Revista Iberoamericana de Automática e Informática Industrial

Revista Iberoamericana de Automática e Informática Industrial 16 (2019) 249-263

www.revista-riai.org

Una revisión de técnicas para la optimización del despliegue y planificación de sistemas de tiempo real distribuidos

Andoni Amurrio[a,*], Ekain Azketa[a], J. Javier Gutiérrez[b], Mario Aldea[b], Jorge Parra[a]

[a] *IK4-Ikerlan Centro de Investigaciones Tecnológicas, Área de Sistemas Embebidos Confiables, Arrasate-Mondragón, España.*
[b] *Universidad de Cantabria, Grupo de Ingeniería Software y Tiempo Real, Santander, España.*

Resumen

En las últimas tres décadas, se ha realizado un gran número de propuestas sobre la optimización del despliegue y planificación de sistemas de tiempo real distribuidos bajo diferentes enfoques algorítmicos que aportan soluciones aceptables a este problema catalogado como NP-difícil. En la actualidad, la mayor parte de los sistemas utilizados en el sector industrial son sistemas de criticidad mixta en los que se puede usar la planificación cíclica, las prioridades fijas y el particionado, que proporciona aislamiento temporal y espacial a las aplicaciones. Así, en este artículo se realiza una revisión de los trabajos publicados sobre este tema y se presenta un análisis de las diferentes soluciones aportadas para sistemas de tiempo real distribuidos basados en las políticas de planificación que se están usando en la práctica. Como resultado de la comparación, se presenta una tabla a modo de guía en la que se relacionan los trabajos revisados y se caracterizan sus soluciones.

Palabras Clave:
Sistemas de tiempo real, Algoritmos de planificación, Particiones, Optimización

A review on optimization techniques for the deployment and scheduling of distributed real-time systems

Abstract

In the last three decades, a large number of proposals has been carried out for the optimization of the deployment and scheduling of distributed real-time systems under different algorithmic approaches that provide acceptable solutions for this NP-hard problem. Nowadays, most of the systems used in industry are mixed-criticallity systems which use cyclic scheduling, fixed-priority scheduling and partitioning, which provides both temporal and spatial isolation in the execution of applications. Thus, in this work a review of the works published on this topic is performed, as well as an analysis of the different proposed solutions for distributed real-time systems based on the scheduling policies that are used in practice. As a result of the comparison, a table intended as a guide is elaborated in which all the reviewed works are reported and their solutions are characterized.

Keywords:
Real-time systems, Scheduling algorithms, Partitions, Optimization

1. Introducción

En un sistema de tiempo real el funcionamiento correcto no sólo depende de que los resultados sean correctos, sino de que éstos se produzcan a tiempo, debiendo cumplir requisitos temporales que se imponen en el software. En la actualidad muchos de estos sistemas se corresponden con sistemas ciberfísicos, que son sistemas informáticos que integran las capacidades de computación, almacenamiento y comunicación junto con las capacidades de sensorización y/o control de elementos o dispositivos en el mundo físico. Desde el punto de vista de su arquitectura, estos sistemas ciberfísicos son en realidad sistemas de tiempo real distribuidos, con varios procesadores conectados por una o más redes de comunicación. Este tipo de sistemas, así como la tecnología que se desarrolla a su alrededor, tienen un

*Autor para correspondencia: aamurrio@ikerlan.es
To cite this article: A. Amurrio, E. Azketa, J.J. Gutiérrez, M. Aldea, J. Parra. 2019. A review on optimization techniques for the deployment and scheduling of distributed real-time systems. Revista Iberoamericana de Automática e Informática Industrial 16, 249-263. https://doi.org/10.4995/riai.2019.10997

importancia relevante por ejemplo en el marco conceptual de la Industria 4.0 (o cuarta revolución industrial) en la que tanto entidades de investigación como empresas están poniendo su foco de atención en la actualidad.

La evolución de los sistemas ciberfísicos ha adquirido en los últimos años nuevas dimensiones al incorporar tanto sistemas multi-núcleo como técnicas de virtualización que permiten ejecutar distintas funcionalidades del sistema en entornos protegidos espacial y temporalmente (particiones) en una misma plataforma hardware. Cada partición puede tener distintos requisitos no funcionales de garantía de tiempo de respuesta, seguridad, confidencialidad, etc., que configuran los llamados sistemas de criticidad mixta. En estos sistemas es necesaria la total independencia de las particiones con el fin de que el proceso de especificación, diseño, implementación, certificación (en los sistemas que lo requieran) y ejecución sea totalmente independiente entre distintos componentes del sistema (Directorate, 2012) (Baruah et al., 2010) (Goossens et al., 2013) (Crespo et al., 2014). En el projecto europeo MultiPARTES (Trujillo et al., 2014) por ejemplo, se propusieron una serie de herramientas para el desarrollo de sistemas de criticidad mixta basados en particionado, desde arquitecturas hardware y software hasta herramientas de planificación de particiones.

Un sistema crítico es aquel que tiene funciones cuyo fallo puede acarrear graves consecuencias humanas, materiales y/o medioambientales. Una de las principales características que debe tener un sistema crítico es la confiabilidad, que aglutina los conceptos de disponibilidad, fiabilidad, integridad, mantenibilidad y seguridad. La seguridad se cuantifica mediante el "Nivel de Integridad de Seguridad", en inglés *Safety Integrity Level* (en adelante SIL), que es el nivel relativo de reducción del riesgo que provee una función de seguridad o el nivel objetivo para la reducción de riesgo (Smith and Simpson, 2004). Los niveles de seguridad van desde SIL0, el más bajo, hasta SIL4, el más alto.

Habitualmente los sistemas complejos de automóviles, trenes y aviones combinan funciones de no-seguridad (SIL0) con otras de seguridad de diferentes niveles. En el pasado estas funciones eran desplegadas en sistemas físicos independientes para evitar interferencias de la parte de no-seguridad en la parte de seguridad. Este paradigma suele requerir típicamente una gran cantidad de equipamiento heterogéneo, dando lugar a un incremento en el coste de la instalación, de la puesta en marcha y del mantenimiento. Con el objetivo de reducir costos, la tendencia actual es desplegar las funciones de distinto SIL compartiendo recursos sobre los mismos sistemas físicos y lógicos, dando lugar a los mencionados sistemas de criticidad mixta (Vestal, 2007). En (Burns and Davis, 2017) se hace una revisión detallada del trabajo desarrollado en la última década alrededor de este tipo de sistemas, desde los aspectos más teóricos de planificación o de diseño, hasta los mecanismos básicos de implementación de los mismos.

En los sistemas críticos típicamente se emplean sistemas operativos de tiempo real para garantizar la ejecución determinista de las tareas, que está guiada por una política de planificación concreta o una combinación de ellas, entre las que se encuentran los ejecutivos cíclicos, las prioridades fijas o las basadas en plazos (Liu, 2000). Los ejecutivos cíclicos son los que más se han venido usando hasta la irrupción de las planificaciones basadas en prioridades, que han tenido gran repercusión en la inmensa mayoría de los diseños (Pop et al., 2004a), y está presente en sistemas operativos como el estándar POSIX (IEEE Portable Application Standards Committee, 2003) o AUTOSAR (AUTOSAR, 2003), en redes de comunicaciones como CAN (Bosch Gmbh, 1991) y en lenguajes de programación como el estándar Ada (ISO/IEC, 2012). En la actualidad, los sistemas particionados permiten combinar las características del ejecutivo cíclico y las prioridades fijas en un sistema de planificación jerárquico, con las particiones como nivel de planificación básico y el uso de prioridades dentro de las particiones; es el caso de VxWorks (WindRiver, 2016) usado en aviónica. En (Guevara López et al., 2014) se formaliza una clasificación de las distintas estrategias de planificación basándose en la teoría de conjuntos.

Determinar cuándo se ejecutan las tareas y se transmiten los mensajes de un sistema de tiempo real distribuido para que se cumplan todos los plazos hasta en la peor de las situaciones es lo que se conoce como planificación, y es un problema complejo para casos no triviales. Si además se pretende determinar conjuntamente en qué computadores se ejecutan las tareas y en qué redes se transmiten los mensajes, lo que se conoce como despliegue, la complejidad aumenta hasta convertirse en un problema NP-difícil (Tindell et al., 1992). En la actualidad no se conocen algoritmos que den soluciones óptimas en tiempo polinómico para este tipo de problemas, por lo que normalmente se emplean algoritmos genéricos y dedicados de búsqueda y optimización.

En este contexto, se presenta este trabajo como una contribución en dos aspectos. Por un lado, complementar el trabajo (Burns and Davis, 2017) en el que se lleva a cabo una revisión pormenorizada de los sistemas de criticidad mixta focalizándose en el análisis temporal tanto de sistemas de un solo procesador como multiprocesador, y en el que también se describen distintos modelos de sistema y aplicaciones industriales reales. El presente trabajo complementa esta revisión en cuanto a la optimización del despliegue y planificación de los sistemas de tiempo real distribuidos, aspectos que son tratados de una manera menos extensa y que son de gran interés en el desarrollo de sistemas actuales. Por otro lado, también se recogen otros trabajos de optimización del despliegue y planificación que aunque no están centrados en los sistemas de criticidad mixta, sí que abordan arquitecturas de tiempo real distribuidas que son de interés en el diseño de aplicaciones industriales actuales.

Así, en este artículo se recopilan los trabajos que abordan este problema, desde que se hiciera por primera vez en (Tindell et al., 1992) hasta la primera mitad del año 2018, ordenados en su mayoría cronológicamente y clasificados según el enfoque algorítmico que se emplea para resolverlo. Todos los trabajos se recogen en una tabla en la que se muestran sus aspectos más destacables, con el objetivo de elaborar un resumen claro del estado del arte actual y facilitar su consulta para futuras labores en la optimización del despliegue y planificación de sistemas de tiempo real distribuidos.

El presente documento está organizado de la siguiente manera. En el apartado 2 se realiza una descripción detallada del modelo de sistema que se ha considerado y la metodología para la recopilación de trabajos. En el apartado 3 se realiza la revisión de la literatura, y en las conclusiones del apartado 4 se

muestra una tabla que resume todos los trabajos recogidos en este artículo así como las futuras líneas que pueden surgir de este estudio.

2. Modelo de sistema y metodología

En este apartado se describe el modelo de sistema distribuido de tiempo real que se ha considerado para la recopilación de trabajos de esta revisión. También se dan unas nociones básicas acerca de las técnicas de optimización en función de las cuales se han clasificado los trabajos recogidos en el apartado 3.

Para la elaboración de esta revisión se va a poner el foco en los sistemas de tiempo real distribuidos. El modelo de sistema habitual que describe este tipo de sistemas puede definirse mediante las arquitecturas físicas y lógicas del sistema. Este modelo sirve para describir la gran mayoría de los trabajos en este campo.

