduling on Controller Area Network,” Computing & Control Engineering Journal, vol. 13, no. 4, págs. 163-170, 2002.

[PER09] S. Perathoner, E. Wandeler, L. Thiele, A. Hamann, S. Schliecker, R. Henia, R. Racu, R. Ernst and M. González Harbour, "Influence of different abstractions on the performance analysis of distributed hard real-time systems," Design Automation for Embedded Systems vol. 13, no. 1-2, págs. 27-49, 2009.

[PER09b] S. Perathoner, “Analysis of EADS Case Study: Distributed Heterogeneous Communication System”, Presentation at COMBEST Technial Meeting, Ottobrunn (Alemania), http://www.simonperathoner.info/talks/Perathoner_2009d.pdf, 2009.

[PERC] Atego Perc, página web http://www.atego.com/products/atego-perc/

[PHI97] C.A. Phillips, C. Stein, E. Torng and J. Wein, "Optimal time-critical scheduling via resource augmentation," Proc. of the 29th annual ACM symposium on Theory of computing, págs. 140-149, 1997.

[POSIX] IEEE Standard 1003.1b, “POSIX (Portable Operating System Interface)”. IEEE Standard for Information Technology, 1993.

[POSIXb] IEEE Portable Applications Standards Committee, ISO/IEC 9945-2: 2OO3(E), “Information technology- Portable Operating System Interface (POSIX)-Part 2: System Interfaces,” IEEE Standards Association, 2003.

[PYTAB] F. Alted, I. Vilata and others, “PyTables: Hierarchical Datasets in Python”. 2002--, http://www.pytables.org.

[RAPID] RAPID RMA, Tri-Pacific Software Inc, página web http://www.tripac.com/rapid-rma

[RES] Red Española de Supercomputación, página web http://www.bsc.es/marenostrum-support-services/res

[RIC03] K. Richter, R. Racu and R. Ernst, "Scheduling analysis integration for heterogeneous multiprocessor SoC," Proc. of the 24th Real-Time Systems Symposium (RTSS), págs. 236-245, 2003.

[RIV08] J.M. Rivas and J.J. Gutiérrez (Director), “Estudio de la asignación de plazos en sistemas distribuidos de tiempo real con planificación EDF : propuesta de un algoritmo heuristic,” Proyecto Fin de Carrera, Universidad de Cantabria, 2008.

[RIV09] J.M. Rivas and J.J. Gutiérrez (Director), “Algoritmo de asignación de plazos globales en sistemas distribuidos de tiempo real con planificación EDF: comparativa de estrategias de planificación,” M.S. Thesis, University of Cantabria, October 2009.

[RIV10] J.M. Rivas, J.J. Gutiérrez, J.C. Palencia and M. González Harbour, "Optimized Deadline Assignment and Schedulability Analysis for Distributed Real-Time Systems with Local EDF Scheduling," Proc. of the 8th International Conference on Embedded Systems and Applications (ESA), Las Vegas (Nevada, USA), págs. 150-156, 2010.

[RIV11] J.M. Rivas, J.J. Gutiérrez, J.C. Palencia and M. González Harbour, "Schedulability analysis and optimization of heterogeneous EDF and FP distributed real-time systems," Proc. of the 23rd Euromicro Conference on Real-Time Systems (ECRTS), Porto (Portugal), págs. 195-204, 2011.

[RIV12] J.M. Rivas, J.J. Gutiérrez and M. González Harbour, “Fixed Priorities or EDF for Distributed Real-Time Systems?,” Proc. of the WiP Session of the 33rd IEEE Real-Time Systems Symposium (RTSS WiP), San Juan (Puerto Rico), 2012. In ACM SIGBED Review, vol. 10, no 2, págs. 21-21, 2013.

[RTLIN] M. Barabanov and V. Yodaiken, "Real-time linux," Linux journal no. 23, 1996.