Por un lado, la arquitectura física de los sistemas distribuidos la componen computadores y redes de comunicaciones. Los computadores pueden ser homogéneos o heterogéneos en cuanto a su arquitectura de núcleo y velocidad de computación y su interconexión se lleva a cabo mediante redes de comunicaciones. Estas redes pueden seguir distintos esquemas de planificación, como las redes basadas en los estándares *Time Sensitive Networks* (TSN) desarrollados por distintos grupos de trabajo de *IEEE 802.1*, así como esquemas particionados (Airlines Electronic Engineering Committee, 2009).

La arquitectura lógica de las aplicaciones está compuesta por tareas y mensajes. Ambos están normalmente caracterizados por los plazos (D), que son requisitos temporales que deben cumplir, y por sus tiempos de ejecución/transmisión de peor caso. Su activación puede ser periódica (según el periodo T) o no. La tarea constituye la unidad mínima planificable en un computador, que puede ejecutarse únicamente si se dispone de los recursos hardware y software requeridos. Su activación puede sufrir cierta variabilidad, lo que se conoce como *jitter*. Los mensajes, cuya unidad mínima planificable son los paquetes, posibilitan la comunicación entre tareas con relaciones de precedencia formando los denominados flujos de extremo a extremo, en adelante flujos e2e. Todas las tareas y mensajes se activan al producirse ciertos estímulos. Según su naturaleza, se reconocen dos paradigmas que clasifican los sistemas de tiempo real (Kopetz, 2011): los que se activan por tiempo o *Time-Triggered* (TT) y los que lo hacen por eventos o *Event-Triggered* (ET). En este artículo se contemplan ambos paradigmas.

La planificación de las tareas y mensajes se puede llevar a cabo mediante la asignación de prioridades, la generación de ejecutivos cíclicos o la combinación de ambos enfoques, denominada planificación jerárquica. Con respecto a la asignación de prioridades se considerará para la recopilación de trabajos una asignación de prioridades fijas siguiendo distintos criterios. La asignación tradicional en función del plazo impuesto se denomina *Deadline Monotonic Scheduling* (DMS), mientras que si se hace en función del periodo de las tareas y mensajes se denomina *Rate Monotonic Scheduling* (RMS). También se consideran las redes de comunicaciones basadas en prioridades, por ejemplo CAN. Se recogerán también aquellas contribuciones que contemplen las planificaciones cíclicas, en las que se determina un macro-periodo o *Major Frame* (MF) periódico, dentro del cual se lleva a cabo la ejecución de las tareas. Algunos de los trabajos se basan en el esquema *Round-Robin* (RR), en el que el tiempo de ejecución se divide en porciones, equitativas o no, en las cuales se ejecutan las tareas asignadas a dicha porción.

Dada la tendencia actual hacia el diseño de sistemas que implementan funcionalidades de distintos niveles de criticidad sobre las mismas plataformas hardware, también se van a recopilar los trabajos en los que se emplea el particionado para garantizar el aislamiento tanto temporal como espacial de su ejecución. Se van a recoger, siempre y cuando sus modelos contemplen una arquitectura de tiempo real distribuida, aquellas contribuciones que sigan el esquema de particionado basado en *Integrated Modular Avionics* (IMA) (John, 1999). Las particiones espaciales son regiones de memoria reservadas e inviolables que contienen datos y código de las tareas y mensajes. Por otro lado, las particiones temporales son ventanas temporales o *slots* en los cuales se ejecutan sólamente las tareas que están asociadas a ellas. Se contempla el uso de redes de comunicaciones cuyo acceso se regula igualmente mediante particiones temporales, siguiendo un esquema de acceso a la red por división de tiempo o *Time Division Multiple Access* (TDMA), como por ejemplo las redes de protocolo *Time-Triggered Protocol* (TTP).

En resumen, el modelo de sistema lo componen flujos e2e compuestos por tareas y mensajes con relaciones de precedencia, que pueden ser distribuidos y son desplegados en computadores y redes de comunicaciones. Los computadores pueden seguir distintas políticas de planificación, como la cíclica, *Round-Robin*, basada en la asignación de prioridades o jerárquica, en la que la ejecución se lleva a cabo según una asignación de prioridades dentro de un ciclo periódico. Esta última se emplea habitualmente en los computadores basados en particiones, en los que se debe garantizar el aislamiento en la ejecución de las aplicaciones alojadas en ellos. Durante la elaboración de este artículo, se ha constatado que hay una gran cantidad de trabajos destacados que contemplan los sistemas de criticidad mixta. Sin embargo, en aras de acotar el ya extenso área que forman las contribuciones que se ajustan al modelo de sistema descrito, solo se recopilarán aquellos que aborden de forma explícita sistemas con requisitos de tiempo real, en los que se busca satisfacer plazos estrictos, e implementados sobre arquitecturas distribuidas, en las cuales tan importante es la planificación a nivel de computador como de red.

Una de las maneras para determinar que una planificación y despliegue cumplen todos los requisitos temporales es mediante el cálculo de los tiempos de respuesta de peor caso de las tareas y los mensajes. Los métodos matemáticos que permiten obtener esos tiempos se denominan técnicas de análisis de planificabilidad. Se dice que un sistema es planificable si los tiempos de respuesta de peor caso de las tareas y mensajes no superan sus correspondientes plazos. Desde el trabajo seminal de Liu y Layland en 1973 (Liu and Layland, 1973) se ha desarrollado una gran cantidad de técnicas que permiten garantizar la planificabilidad de los sistemas de tiempo real, tal y como puede verse en los trabajos recopilatorios (Sha et al., 2004) y (Burns and Davis, 2017). Más recientemente se han desarrollado técnicas de análisis temporal para sistemas de tiempo real distribuidos que permiten el uso de distintas políticas de planificación en sistemas heterogéneos (Rivas et al., 2011), y también en sistemas particionados distribuidos, tanto en los procesado-

res (Palencia et al., 2017) como en las redes de comunicaciones (Gutiérrez et al., 2014).

Tal y como se ha mencionado en el apartado anteior, debido a la naturaleza NP-difícil del problema que supone el despliegue y planificación de un sistema de tiempo real distribuido, se emplean algoritmos de optimización. Éstos proporcionan soluciones válidas que pueden ser óptimas o sub-óptimas. En este artículo se recogen trabajos que emplean distintos enfoques algorítmicos, de los que se da una breve descripción a continuación:

- Los algoritmos genéticos, en inglés *Genetic Algorithm* (GA) (Holland, 1975), son metaheurísticas probabilísticas genéricas que forman parte de los llamados algoritmos evolutivos. Estos algoritmos imitan mecanismos biológicos (adaptación al medio, mutación, cruce, herencia...) que guían el proceso de evolución de las especies, y se emplean para buscar soluciones a diversos problemas en amplios espacios de búsqueda.

- La búsqueda tabú, en inglés *Tabu Search* (TS) (Glover, 1986), es un procedimiento metaheurístico de búsqueda y optimización de propósito general, que forma parte del grupo de técnicas de búsqueda local. Su principal característica es la posibilidad de evitar soluciones sub-óptimas locales dentro del espacio de soluciones para un problema concreto.

- El templado simulado, en inglés *Simulated Annealing* (SA) es una metaheurística probabilística genérica para aproximar la optimización global en una extensa área de búsqueda. Como puede verse en (Kirkpatrick, 1984), viene inspirado por el proceso de templado de la metalurgia.

- La programación matemática (Minoux, 1986) es una familia de técnicas de optimización usada para maximizar o minimizar una función, eligiendo sistemáticamente valores de entrada de entre unos valores permitidos y calculando el valor de la función. Esta función puede estar sujeta a ciertas restricciones expresadas en términos de ecuaciones y/o inecuaciones. Dependiendo del tipo de funciones y restricciones, así como de los valores asignados a ellas, existen distintas variantes de programación matemática.

  El objetivo de la programación lineal,en inglés *Linear Programming* (LP) (Schrijver, 1998) y que forma parte de la programación matemática, es maximizar o minimizar una función lineal sujeta a restricciones, que está formulada mediante ecuaciones e inecuaciones lineales simples. Si todas las variables son enteros, se denomina programación lineal entera o *Integer Linear Programming* (ILP), mientras que si sólo algunas lo son, se le denomina programación lineal entera mixta o *Mixed Integer Linear Programming* (MILP). Si no se tratan ecuaciones e inecuaciones lineales se denomina simplemente *Mixed Integer Programming* (MIP). La programación geométrica o *Geometric Programming* (GP) (Boyd et al., 2007) es el tipo de programación matemática usada para maximizar o minimizar funciones posinómicas con restricciones. Éstas se formulan mediante ecuaciones monómicas iguales a 1 e inecuaciones posinómicas menores o iguales a 1. Por último, la programación de satisfacción de restricciones o *Constraint Satisfaction Programming* (CSP) es un tipo de programación matemática basada en determinar el valor de las variables que garanticen cumplir con una función objetivo (Tsang, 2014). Si las variables de la función objetivo y las funciones de restricciones son booleanas, se denomina *Problema de Satisfacción Booleana* o simplemente SAT. Por último, también se consideran los trabajos que emplean técnicas basadas en teorías de satisfacción de módulo, en inglés *Satisfiability Modulo Theories* (SMT), que consisten en fórmulas en lógica de primer orden en las que alguna función tiene interpretaciones adicionales, y debe determinarse si una fórmula se puede satisfacer (Barrett and Tinelli, 2018).

- Ramificar y acotar, en inglés *Branch and Bound* (BB) (Land and Doig, 1960), es un algoritmo genérico empleado en problemas de optimización combinatoria y discreta y optimización matemática. Las soluciones candidatas se enumeran formando un árbol y se realizan descartes basándose en estimaciones superiores e inferiores de los factores que deben ser optimizados.

- Finalmente, un heurístico (HEU) (Pearl, 1984) es aquel método de resolución de problemas basado en reglas construidas según criterios ligados a los mismos, generalmente basándose en la experiencia previa. A pesar de que obtener una solución óptima no está garantizado, pueden lograrse soluciones razonablemente buenas en un tiempo aceptable si están bien construidos.

La revisión de los trabajos se va realizar de acuerdo a la siguiente metodología. Por un lado, se va a prestar atención a los problemas que se abordan en cada uno, desde el punto de vista del despliegue, el tipo de planificación y el particionado. Estas características son fundamentales en los sistemas de tiempo real que implementan funcionalidades de distinto nivel de criticidad. La revisión también se va a centrar en los objetivos que persigue cada trabajo. Todos tienen por objetivo la planificabilidad del sistema, y también se han considerado aquellos que además buscan la minimización de los siguientes parámetros: el número de computadores, la utilización de recursos y los tiempos de respuesta. Por último se constatarán, en aquellos en los que sea posible afirmarlo, las restricciones que presentan los trabajos en cuanto al uso de memoria y a la relación entre los periodos y los plazos en el modelo de sistema que se describe. Todas estas características se recogerán en una tabla a modo de resumen, que constituye en realidad una guía en la que localizar de manera sencilla las soluciones disponibles para un problema concreto de interés.

3. Revisión de la literatura

En este apartado se recogen los trabajos realizados en el ámbito del despliegue y planificación de sistemas de tiempo real distribuidos. Estos trabajos están clasificados según el enfoque algorítmico empleado, y se realiza una breve reseña de cada uno de ellos. Se menciona, al final de cada subapartado, algún aspecto relevante de dicho algoritmo a modo de conclusión parcial que complemente a la tabla del final del documento.

3.1. Algoritmo Genético (GA)

En (Mitra and Ramanathan, 1993), (Hou et al., 1994) y (Monnier et al., 1998) se emplean distintas variaciones de GA para desplegar tareas en elementos procesadores. Todos ellos se usan para crear planificaciones cíclicas con el objetivo de minimizar el tiempo de respuesta de las mismas. En (Mitra and Ramanathan, 1993) y (Monnier et al., 1998) consideran flujos e2e definidos mediante sus plazos y periodos, y en (Hou et al., 1994) se aborda un sistema multiprocesador.

En (Dick and Jha, 1998) se desarrolla un algoritmo multiobjetivo para decidir el número de recursos hardware en un *System On Chip* (SoC), así como para desplegar las tareas en los nodos procesadores y construir ejecutivos cíclicos. El uso combinado de GA y TS pretende minimizar el consumo energético y el precio del sistema resultante, mientras se cumplen todas las restricciones temporales.

El trabajo (Faucou et al., 2000) también aborda el despliegue de tareas periódicas y la creación de planificaciones cíclicas. Sin embargo, el modelo de sistema contempla los elementos procesadores idénticos, conectados mediante redes TDMA. Para ello se lleva a cabo una asignación explícita de turnos.

En los trabajos (Oh and Wu, 2004) y (Yoo, 2009) se emplean GAs para desplegar tareas en procesadores idénticos y determinar planificaciones cíclicas en sistemas de tiempo real distribuidos. Ambos consideran flujos e2e y plazos predefinidos, y su principal objetivo es minimizar la suma de todas las diferencias entre el tiempo de respuesta de peor caso y el plazo de todos los flujos. El número de elementos procesadores también pretende ser minimizado. Estos trabajos se diferencian en los mecanismos de convergencia del GA hacia una solución adecuada.

En (Hamann et al., 2006) se propone un GA multiobjetivo para asignar prioridades a tareas, así como para determinar los *slots* temporales para los mensajes transmitidos a través de una red TDMA. Se emplea una técnica de análisis temporal específica, que permite ajustar las variables y los objetivos a minimizar.

En (Shang et al., 2007) se contempla un modelo de sistema basado en computadores con recursos de memoria y flujos e2e cuyo periodo y plazo son predefinidos. Su principal objetivo es minimizar el precio y el consumo energético del sistema resultante, en el que son determinados el número de procesadores, elementos de comunicación y matrices programables de puebas (en inglés FPGAs) que lo compondrán. La planificación se lleva a cabo mediante un algoritmo heurístico que emplea criterios de optimización de Pareto para evaluar las soluciones candidatas (Goldberg and Holland, 1988), (Fonseca and Fleming, 1998).

En (Samii et al., 2009) se utilizan algoritmos genéticos para asignar prioridades fijas a tareas y prioridades fijas e identificadores de trama a los mensajes en un sistema de tiempo real distribuido conectado mediante una red FlexRay. El objetivo principal es optimizar el tiempo de respuesta medio del sistema.

El trabajo (Azketa et al., 2011b) desarrolla una codificación permutacional para un GA cuyo objetivo es la asignación de prioridades a tareas y mensajes que componen un sistema distribuido de tiempo real. Los resultados obtenidos en este trabajo muestran mejorar los obtenidos en (Gutiérrez and González Harbour, 1995) mediante un heurístico llamado HOPA. Después, en (Azketa et al., 2012), emplea esta misma codificación para planificar las tareas del mencionado sistema, usando HOPA esta vez para generar la primera población de soluciones que el algoritmo optimizará. Por último, en (Azketa et al., 2011a) se emplea el algoritmo genético para desplegar y planificar tareas y mensajes en sistemas distribuidos de tiempo real, minimizando parámetros tales como la utilización de los computadores, recursos de memoria y comunicaciones, tiempos de respuesta y el número de computadores utilizados, mientras se cumplen todos los requisitos de tiempo real.

En (Wozniak et al., 2013) se utiliza un GA para la síntesis de sistemas de tiempo real distribuidos, compuestos por elementos procesadores heterogéneos. El proceso de síntesis consta de las siguientes fases: desplegar los componentes software (compuestos por conjuntos de ejecutables) en elementos procesadores y de señales en los buses de comunicaciones, descomponer los ejecutables en tareas y señales en mensajes y asignarles prioridades fijas. Además de cumplir con todos los plazos, tiene por objetivo la optimización de ciertos parámetros: el tiempo de respuesta de los flujos e2e, el uso de memoria, el rendimiento de los buses de comunicaciones y el tiempo de respuesta de los ejecutables.

El trabajo (Boutekkouk and Oubadi, 2014) propone un GA para minimizar el tiempo de respuesta de un sistema de tiempo real con tareas periódicas y aperiódicas, manteniendo un uso de los elementos procesadores equilibrado. Considera flujos con pseudoperiodos y se contempla el uso de prioridades fijas y dinámicas. Emplea una técnica correctora en las soluciones para guiar la inicialización del algoritmo y mantener las relaciones de precedencia de las tareas. En (Boutekkouk and Oubadi, 2016) se añade al algoritmo genético una fase basada en codificación de lógica cuántica.

En (Ayari et al., 2016a) se emplea un algoritmo genético para el despliegue de tareas independientes en sistemas distribuidos de tiempo real heterogéneos, a las que se les asignan prioridades fijas siguiendo el esquema RMS. El tiempo total de ejecución, el coste de las comunicaciones entre tareas y el consumo de memoria se expresan en funciones de adecuación que han de ser minimizadas. En el siguiente trabajo (Ayari et al., 2018), se presenta un algoritmo genético avanzado (ImGA) que implementa un operador de cruce basado en la planificabilidad de las tareas (Ayari et al., 2016b), lo cual mejora la precisión de los algoritmos genéticos empleados anteriormente.

De los trabajos que emplean algoritmos genéticos recogidos puede destacarse que este enfoque es utilizado habitualmente como optimizador multiobjetivo, ya que busca soluciones en las que más de un parámetro ha de ser minimizado (costes, número de computadores...).

3.2. Búsqueda Tabú (TS)

En (Porto et al., 2000) se presenta el uso de TS para desplegar tareas con relaciones de precedencia en procesadores y crear planificaciones cíclicas para sistemas multiprocesadores heterogéneos y de tiempo real, compuestos por computadores idénticos salvo uno de mayor capacidad de procesamiento. El objetivo principal es minimizar el tiempo de respuesta del sistema.

En (Chen and Lin, 2000) se desarrolla un algoritmo híbrido basado en TS para resolver el problema del despliegue de tareas con relaciones de precedencia en una arquitectura multiprocesador. El objetivo es cumplir con los requisitos de tiempo real del sistema satisfaciendo las restricciones de capacidad de computación máxima que tienen los computadores, así como minimizar el tráfico de red y el número de computadores utilizados.

El trabajo (Lin et al., 2000) propone los valores de los operadores y parámetros de configuración para los algoritmos TS y GA, con el objetivo de desplegar tareas de tiempo real y crear planificaciones cíclicas en arquitecturas de multiprocesadores, cuyos procesadores considera idénticos. El objetivo es minimizar el tiempo de respuesta global del sistema.

En (Tămaş-Selicean and Pop, 2011b) se propone un método de optimización para sistemas de tiempo real distribuidos particionados. Mediante un algoritmo basado en TS pretende desplegar tareas en elementos procesadores y definir las ventanas temporales de ejecución de las mismas dentro del MF. El algoritmo también se encarga de asignar las tareas a estas particiones temporales y de generar un ejecutivo cíclico. El objetivo es cumplir con los plazos de todas las tareas a la vez que se maximiza el tiempo de desuso de las particiones temporales. Posteriormente el trabajo es extendido en (Tămaş-Selicean and Pop, 2015), donde se proponen una serie de técnicas para posibilitar la planificación de las aplicaciones, para lo que tiene en cuenta los costes de certificación para albergar aplicaciones de criticidad mixta, que son minimizados empleando distintas técnicas.

En (Jiang et al., 2017) se emplea un algoritmo multiobjetivo basado en TS para planificar tareas y mensajes en sistemas de tiempo real distribuidos embebidos. El objetivo es desarrollar una implementación que cumpla con los requisitos de tiempo real del sistema, ademas de los de seguridad y confiabilidad, mientras se minimiza el consumo energético. Se asignan prioridades a las tareas en orden descendente, y se asume que han sido desplegadas en los elementos procesadores previamente.

A pesar de no haber un elevado número de trabajos en los que se emplee TS, puede verse en la tabla 1 del apartado 4 que se contemplan problemas abordados y optimizaciones variadas, incluyendo sistemas basados en particiones.

3.3. Templado Simulado (SA)

En (Tindell et al., 1992) se emplea SA para desplegar tareas y asignarles prioridades fijas en sistemas de tiempo real distribuidos, en los que se emplea una red de comunicaciones TDMA. En este modelo de sistema se consideran los recursos de memoria de los computadores y las tareas tienen una serie de computadores candidatos en los que pueden alojarse. Además, algunas tareas son réplicas de otras y no pueden desplegarse en el mismo elemento procesador. El plazo de las tareas es igual a su periodo, y las prioridades se asignan siguiendo el criterio RMS. El objetivo del trabajo es minimizar la carga de la red mientras todas las tareas cumplen sus plazos. Otra restricción a considerar es el uso de memoria de los computadores, que no debe rebasarse.

El trabajo (Burns et al., 1993) propone un algoritmo basado en SA para desplegar tareas en elementos procesadores y asignarles prioridades fijas. Se considera que cada tarea tiene una serie de computadores candidatos en los que puede ser desplegada. También contempla flujos e2e con periodos y plazos. El objetivo es minimizar el tiempo de respuesta de peor caso de todos los flujos así como el número de tareas auxiliares que enrutan las comunicaciones.