[RTSJ] RTSJ (Real-Time Specification for Java), página web http://www.rtsj.org/

[SCH06] D.C. Schmidt, "Guest editor's introduction: Model-driven engineering." IEEE Computer, vol 39, no 2, págs. 25-31, 2006.

[SER10] N. Serreli, G. Lipari and E. Bini, "The Distributed Deadline Synchronization Protocol for real-time systems scheduled by EDF," IEEE Conference on Emerging Technologies and Factory Automation (ETFA), Bilbao (Spain), págs. 1-8, 2010.

[SHA86] L. Sha, J.P. Lehoczky and R. Rajkumar, "Solutions for Some Practical Problems in Prioritized Preemptive Scheduling." Proc. of the 7th IEEE Real-Time Systems Symposium, New Orleans (Lousiana) vol. 86, págs. 181-191, 1986.

[SHA90] L. Sha, R. Rajkumar and J.P. Lehoczky, "Priority inheritance protocols: An approach to real-time synchronization," IEEE Transactions on Computers, vol 39, no. 9, págs. 1175-1185, 1990.

[SHA90b] L. Sha, R. Rajkumar and J.P. Lehoczky, "Real-time scheduling support in Futurebus+," Proc. of the 11th Real-Time Systems Symposium (RTSS), págs. 331-340. 1990.

[SHA90c] L. Sha and J.B. Goodenough, “Real-Time Scheduling Theory and Ada,” IEEE Computer, vol. 23, no. 4, págs. 53-62, 1990.

[SHA04] L. Sha, T. Abdelzaher, K.E. Årzén, A. Cervin, T. Baker, A. Burns, G.C. Buttazzo, M. Caccamo, J. Lehoczky, A.K. Mok, "Real time scheduling theory: A historical perspective," Real-time systems, vol 2 págs. 101-155, 2004.

[SHARK] S.Ha.R.K (Soft Hard Real-time Kernel), página web htttp://shark.sssup.it

[SIN04] F. Singhoff, J. Legrand, L. Nana and L. Marcé, “Cheddar : a Flexible Real Time Scheduling Framework,” ACM SIGAda Ada Letters, vol. 24, no. 4, ACM Press, New York (USA), págs. 1-8, 2004.

[SON01] J.H. Son and M. Ho Kim, "Improving the performance of time-constrained workflow processing," Journal of Systems and Software, vol. 58, no. 3, págs. 211-219, 2001.

[SPI00] C.R. Spitzer, “The Avionics Handbook,”, Second Edition, CRC Press, 2000.

[SPR89] B. Sprunt, L. Sha and J.P. Lehoczky, "Aperiodic task scheduling for hard-real-time systems," Real-Time Systems, vol 1, no. 1, págs. 27-60, 1989.

[SPU96] M. Spuri, “Analysis of Deadline Scheduled Real-Time Systems,” Research Report RR-772, INRIA, France, 1996.

[SPU96b] M. Spuri and G. Buttazzo, “Scheduling aperiodic tasks in dynamic priority systems,” Real-Time Systems, vol. 10, no. 2, págs. 179-210, 1996.

[SUN96] J. Sun and J. Liu, “Synchronization protocols in distributed real-time systems, “Proc. of the 16th International Conference on Distributed Computing Systems (ICDS), págs. 38-45, 1996.

[STA88] J.A. Stankovic and K. Ramamritham, “Hard Real-Time Systems”, IEEE Computer Society, catalog no. EH0276-6, 1988.

[STR88] J.K. Strosnider, T. Marchok and J.P. Lehoczky, "Advanced real-time scheduling using the IEEE 802.5 token ring," Proceeding of the Real-Time Systems Symposium, págs. 42-52, 1988.

[STR95] J.K. Strosnider, J.P. Lehoczky and L. Sha. "The deferrable server algorithm for enhanced aperiodic responsiveness in hard real-time environments," IEEE Transactions on Computers, vol. 44, no. 1, págs. 73-91, 1995.