En (Coli and Palazzari, 1995) se utiliza SA para generar planificaciones cíclicas en sistemas de tiempo real distribuidos. El objetivo es minimizar el *jitter* de las tareas mientras los plazos de los flujos e2e se cumplen.

En (Di Natale and Stankovic, 1995) se emplea SA para crear planificaciones cíclicas en sistemas distribuidos de tiempo real basados en flujos compuestos por tareas periódicas. El objetivo es minimizar el *jitter* de las tareas mientras los plazos de las tareas y los flujos se cumplen.

El trabajo (Vargas and de Oliveira, 2005) presenta un estudio comparativo de algoritmos *Multi-start* (Martí et al., 2016), SA y TS empleados para la síntesis de sistemas de tiempo real distribuidos, conectados a través de una red de comunicaciones *Foundation Fieldbus*. Los resultados muestran que SA es el algoritmo más difícil de configurar, así como que TS es el que más y mejores soluciones obtiene.

En (He et al., 2010) se utiliza SA para desplegar tareas en computadores homogéneos, además de asignar prioridades y periodos a las tareas y configurar el acceso a la red de comunicaciones basada en TTP. El plazo de las tareas es igual a su periodo, y el objetivo es minimizar el tiempo de ejecución de peor caso de todos los flujos e2e. La propuesta se basa en dos fases. La primera emplea SA para el despliegue y la asignación de las tareas. Después, sobre esta solución, se emplea GP para fijar los plazos de las tareas y gestionar el acceso a la red de comunicaciones.

En (Emberson and Bate, 2010) se propone el uso de SA para el despliegue de tareas y mensajes y la asignación de prioridades con el objetivo de facilitar la extensión o actualización de escenarios minimizando el impacto de estos cambios en cuanto a los requisitos del sistema original.

En el trabajo (Tămaş-Selicean and Pop, 2011a) se propone utilizar SA para resolver un problema de optimización de sistemas de tiempo real distribuidos basados en particiones. El particionado temporal se implementa mediante porciones temporales en las que se divide la ejecución de tareas dentro de un elemento procesador, y es el algoritmo quien decide el orden y longitud de estas porciones dentro del procesador. Se considera que las tareas son desplegadas en los elementos procesadores previamente. La planificación es cíclica para las tareas críticas, mientras que las no críticas se planifican mediante prioridades fijas con expulsión. El objetivo es optimizar la secuencia y longitud de las ventanas temporales y obtener una planificación que permita a todas las aplicaciones de seguridad crítica cumplir con sus plazos.

En (Pishdar and Akkasi, 2015) también se lleva a cabo un estudio comparativo, esta vez entre SA y GA. Se evalúa la planificación de tareas en arquitecturas multiprocesador homogéneas, con el objetivo de minimizar el tiempo de respuesta del sistema respetando las relaciones de precedencia de las tareas. El estudio demuestra que tras 9 casos de estudio, SA obtiene 6 soluciones aceptables mientras que GA obtiene 5.

La mayoría de los trabajos que han hecho uso de SA han abordado el problema del despliegue además del de la planifi-

cación. También se aborda por primera vez la minimización del *jitter* de las tareas.

3.4. Programacion Matemática

En (Szymanek and Kuchcinski, 2001) se emplea CSP para desplegar tareas en elementos procesadores y crear planificaciones cíclicas no expulsoras en arquitecturas distribuidas de tiempo real. Estas arquitecturas están compuestas por redes heterogéneas que conectan computadores con recursos de memoria y ASICs (circuitos integrados de aplicación específica). Las tareas son seleccionadas secuencialmente y desplegadas en el procesador con menor coste. Tras el despliegue, se emplea la técnica BB para asignar un tiempo de ejecución a cada tarea. El objetivo es minimizar el tiempo de toda la planificación sin sobreutilizar los recursos de memoria. En (Szymanek and Krzysztof, 2003) se presenta un método para facilitar el problema del despliegue y la planificación mencionado anteriormente.

En el trabajo (Ekelin and Jonsson, 2001) se utiliza CSP para desplegar tareas en computadores y crear planificaciones cíclicas en sistemas de tiempo real distribuidos. Los computadores son heterogéneos y están conectados mediante un bus de comunicaciones. La contribución principal de este trabajo es la evaluación de varios métodos heurísticos de búsqueda local que deciden qué variables y con qué valor se asignan en cada iteración. Se pretende cumplir con 3 objetivos: minimizar el tiempo total de la planificación, minimizar las comunicaciones a través del bus y mantener un uso equilibrado de los elementos procesadores.

En (Metzner and Herde, 2006) se emplea SAT para desplegar y planificar tareas en sistemas de tiempo real distribuidos compuestos por elementos procesadores heterogéneos. Para ello, se asignan prioridades a las tareas de cada computador siguiendo el esquema DMS. Cada tarea tiene periodo, plazo, requisitos de memoria, un subconjunto de computadores candidatos en los que desplegarse y algunas no pueden ser desplegadas en el mismo computador que otro subconjunto determinado.

En (Davare et al., 2007) se emplea GP para optimizar los periodos de tareas y mensajes. En este caso, ya se encuentran desplegadas y las prioridades también han sido asignadas previamente, por lo que el objetivo es minimizar la suma de los tiempos de ejecución de peor caso.

En el trabajo (Zheng et al., 2007b) se usa MILP para desplegar tareas en computadores, empaquetar señales en mensajes y asignar prioridades fijas a las tareas y mensajes en sistemas distribuidos de tiempo real basadas en redes CAN. Este modelo considera flujos e2e compuestos por tareas que intercambian mensajes y que tienen plazos y periodos. El objetivo es minimizar el tiempo de ejecución de peor caso de los flujos mientras se cumplen sus plazos. Dado que a escala industrial las técnicas basadas en MILP son computacionalmente muy costosas, el problema se divide en dos subproblemas que pueden ser resueltos por separado.

Las aplicaciones estrictamente periodícas pueden tener tiempos de respuesta mucho mayores que las activadas por eventos. Por ello, cuando un plazo de una tarea o mensaje no se cumple, es posible aplicar el paradigma ET para que su tiempo de respuesta disminuya y el plazo pueda cumplirse. En (Zheng et al., 2007a) se emplea MILP para ajustar el modelo de activación con el objetivo de cumplir todas las restricciones temporales.

En (Hladik et al., 2008) se emplea CSP para desplegar y planificar tareas en sistemas de tiempo real distribuidos compuestos por computadores homogéneos conectados a través de una red CAN. Se asume que las tareas y mensajes tienen plazos iguales a sus periodos, y las prioridades se asignan en una fase anterior a este proceso. Todas las tareas tienen un subconjunto de computadores candidatos en los que pueden ser desplegadas, y pueden requerir serlo en el mismo o distinto computador que otras tareas. El objetivo es obtener un despliegue que permita cumplir todos los plazos. Para ello, se proponen dos métodos. El primero esta basado en *backtracking* y aborda simultáneamente los problemas de despliegue y planificación, y el segundo método divide estos dos subproblemas de acuerdo a un esquema *Benders* (Schrijver, 1998) para abordarlos por separado.

En (Zhu et al., 2009) se emplea MILP para desplegar tareas en procesadores y señales en mensajes, así como para asignar prioridades fijas a tareas y mensajes, en sistemas de tiempo real distribuidos que consideran un modelo basado en flujos e2e con plazos fijos. El objetivo es maximizar la extensibilidad: cuánto puede prolongarse el tiempo de ejecución de peor caso sin que afecte al cumplimiento de las restricciones temporales de las tareas mientras se cumplen todos los plazos. El método propuesto asume una asignación de prioridades inicial para los mensajes y las tareas.

En (Al Sheikh et al., 2010) se presenta un algoritmo basado en MILP empleado para desplegar y planificar particiones temporales estrictamente periódicas en arquitecturas distribuidas basadas en IMA. Una serie de restricciones son definidas matemáticamente (evitar solapamientos temporales, acceso compartido a recursos, costos de comunicación...) y resueltas mediante una herramienta comercial, demostrando obtener resultados casi óptimos.

En (Zhu et al., 2012) se utiliza MILP para desplegar tareas sobre arquitecturas distribuidas con elementos procesadores heterogéneos interconectados mediante redes CAN, que a su vez están segmentadas mediante *Gateways*. También se asignan prioridades a las tareas y mensajes, con el objetivo de cumplir los requisitos de tiempo real estricto y minimizar la latencia de los flujos e2e.

En el trabajo (Craciunas and Oliver, 2014) se plantea utilizar CSP para desplegar y planificar tareas y mensajes en sistemas de tiempo real distribuidos. Después de construir las restricciones lógicas que definen el sistema, emplea dos algoritmos basados en SMT. El primero, denominado *One-shot*, aborda la planificación de todos los tipos de tareas definidas en el modelo de sistema. El segundo método no considera la planificación de las tareas independientes en una primera aproximación, de manera que no añaden dificultad a la formulación del algoritmo SMT. Tras llevar a cabo la planificación basándose en el algoritmo propuesto en (Craciunas et al., 2014), se añaden las tareas independientes. En este trabajo se considera la planificación de los elementos procesadores y de la red de comunicaciones, y en su siguiente trabajo (Craciunas et al., 2016) se extiende el análisis de planificabilidad aplicado a redes TSN.

En (Zhang et al., 2014) se utiliza MIP para planificar sistemas de tiempo real distribuidos, tanto a nivel de computador como de red. Se considera que las tareas se encuentran desple-

gadas en los computadores con un solo procesador. El objetivo, representado mediante la función objetivo, es minimizar los tiempos de respuesta de las aplicaciones que se ejecutan mientras se cumple con todos los plazos de las tareas. Las restricciones temporales se representan mediante ecuaciones e inecuaciones, para las que puede establecerse el máximo tiempo de respuesta tolerable.

En los trabajos (Althaus et al., 2012) y (Althaus et al., 2014) se presenta un algoritmo basado en formulaciones ILP que se utiliza para el despliegue y planificación de tareas periódicas en sistemas distribuidos de tiempo real. La arquitectura se basa en múltiples subsistemas que se comunican mediante un bus global. Las tareas se planifican siguiendo el esquema DMS y forman flujos e2e junto con los mensajes que emplean para comunicarse.

En (Minaeva et al., 2018) se proponen tres aproximaciones para planificar tareas y mensajes con *jitter* y relaciones de precedencia en sistemas de tiempo real. Se considera que el despliegue de las tareas en los computadores se ha realizado previamente. Primero se formula una aproximación basada en SMT, en la que el espacio de soluciones se define mediante cinco conjuntos de restricciones. Después se describe un modelo ILP de manera similar, con la diferencia de las restricciones lineales requeridas por este método. A pesar de que las soluciones obtenidas de estas dos aproximaciones son óptimas, se presenta un tercer heurístico basado en tres niveles, que si bien no resulta óptimo, demuestra obtener soluciones aceptables en un tiempo razonable y para una escala mucho mayor del mismo problema.

En (Blikstad et al., 2018) se desarrolla un algoritmo basado en MILP para planificar tareas periódicas en sistemas de tiempo real distribuidos basados en la arquitectura IMA. Mediante la formulación matemática de las restricciones y requisitos de tiempo real del sistema, se construye un ejecutivo cíclico cuyo periodo es el mínimo común múltiplo de los periodos de las tareas. Se considera que las tareas ya han sido desplegadas previamente. Emplea el uso de particiones de la siguiente manera: los nodos que forman esta arquitectura implementan distintos módulos para proporcionar el particionado espacial, y mediante el ejecutivo cíclico resultante de la planificación se obtiene el aislamiento temporal. Dentro de cada partición temporal las tareas se planifican según el esquema RMS.

Este método es uno de los más utilizados. Por lo general no se abordan varios problemas a la vez, quizás debido a la complejidad de su formulación matemática.

3.5. *Ramificar y Acotar (BB)*

En (Hou and Shin, 1997) se emplea BB para desplegar tareas en computadores y generar planificaciones cíclicas en sistemas de tiempo real distribuidos compuestos por computadores idénticos. El objetivo es maximizar la probabilidad de que todas las tareas cumplan sus plazos.

En (Peng et al., 1997) se utiliza BB para desplegar tareas en computadores y crear planificaciones cíclicas en sistemas de tiempo real distribuidos compuestos por computadores heterogéneos. Se contempla un modelo de flujos e2e con periodo y plazo, compuestos por tareas con relaciones de precedencia y que pueden comunicarse con tareas de otros flujos. Además, las tareas pueden tener diferentes tiempos de ejecución de peor caso en distintos computadores. El algoritmo propuesto despliega todas las tareas de un flujo en un computador y establece el orden de ejecución de las mismas, y tiene como objetivo la minimización de la ratio entre el tiempo de respuesta de peor caso y el plazo de los flujos.

En (Richard et al., 2003) se despliegan tareas en elementos procesadores y se les asignan prioridades fijas empleando BB. La arquitectura está compuesta por distintos conjuntos de computadores conectados mediante una red de comunicaciones CAN. Dentro de cada uno de estos conjuntos todos los computadores son idénticos, aunque entre distintos conjuntos pueden ser distintos. Las tareas tienen plazos determinados y pueden comunicarse entre ellas. La planificación se lleva a cabo en cada subconjunto de computadores. Las tareas son desplegadas siguiendo un orden creciente de plazo, y se les asignan prioridades siguiendo el esquema DMS. Las ramas del árbol de búsqueda se podan en función de un límite inferior de tiempo de respuesta de peor caso.

BB se trata de uno de los métodos menos utilizados, ya que como puede observarse los pocos trabajos que hay datan de fechas ciertamente lejanas en comparación con otros trabajos recogidos aquí. Con este método se han abordado tanto planificaciones cíclicas como basadas en prioridades.

3.6. *Heurísticos (HEU)*

En (Gutiérrez and González Harbour, 1995) se desarrolla el heurístico denominado *Heuristic Optimized Priority Assignment* (HOPA) para la asignación de prioridades fijas a tareas y mensajes en sistemas de tiempo real distribuidos. Primero, se asignan plazos arbitrarios a las tareas y mensajes que componen flujos e2e, y después se emplea el esquema DMS para asignarles prioridades. Los tiempos de ejecución de peor caso se calculan mediante un análisis holístico o mediante análisis temporales basados en prioridades. Si no se cumple algún plazo en cualquier flujo, se computan una serie de métricas para determinar las desviaciones de dicho flujo sobre su plazo y se vuelven a calcular los plazos locales y a asignar las prioridades. Este proceso se repite hasta que se obtiene una solución válida o se alcanza un número máximo de iteraciones.

En el trabajo (Ramamritham, 1995) se propone un heurístico para desplegar tareas en computadores homogéneos y crear planificaciones cíclicas. Está basado en un modelo compuesto por plazos y periodos, y contempla la posibilidad de desplegar ciertas tareas replicadas en distintos computadores. En la primera fase del algoritmo se determina la idoneidad de agrupar y desplegar en el mismo computador parejas de tareas que se comunican entre sí. En la segunda se despliegan y planifican las tareas siguiendo el criterio *Latest Start Time / Maximum Immediate Successors First* (LST/MISF).

En (Braun et al., 2001) se realiza un estudio comparativo sobre once heurísticos empleados para desplegar tareas y crear planificaciones estáticas. El modelo se compone de tareas independientes con distintos tiempos de ejecución de peor caso según el elemento procesador en que sean desplegadas, y el objetivo es minimizar el tiempo de respuesta de la planificación total. Entre los once, los mejores resultados los obtienen un algoritmo genético y un heurístico que despliegan primero las tareas con el menor tiempo de ejecución en el computador en el que tengan el menor tiempo de ejecución de peor caso.

El mismo grupo de investigación presenta en (Braun et al., 2008) otro estudio comparativo de tres métodos para desplegar tareas en computadores y generar planificaciones cíclicas en sistemas de tiempo real distribuidos compuestos por computadores heterogéneos. Las tareas tienen relaciones de precedencia, prioridades, plazos y más de una versión de cada una de ellas. También tienen distinto tiempo de ejecución de peor caso según el computador en el que sean desplegadas, y se les asigna mayor prioridad cuanto mayor sea este tiempo de ejecución. Los algoritmos comparados son un heurístico y dos genéticos. El heurístico obtiene buenos resultados, y los dos genéticos mejoran estos resultados en un 2 % y 5 % respectivamente.

En (Ali et al., 2002) se proponen tres heurísticos de asignación de recursos a flujos e2e en sistemas de tiempo real distribuidos compuestos por computadores basados en planificación de tipo RR. Se contemplan flujos e2e con plazos y periodos. La carga del sistema se modela mediante la ratio de generación de información de los sensores de entrada a los flujos e2e. El objetivo es realizar tal despliegue de las tareas que maximice el crecimiento de carga permitido.

En (Eles et al., 2000) se proponen heurísticos para desplegar tareas en computadores y crear planificaciones cíclicas de tareas y mensajes en sistemas de tiempo real distribuidos compuestos por distintos tipos de procesadores. Un planificador basado en un heurístico *list-scheduling* es el responsable de asignar los recursos a las tareas o mensajes que se activan en función de sus prioridades. Asimismo, también se desarrolla un heurístico para optimizar el esquema de acceso a una red TTP. El mismo grupo de investigación ha incidido en la optimización del esquema de acceso a la red TTP en (Pop et al., 2000) y (Pop et al., 2004b), donde se proponen y analizan diferentes estrategias de mensajería y se exponen heurísticos para optimizar cada una de ellas.

Los trabajos (Pop et al., 2003b) y (Pop et al., 2004c) proponen heurísticos para desplegar tareas en computadores y planificar tareas y mensajes en sistemas de tiempo real distribuidos que combinan los paradigmas TT y ET. El modelo se basa en un grupo de computadores de clase TT conectados a través de una red de comunicaciones TTP, otro grupo de computadores de clase ET unidos mediante una red CAN y un *gateway* que enlaza ambas redes y por el que discurren los mensajes intercambiados entre ambos grupos. Los heurísticos propuestos despliegan las tareas en uno de los dos grupos, despliegan las tareas en alguno de los computadores del grupo y planifican las tareas y mensajes en sus respectivos recursos para que se cumplan los plazos de los flujos.

Otro enfoque abordado por los mismos autores en (Pop et al., 2001a) (Pop et al., 2001b) es el despliegue y la planificación de aplicaciones en sistemas de tiempo real distribuidos con el objetivo de optimizar la minimización de cambios en las aplicaciones ya desplegadas en el sistema y maximizar la probabilidad de que futuras aplicaciones puedan ser desplegadas en el mismo. Para ello se proponen unas métricas que modelan la probabilidad de que aplicaciones futuras requieran cierto tiempo de procesamiento cada cierto periodo.

En (Pop et al., 2003a), (Pop et al., 2005) y (Pop, 2007) proponen métodos para generar y optimizar la planificación de sistemas mixtos TT/ET. Uno de los métodos esta basado en ILP y es óptimo, mientras que el otro es un heurístico que obtiene resultados sub-óptimos en tiempos mas reducidos. El heurístico es guiado por unas reglas cuyo objetivo es aumentar el grado de planificabilidad del sistema a través de la modificación de las políticas de planificación de las tareas, el despliegue de las tareas a los computadores y el esquema de acceso a la red de comunicaciones.

En (Pedreiras and Almeida, 2004) se proponen dos heurísticos para la asignación de plazos locales a mensajes con plazos globales que cruzan múltiples redes TT desde su origen a su destino. El primer heurístico se denomina *Isometric Allocation* (ISO) y se basa en dividir el plazo global en tantas partes iguales como redes cruce el mensaje y asignar en cada una de ellas un plazo local igual a dichas partes. El segundo heurístico es *Maximum Schedulability Laxity* (MSL) y se basa en asignar plazos locales de forma proporcional al ancho de banda disponible en cada una de las redes cruzadas.

En (Qin and Jiang, 2006) se presenta un algoritmo para el despliegue y asignación de prioridades de tareas en sistemas distribuidos de tiempo real. Los procesadores son heterogéneos, por lo que las tareas tendrán distintos tiempos de ejecución según en cuál sean alojadas. Toda tarea tiene una copia que se ejecuta en caso de fallo y que debe ser desplegada en un computador distinto. Los objetivos son minimizar el tiempo de respuesta del sistema de manera que se cumplan todos los plazos, además de extender la fiabilidad del sistema en la medida de lo posible.

En (Pop, 2007) se proponen una serie de heurísticos para sintetizar, analizar y optimizar el despliegue de tareas en computadores, la asignación de políticas de planificación y prioridades a tareas y la asignación de *slots* de comunicación en computadores en sistemas de tiempo real distribuidos. La combinación de los paradigmas TT y ET se da a nivel tanto de computador como de red de comunicaciones. Las tareas TT se activan en instantes predefinidos en tablas de planificación, tienen máxima prioridad y no pueden ser expulsadas. Por su parte, las tareas ET se activan por medio de eventos externos y mensajes, y son expulsadas por las tareas TT y las tareas ET de mayor prioridad. Todas las tareas de un flujo son de alguno de los dos tipos descritos. Dado que la combinación de tareas y planificadores basados en diferentes paradigmas requiere la extensión de las técnicas de análisis de planificación existentes, se propone un heurístico de planificación y una extensión al análisis holístico del trabajo (Tindell and Clark, 1994) para sistemas que combinan tareas TT y ET.