[SYMT] SymTA/S, página web https://www.symtavision.com/products/symtas-traceanalyzer/

[TER06] F. Terrier and S. Gérard. "MDE benefits for distributed, real time and embedded systems," From Model-Driven Design to Resource Management for Distributed Embedded Systems. Springer US, págs.15-24, 2006.

[THI00] L. Thiele, S. Chakraborty and M. Naedele, "Real-time calculus for scheduling hard real-time systems," Proc. of the International Symposium on Circuits and Systems (ISCAS), Geneva (Switzerland), vol. 4, págs. 101-104, 2000.

[THI01] L. Thiele, S. Chakraborty, M. Gries, A. Maxiaguine and J. Greutert, "Embedded software in network processors—models and algorithms," Embedded Software, Springer, Berlin Heidelberg, págs. 416-434, 2001.

[TIN92] K.W. Tindell, A. Burns and A.J. Wellings, "Allocating Real-Time Tasks. An NP-Hard Problem Made Easy, " Real-Time Systems Journal, vol. 4, no. 2, págs. 145-165, 1992.

[TIN94] K.W. Tindell and J. Clark, “Holistic Schedulability Analysis for Distributed Hard Real-Time Systems,” Microprocessing and Microprogramming, vol. 50, no. 2-3, págs. 117-134, 1994.

[TIN94b] K.W. Tindell, “Adding Time-Offsets to Schedulability Analysis,” Department of Computer Science, University of York, Technical Report YCS-221, 1994.

[TIN94c] K.W. Tindell, A. Burns and A.J. Wellings, “Calculating Controller Area Network (CAN) Message Response Times,” Control Engineer Practice, vol. 3, no. 8, Elsevier, págs. 1163-1169, 1994.

[TIN94d] K.W. Tindell, A. Burns and A.J. Wellings, "An extendible approach for analyzing fixed priority hard real-time tasks," Real-Time Systems vol. 6, no. 2, págs. 133-151, 1994.

[TORQ] Torque Resource Manager, página web http://www.adaptivecomputing.com/products/open-source/torque/

[TRESM] CPD Tres Mares, página web http://www.atc.unican.es

[TOKR] IEEE Standard 802.5, “Token ring Access Method and Physical Layer Specification,” IEEE, New york, 1989.

[TUC12] S. Tucker, R.A. Duff, L Bruckardt, E. Ploedereder, P. Leroy and E. Ploedereder, “Ada 2012 Reference Manual. Language and Standard Libraries: International Standard ISO/IEC 8652/2012 (E),” Springer, 2012.

[UPPA] K.G. Larsen, P. Pettersson and W. Yi, “Uppaal in a Nutshell,” International Journal on Software Tools for Technology Transfer (STTT), vol. 1, no. 1, págs. 134-152, 1997.

[URE06] S. Urueña, J. Zamorano, D. Berjón, J.A. Pulido and J.A. de la Puente, "The arbitrated real-time protocol (AR-TP): a ravenscar compliant communication protocol for high-integrity distributed systems," Reliable Software Technologies-Ada-Europe, Springer Berlin Heidelberg, págs. 215-226, 2006.

[URU10] R. Urunuela, A. Deplanche and Yvon Trinquet, "Storm: a simulation tool for real-time multiprocessor scheduling evaluation," IEEE Conference on Emerging Technologies and Factory Automation (ETFA), págs. 1-8, 2010.

[VXWRK] Wind River, página web, http://www.windriver.com

[WINCE] D. Boling, “Programming Windows Embedded CE 6.0 Developer Reference,” Microsoft Press, 2007.

[WU13] Z. Wu, X. Liu, Z. Ni, D. Yuan and Y. Yang, "A market-oriented hierarchical scheduling strategy in cloud workflow systems," The Journal of Supercomputing, vol. 63, no. 1, págs. 256-293, 2013.

[YU05] J. Yu and R. Buyya, "A taxonomy of workflow management systems for grid computing," Journal of Grid Computing, vol. 3, no. 3-4, págs. 171-200, 2005.