Siguiendo el modelo de sistema descrito en (Al Sheikh et al., 2010), en (Al Sheikh et al., 2011) se emplea un heurístico basado en la teoría de juegos para desplegar y planificar particiones temporales en arquitecturas distribuidas basadas en IMA. Este algoritmo trabaja con el concepto de jugadores que adaptan sus estrategias basándose en las estrategias más recientes conocidas del resto de jugadores. Se demuestra lograr resultados casi óptimos en un tiempo de computación mucho menor que el logrado en (Al Sheikh et al., 2010). Tiene como objetivo, además de obtener una planificación que permita cumplir los requisitos de tiempo real, maximizar el tiempo ocioso del procesador dentro del MF para poder extender las funcionalidades del sistema en el futuro sin afectar a la planificación.

En (Eisenbrand et al., 2010) se propone un método híbrido basado en heurísticos y formulaciones matemáticas para resolver el despliegue y planificación de tareas periódicas en siste-

mas de tiempo real distribuidos. La arquitectura corresponde a la de un avión, y se divide en dos *cabinets*, formados por elementos procesadores idénticos comunicados mediante una red de comunicaciones. El objetivo es minimizar el número de procesadores mientras se cumplen todos los plazos de las tareas. Dichas tareas pueden tener requerimientos de co-habitación o exclusión mutua. No se considera la planificación de la red de comunicaciones.

El trabajo (Neukirchner et al., 2011) desarrolla un heurístico para la asignación de prioridades fijas a tareas y mensajes en sistemas de tiempo real distribuidos basados en un modelo de flujos con periodos y plazos. Para cada recurso planificable se ejecuta un módulo de asignación de prioridades y un módulo de análisis temporal comercial. Este heurístico obtiene peores resultados que el algoritmo genético propuesto en (Hamann et al., 2006), aunque requiere de unos tiempos de computación mucho menores.

En (Mehiaoui et al., 2013) se realiza una extensión al trabajo de (Wozniak et al., 2013), mencionado anteriormente, considerando el mismo modelo de sistema. El objetivo es minimizar las latencias de los flujos e2e mientras se cumplen los requisitos de tiempo real estrictos. Se desarrolla un heurístico basado en dos etapas. Primero se aborda el problema de distribuir los ejecutables en tareas y el del despliegue y planificación de éstas. La segunda etapa se basa en un proceso iterativo de dos niveles que da como resultado una solución optima, aunque sólo para sistemas a muy pequeña escala.

En (Klobedanz et al., 2013) se presenta un método para el despliegue y planificación de tareas y mensajes en sistemas de tiempo real distribuidos basados en .AUTOSAR. El objetivo es obtener una topología ECU-red reconfigurable y tolerante a fallos, mientras se cumplen los requerimientos de tiempo real. Cuando se detectan fallos se activan tareas redundantes alojadas en distintos nodos. La planificación de cada computador se realiza por prioridades según DMS.

En (Garibay Martínez et al., 2014) se muestra un algoritmo empleado para planificar tareas paralelas y distribuidas con requerimientos de tiempo real. Las denominadas tareas *fork/join* son planificadas mediante prioridades fijas de acuerdo al esquema DMS y ejecutadas en una plataforma distribuida compuesta por elementos procesadores de un solo núcleo idénticos, interconectados mediante una red de tiempo real. En un trabajo posterior (Garibay-Martínez et al., 2015) se propone un heurístico llamado DOPA (Distributed using Optimal Priority Assignment), como extensión del clásico OPA (Audsley, 1991) aplicado sobre sistemas de tiempo real distribuidos. Este algoritmo permite abordar dos problemas relacionados entre sí. Primero, la asignación óptima de prioridades se lleva a cabo para tareas independientes, para lo que las tareas con relaciones de precedencia deben transformarse en independientes mediante plazos intermedios. El segundo problema se refiere a la descomposición de ejecutables en tareas planificables. El algoritmo también busca alojar el mayor número de tareas en un mismo computador para minimizar la carga de mensajes enviados a través de la red.

En el trabajo (Yoon and Ryu, 2015) se desarrolla un algoritmo de planificación de tareas en sistemas de tiempo real distribuidos basados en arquitecturas AUTOSAR. Consiste en descomponer los flujos e2e en tareas con plazos locales y asignarles prioridades, de manera que estos plazos locales se cumplan. Primero el algoritmo resuelve estas dos cuestiones por separado, y después se desarrolla otro algoritmo que los aborda a la vez. No considera ningún tipo de planificación en la red de comunicaciones.

En (Hu et al., 2015) se presenta un algoritmo para planificar tareas y mensajes en sistemas de tiempo real distribuidos, en los que el despliegue de tareas sobre computadores se considera realizado previamente. Los flujos e2e pueden estar compuestos por tareas y mensajes periódicos o aperiódicos, y el objetivo es que todas las instancias de todas las aplicaciones cumplan sus plazos. El algoritmo primero ordena los flujos asignándoles prioridades, y después las tareas y mensajes que los conforman se planifican según el algoritmo SHLF (*Synchronized Highest Level First*). Finalmente se detallan dos procedimientos para dar soporte a futuras re-planificaciones.

En el trabajo (Chen et al., 2016) se modelan particiones (espacio-temporales) como tareas estrictamente periódicas y no expulsables, que son desplegadas y planificadas sobre arquitecturas distribuidas basadas en IMA. Primero se aplica un algoritmo basado en MILP para obtener un factor de escalado máximo para aplicar a cada partición temporal, y después un heurístico basado en la teoría de juegos permite determinar el despliegue y planificación de cada una de ellas. Pueden cumplirse requisitos de tiempo real ajustando el factor de escalado a los requisitos temporales concretos.

En (Xie et al., 2016) se despliegan y planifican tareas en un sistema de tiempo real distribuido, compuesto por computadores heterogéneos y varias redes interconectadas mediante *Gateways* que implementan funciones de criticidad mixta. La planificación se realiza mediante RR, y el objetivo del algoritmo es minimizar la ratio de plazos no superados mientras se mantiene un rendimiento satisfactorio.

En (Deroche et al., 2016) se desarrolla un algoritmo para el despliegue y la planificación de particiones en elementos procesadores de sistemas distribuidos de tiempo real. Las particiones temporales vienen previamente definidas mediante su periodo y su plazo. Se agrupan en flujos que se comunican mediante mensajes que también son considerados para el cómputo del tiempo de ejecución de peor caso total. El propio algoritmo decide el número de elementos procesadores empleados. En (Deroche et al., 2017) se extiende este trabajo empleando un algoritmo que limita el espacio de búsqueda de la solución óptima, descartando posibles soluciones sub-óptimas en cada etapa del algoritmo.

En (Bhat et al., 2017) se desarrolla un algoritmo para desplegar tareas en elementos procesadores de una arquitectura distribuida, planificándolas siguiendo el esquema RMS. El objetivo del algoritmo es minimizar el número de procesadores y las latencias de las tareas desplegadas en ellos, mientras se cumplen todos sus plazos. Se asume que estos plazos son iguales al periodo de cada tarea.

Este enfoque algorítmico es el más utilizado en la optimización del despliegue y planificación de sistemas de tiempo real distribuidos. Es probable que se deba a que son algoritmos eficientes, diseñados *ad-hoc* para resolver el problema que se plantea.

Tabla 1: Clasificación de los trabajos revisados

Algoritmo	Trabajo	Problemas abordados			Objetivos a minimizar			Restricciones	
		Despliegue	Planificación	Particiones	Computadores	Utilizaciones	Tiempos	Plazo-Periodo	Memoria
GA	(Mitra and Ramanathan, 1993)	Sí	Cíclica				Respuesta		
	(Hou et al., 1994)	Sí	Cíclica				Respuesta		
	(Monnier et al., 1998)	Sí	Cíclica				Respuesta	$D = T$	
	(Dick and Jha, 1998)	Sí	Cíclica		Coste	Energia		$T < D$	
	(Faucou et al., 2000)	Sí	Cíclica					$D = T$	
	(Oh and Wu, 2004)	Sí	Cíclica		Número		Respuesta		
	(Yoo, 2009)	Sí	Cíclica		Número		Respuesta		
	(Hamann et al., 2006)		Prioridades					$D < T$	
	(Shang et al., 2007)	Sí	Cíclica		Coste	Energia			Sí
	(Samii et al., 2009)		Prioridades				Respuesta		
	(Azketa et al., 2011a)	Sí	Prioridades		Número		Respuesta		
	(Wozniak et al., 2013)	Sí	Prioridades				Respuesta		Sí
	(Boutekkouk and Oubadi, 2014)	Sí	Prioridades				Respuesta		
	(Ayari et al., 2016a)	Sí	Prioridades				Respuesta		Sí
TS	(Porto et al., 2000)	Sí	Cíclica				Respuesta		
	(Chen and Lin, 2000)	Sí	Prioridades		Número	Red / Procesador	Respuesta		
	(Lin et al., 2000)	Sí	Cíclica				Respuesta		
	(Jiang et al., 2017)		Prioridades			Energía	Respuesta	$D < T$	
	(Tămaş-Selicean and Pop, 2011b)	Sí	Cíclica	Sí			Respuesta	$D < T$	
	(Tămaş-Selicean and Pop, 2015)	Sí	Cíclica	Sí		Coste certificación	Respuesta	$D < T$	
SA	(Tindell et al., 1992)	Sí	Prioridades			Red	Respuesta	$T = D$	Sí
	(Burns et al., 1993)	Sí	Cíclica			Tareas de enrutamiento	Respuesta		
	(Coli and Palazzari, 1995)	Sí	Cíclica				Respuesta		
	(Di Natale and Stankovic, 1995)		Cíclica				jitter		
	(Vargas and de Oliveira, 2005)	Sí	Cíclica						
	(He et al., 2010)	Sí	Prioridades				Respuesta	$D = T$	
	(Tămaş-Selicean and Pop, 2011a)	Sí	Jerárquica				Respuesta	$D \leq T$	
	(Pishdar and Akkasi, 2015)	Sí	Prioridades				Respuesta	$D = T$	
CSP	(Szymanek and Kuchcinski, 2001)	Sí	Cíclica			Memoria	Respuesta		Sí
	(Ekelin and Jonsson, 2001)	Sí	Cíclica			Red / Procesador	Respuesta	$D = T$	
SAT	(Hladik et al., 2008)	Sí	Prioridades					$D = T$	
	(Metzner and Herde, 2006)	Sí	Prioridades						
GP	(Davare et al., 2007)		Prioridades			Respuesta			
MILP	(Zheng et al., 2007b)	Sí	Prioridades				Respuesta		
	(Zheng et al., 2007a)		Prioridades						
	(Zhu et al., 2009)	Sí	Prioridades					T harmónicos	
	(Al Sheikh et al., 2010)		Cíclica						
	(Zhu et al., 2012)	Sí	Prioridades				Respuesta		
	(Blikstad et al., 2018)	Sí	Jerárquica						
CSP	(Craciunas and Oliver, 2014)	Sí	Jerárquica					$D \leq T$	
MIP	(Zhang et al., 2014)		Cíclica						
ILP	(Althaus et al., 2014)	Sí	Cíclica					$D \leq T$	
SMT - ILP	(Minaeva et al., 2018)		Cíclica						
BB	(Hou and Shin, 1997)	Sí	Cíclica						
	(Peng et al., 1997)	Sí	Cíclica				Respuesta		
	(Richard et al., 2003)	Sí	Prioridades						
HEU	(Gutiérrez and González Harbour, 1995)		Prioridades				Respuesta		
	(Ramamritham, 1995)	Sí	Cíclica					$D = T$	
	(Braun et al., 2001)	Sí	Cíclica				Respuesta		
	(Braun et al., 2008)	Sí	Cíclica				Respuesta		
	(Ali et al., 2002)	Sí	RR						
	(Pedreiras and Almeida, 2004)	Sí	Prioridades / Cíclica					$D = T$	
	(Eles et al., 2000)	Sí	Cíclica				Respuesta		
	(Pop et al., 2000)	Sí	Cíclica				Respuesta		
	(Pop et al., 2001a)		Prioridades					$D \leq T$	
	(Pop et al., 2004c)	Sí	Cíclica					$D \leq T$	
	(Pop, 2007)	Sí	Prioridades / Cíclica						
	(Qin and Jiang, 2006)	Sí	Prioridades				Respuesta		
	(Neukirchner et al., 2011)		Prioridades				Respuesta		
	(Eisenbrand et al., 2010)		Cíclica		Número			T harmónicos	
	(Al Sheikh et al., 2011)	Sí	Cíclica	Sí			Respuesta		
	(Mehiaoui et al., 2013)	Sí	Prioridades				Respuesta		
	(Klobedanz et al., 2013)	Sí	Prioridades				Respuesta		
	(Yoon and Ryu, 2015)	Sí	Prioridades				Respuesta		
	(Garibay-Martínez et al., 2015)	Sí	Prioridades			Uso de red		$D \leq T$	
	(Hu et al., 2015)		Prioridades						
	(Xie et al., 2016)	Sí	RR						
	(Chen et al., 2016)	Sí	Cíclica	Sí					
	(Deroche et al., 2016)	Sí	Cíclica	Sí					
	(Bhat et al., 2017)	Sí	Cíclica		Número		Respuesta	$D = T$	

4. Resultados y trabajo futuro

La Tabla 1 recoge todos los trabajos que se han revisado en este artículo. En ella se muestran los aspectos más característicos de cada contribución de acuerdo con la metodología expuesta en el apartado 2 y la revisión detallada del apartado 3. Cabe destacar que dado que en todos los trabajos recogidos el objetivo es la planificabilidad del sistema, no se hace mención explícita de ello en la tabla.

El despliegue y planificación de sistemas distribuidos de tiempo real se ha abordado en gran cantidad de trabajos a lo largo del tiempo. Existe una gran diversidad en los enfoques empleados a la hora de proponer herramientas de optimización para estos sistemas; si bien todos los recogidos en este artículo tienen en común el objetivo de cumplir con sus requerimientos de tiempo real, los hay que simplemente plantean la asignación de prioridades a las tareas (Gutiérrez and González Harbour, 1995), mientras que otros proponen la elaboración de ejecutivos cíclicos (Di Natale and Stankovic, 1995). Otros como (He et al., 2010) o (Pop, 2007) abordan a la vez, además de la planificación, el despliegue de las tareas y los mensajes en arquitecturas concretas. Además, hay trabajos que buscan, sin perder de vista su objetivo principal de cumplir con todos los plazos, optimizar ciertos parámetros como por ejemplo el uso de recursos, ya sean de computación o equipamiento (Azketa et al., 2011a).

El presente trabajo no tiene por objetivo realizar un estudio comparativo entre los trabajos recopilados. Ello se debe a que para poder llevar a cabo una comparación en la que se determine cuáles de todos los trabajos obtienen mejores resultados, los modelos de sistema de todos ellos deberían ser completamente homogéneos, así como los casos de estudio sobre los que se han implementado y validado. Ésto resulta imposible, por lo que las apreciaciones que pueden hacerse sobre los resultados obtenidos en este estudio son comentarios cualitativos obtenidos mediante una interpretación global de la tabla. Ésta en sí misma constituye un resultado que puede ser utilizado por futuros investigadores para encontrar las referencias que se adecúen a los modelos o algoritmos con los que pretendan trabajar.

Cabe destacar, tal y como puede verse en la tabla, el reducido número de trabajos que abordan el despliegue y la planificación de sistemas de tiempo real distribuidos basados en particiones. Esto abre la vía para la investigación en técnicas que contemplen esta característica tan necesaria en el desarrollo de las aplicaciones de criticidad mixta más modernas. Los avances en la investigación de técnicas de análisis temporal para sistemas particionados sin duda ayudarán a incrementar el número de soluciones propuestas para este problema que se encuentra en auge.

Otro aspecto destacable es que una gran mayoría de los trabajos busca configuraciones en las que se minimice el tiempo de respuesta del sistema. Resulta coherente, ya que en sistemas en los que el no cumplimiento de plazos puede producir graves consecuencias minimizar el tiempo de respuesta reduce las probabilidades de que ésto ocurra.

Por último, podría decirse que tras el exhaustivo análisis del estado del arte realizado, quedan modelos de sistema realistas que no han sido aún estudiados. También se ha constatado que determinados algoritmos y técnicas de búsqueda y optmización no han sido aplicados sobre sistemas que siguen el modelo y los esquemas de planificación utilizados en este trabajo, lo que nos motiva a continuar las labores de investigación en este sentido.

Agradecimientos

Este trabajo ha sido financiado en parte por el Gobierno de España y los fondos FEDER (AEI/FEDER, UE) en el proyecto TIN2017-86520-C3-3-R (PRECON-I4).

Referencias

Airlines Electronic Engineering Committee, A. R. I., 2009. Arinc specification 664p7: Aircraft data network, part 7 - avionics full duplex switched ethernet (afdx) network. AERONAUTICAL RADIO, INC 2551, 21401–7435.

Al Sheikh, A., Brun, O., Hladik, P.-E., 2010. Partition scheduling on an ima platform with strict periodicity and communication delays. In: 18th international conference on real-time and network systems. pp. 179–188.

Al Sheikh, A., Brun, O., Hladik, P.-E., Prabhu, B. J., 2011. A best-response algorithm for multiprocessor periodic scheduling. In: Real-Time Systems (ECRTS), 2011 23rd Euromicro Conference on. IEEE, pp. 228–237.
DOI: 10.1109/ECRTS.2011.29

Ali, S., Kim, J.-K., Siegel, H. J., Maciejewski, A. A., Yu, Y., Gundala, S. B., Gertphol, S., Prasanna, V. K., 2002. Greedy heuristics for resource allocation in dynamic distributed real-time heterogeneous computing systems. In: PDPTA. pp. 519–530.

Althaus, E., Hoffmann, S., Kupilas, J., Thaden, E., 2012. A column generation approach to scheduling of real-time networks. In: Proceedings of the World Congress on Engineering and Computer Science. Vol. 1.

Althaus, E., Hoffmann, S., Kupilas, J., Thaden, E., 2014. Scheduling of real-time networks with a column generation approach. In: IAENG Transactions on Engineering Technologies. Springer, pp. 397–412.
DOI: 10.1007/978-94-007-6818-528

Audsley, N. C., 1991. Optimal priority assignment and feasibility of static priority tasks with arbitrary start times. Technical Report.

AUTOSAR, 2003. Automotive open system architecture.
URL: http://www.autosar.org/

Ayari, R., Hafnaoui, I., Aguiar, A., Gilbert, P., Galibois, M., Rousseau, J.-P., Beltrame, G., Nicolescu, G., 2016a. Multi-objective mapping of full-mission simulators on heterogeneous distributed multi-processor systems. The Journal of Defense Modeling and Simulation.
DOI: 10.1177/1548512916657907

Ayari, R., Hafnaoui, I., Beltrame, G., Nicolescu, G., 2016b. Schedulability-guided exploration of multi-core systems. In: Proceedings of the 27th International Symposium on Rapid System Prototyping: Shortening the Path from Specification to Prototype. ACM, pp. 121–127.
DOI: 10.1145/2990299.2990319

Ayari, R., Hafnaoui, I., Beltrame, G., Nicolescu, G., 2018. Imga: an improved genetic algorithm for partitioned scheduling on heterogeneous multi-core systems. Design Automation for Embedded Systems, 1–15.
DOI: 10.1007/s10617-018-9208-1

Azketa, E., Uribe, J., Marcos, M., Almeida, L., Gutiérrez, J. J., 2011a. Permutational genetic algorithm for fixed priority scheduling of distributed real-time systems aided by network segmentation. In: Proceedings of the 1st Workshop on Synthesis and Optimization Methods for Real-time Embedded Systems.

Azketa, E., Uribe, J. P., Gutiérrez, J. J., Marcos, M., Almeida, L., 2012. Permutational genetic algorithm for the optimized mapping and scheduling of tasks and messages in distributed real-time systems. In: XV Jornadas de Tiempo Real.

Azketa, E., Uribe, J. P., Marcos, M., Almeida, L., Gutierrez, J. J., 2011b. Permutational genetic algorithm for the optimized assignment of priorities to tasks and messages in distributed real-time systems. In: Proceedings of the 8th IEEE International Conference on Embedded Software and Systems, pages 958-965. IEEE, pp. 958–965.
DOI: 10.1109/TrustCom.2011.132

Barrett, C., Tinelli, C., 2018. Satisfiability modulo theories. In: Handbook of Model Checking. Springer, pp. 305–343.

Baruah, S., Li, H., Stougie, L., 2010. Towards the design of certifiable mixed-criticality systems. In: Real-Time and Embedded Technology and Applications Symposium (RTAS), 2010 16th IEEE. IEEE, pp. 13–22.
DOI: 10.1109/RTAS.2010.10

Bhat, A., Samii, S., Rajkumar, R., 2017. Practical task allocation for software fault-tolerance and its implementation in embedded automotive systems. In: Real-Time and Embedded Technology and Applications Symposium (RTAS), 2017 IEEE. IEEE, pp. 87–98.
DOI: 10.1109/RTAS.2017.33

Blikstad, M., Karlsson, E., Lööw, T., Rönnberg, E., 2018. An optimisation approach for pre-runtime scheduling of tasks and communication in an integrated modular avionic system. Optimization and Engineering, 1–28.
DOI: 10.1007/s11081-018-9385-6

Bosch Gmbh, R., 1991. Can specification - version 2.0.

Boutekkouk, F., Oubadi, S., 2014. Periodic/aperiodic tasks scheduling optimization for real time embedded systems with hard/soft constraints. IT4OD, 135.

Boutekkouk, F., Oubadi, S., 2016. Real time tasks scheduling optimization using quantum inspired genetic algorithms. In: Artificial Intelligence Perspectives in Intelligent Systems. Springer, pp. 69–80.
DOI: 10.1007/978-3-319-33625-1-7

Boyd, S., Kim, S.-J., Vandenberghe, L., Hassibi, A., 2007. A tutorial on geometric programming. Optimization and engineering 8 (1), 67.
DOI: 10.1007/s11081-007-9001-7

Braun, T. D., Siegel, H. J., Beck, N., Bölöni, L. L., Maheswaran, M., Reuther, A. I., Robertson, J. P., Theys, M. D., Yao, B., Hensgen, D., et al., 2001. A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. Journal of Parallel and Distributed computing 61 (6), 810–837.
DOI: 10.1006/jpdc.2000.1714

Braun, T. D., Siegel, H. J., Maciejewski, A. A., Hong, Y., 2008. Static resource allocation for heterogeneous computing environments with tasks having dependencies, priorities, deadlines, and multiple versions. Journal of Parallel and Distributed Computing 68 (11), 1504–1516.
DOI: 10.1016/j.jpdc.2008.06.006

Burns, A., Davis, R. I., 2017. A survey of research into mixed criticality systems. ACM Computing Surveys (CSUR) 50 (6), 82.
DOI: 10.1145/3131347

Burns, A., Nicholson, M., Tindell, K., Zhang, N., 1993. Allocating and scheduling hard real-time tasks on a point-to-point distributed system. In: Proceedings of the Workshop on Parallel and Distributed Real-Time Systems. Citeseer, pp. 11–20.

Chen, J., Du, C., Han, P., 2016. Scheduling independent partitions in integrated modular avionics systems. PloS one 11 (12).
DOI: 10.1371/journal.pone.0168064

Chen, W.-H., Lin, C.-S., 2000. A hybrid heuristic to solve a task allocation problem. Computers & Operations Research 27 (3), 287–303.
DOI: 10.1016/S0305-0548(99)00045-3

Coli, M., Palazzari, P., 1995. A new method for optimization of allocation and scheduling in real time applications. In: Real-Time Systems, 1995. Proceedings., Seventh Euromicro Workshop on. IEEE, pp. 262–269.
DOI: 10.1109/EMWRTS.1995.514320

Craciunas, S. S., Oliver, R. S., 2014. Smt-based task-and network-level static schedule generation for time-triggered networked systems. In: Proceedings of the 22nd International Conference on Real-Time Networks and Systems. ACM, p. 45.
DOI: 10.1145/2659787.2659812

Craciunas, S. S., Oliver, R. S., Chmelík, M., Steiner, W., 2016. Scheduling real-time communication in ieee 802.1 qbv time sensitive networks. In: Proceedings of the 24th International Conference on Real-Time Networks and Systems. ACM, pp. 183–192.
DOI: 10.1145/2997465.2997470

Craciunas, S. S., Oliver, R. S., Ecker, V., 2014. Optimal static scheduling of real-time tasks on distributed time-triggered networked systems. In: Emerging Technology and Factory Automation (ETFA), 2014 IEEE. IEEE, pp. 1–8.
DOI: 10.1109/ETFA.2014.7005128

Crespo, A., Alonso, A., Marcos, M., Juan, A., Balbastre, P., 2014. Mixed criticality in control systems. IFAC Proceedings Volumes 47 (3), 12261–12271.
DOI: 10.3182/20140824-6-ZA-1003.02004

Davare, A., Zhu, Q., Di Natale, M., Pinello, C., Kanajan, S., Sangiovanni-Vincentelli, A., 2007. Period optimization for hard real-time distributed automotive systems. In: Proceedings of the 44th annual Design Automation Conference. ACM, pp. 278–283.

Deroche, E., Scharbarg, J.-L., Fraboul, C., 2016. Mapping real-time communicating tasks on a distributed ima architecture. In: Emerging Technologies and Factory Automation (ETFA), 2016 IEEE 21st International Conference on. IEEE, pp. 1–8.
DOI: 10.1109/ETFA.2016.7733586

Deroche, E., Scharbarg, J.-L., Fraboul, C., 2017. A greedy heuristic for distributing hard real-time applications on an ima architecture. In: Industrial Embedded Systems (SIES), 2017 12th IEEE International Symposium on. IEEE, pp. 1–8.
DOI: 10.1109/SIES.2017.7993390

Di Natale, M., Stankovic, J. A., 1995. Applicability of simulated annealing methods to real-time scheduling and jitter control. In: Real-Time Systems Symposium, 1995. Proceedings., 16th IEEE. IEEE, pp. 190–199.
DOI: 10.1109/REAL.1995.495209

Dick, R. P., Jha, N. K., 1998. Mogac: a multiobjective genetic algorithm for hardware-software cosynthesis of distributed embedded systems. IEEE transactions on computer-aided design of integrated circuits and systems 17 (10), 920–935.
DOI: 10.1109/43.728914

Directorate, E. C. I. S. M., 2012. Mixed criticality systems. Report from the Workshop on Mixed Criticality Systems.

Eisenbrand, F., Kesavan, K., Mattikalli, R. S., Niemeier, M., Nordsieck, A. W., Skutella, M., Verschae, J., Wiese, A., 2010. Solving an avionics real-time scheduling problem by advanced ip-methods. In: European Symposium on Algorithms. Springer, pp. 11–22.
DOI: 10.1007/978-3-642-15775-2-2

Ekelin, C., Jonsson, J., 2001. Evaluation of search heuristics for embedded system scheduling problems. In: International Conference on Principles and Practice of Constraint Programming. Springer, pp. 640–654.
DOI: 10.1007/3-540-45578-7-53

Eles, P., Doboli, A., Pop, P., Peng, Z., 2000. Scheduling with bus access optimization for distributed embedded systems. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 8 (5), 472–491.
DOI: 10.1109/92.894152

Emberson, P., Bate, I., 2010. Stressing search with scenarios for flexible solutions to real-time task allocation problems. IEEE Transactions on Software Engineering 36 (5), 704–718.
DOI: 10.1109/TSE.2009.58

Faucou, S., Deplanche, A.-M., Beauvais, J.-P., 2000. Heuristic techniques for allocating and scheduling communicating periodic tasks in distributed real-time systems. In: Factory Communication Systems, 2000. Proceedings. 2000 IEEE International Workshop on. IEEE, pp. 257–265.
DOI: 10.1109/WFCS.2000.882557

Fonseca, C. M., Fleming, P. J., 1998. Multiobjective optimization and multiple constraint handling with evolutionary algorithms. i. a unified formulation. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans 28 (1), 26–37.
DOI: 10.1109/3468.650319

Garibay Martínez, R., Nelissen, G., Ferreira, L. L., Pinho, L. M., 2014. On the scheduling of fork-join parallel/distributed real-time tasks. In: Industrial Embedded Systems (SIES), 2014 9th IEEE International Symposium on. IEEE, pp. 31–40.
DOI: 10.1109/SIES.2014.6871184

Garibay-Martínez, R., Nelissen, G., Ferreira, L. L., Pinho, L. M., 2015. Task partitioning and priority assignment for distributed hard real-time systems. Journal of Computer and System Sciences 81 (8), 1542–1555.
DOI: 10.1016/j.jcss.2015.05.005

Glover, F., 1986. Future paths for integer programming and links to artificial intelligence. Computers & operations research 13 (5), 533–549.
DOI: 10.1016/0305-0548(86)90048-1

Goldberg, D. E., Holland, J. H., 1988. Genetic algorithms and machine learning. Machine learning 3 (2), 95–99.
DOI: 10.1023/A:1022602019183

Goossens, K., Azevedo, A., Chandrasekar, K., Gomony, M. D., Goossens, S., Koedam, M., Li, Y., Mirzoyan, D., Molnos, A., Nejad, A. B., et al., 2013. Virtual execution platforms for mixed-time-criticality systems: the compsoc architecture and design flow. ACM SIGBED Review 10 (3), 23–34.
DOI: 10.1145/2544350.2544353ddff 10.1145/2544350.2544353

Guevara López, P., Valdez Martínez, J. S., Delgado Reyes, G., 2014. Planificadores de tareas en tiempo real concurrentes: Una clasificación basada en funciones y teoría de conjuntos. Computación y Sistemas 18 (4), 809–820.
DOI: 10.13053/CyS-18-4-1543

Gutiérrez, J. J., González Harbour, M., 1995. Optimized priority assignment for tasks and messages in distributed hard real-time systems. In: Parallel and Distributed Real-Time Systems, 1995. Proceedings of the Third Workshop on. IEEE, pp. 124–132.
DOI: 10.1109/WPDRTS.1995.470498

Gutiérrez, J. J., Palencia, J. C., Harbour, M. G., 2014. Holistic schedulability

analysis for multipacket messages in afdx networks. Real-Time Systems 50 (2), 230–269.
DOI: 10.1007/s11241-013-9192-2

Hamann, A., Jersak, M., Richter, K., Ernst, R., 2006. A framework for modular analysis and exploration of heterogeneous embedded systems. Real-Time Systems 33 (1-3), 101–137.
DOI: 10.1007/s11241-006-6884-x

He, X., Gu, Z., Zhu, Y., 2010. Task allocation and optimization of distributed embedded systems with simulated annealing and geometric programming. The Computer Journal 53 (7), 1071–1091.
DOI: 10.1093/comjnl/bxp084

Hladik, P.-E., Cambazard, H., Déplanche, A.-M., Jussien, N., 2008. Solving a real-time allocation problem with constraint programming. Journal of Systems and Software 81 (1), 132–149.
DOI: 10.1016/j.jss.2007.02.032

Holland, J., 1975. Adaptation in artificial and natural systems. Ann Arbor: The University of Michigan Press.

Hou, C.-J., Shin, K. G., 1997. Allocation of periodic task modules with precedence and deadline constraints in distributed real-time systems. IEEE transactions on computers 46 (12), 1338–1356.
DOI: 10.1109/12.641934

Hou, E. S., Ansari, N., Ren, H., 1994. A genetic algorithm for multiprocessor scheduling. IEEE Transactions on Parallel and Distributed systems 5 (2), 113–120.
DOI: 10.1109/71.265940

Hu, M., Luo, J., Wang, Y., Veeravalli, B., 2015. Scheduling periodic task graphs for safety-critical time-triggered avionic systems. IEEE Trans. Aerospace and Electronic Systems 51 (3), 2294–2304.
DOI: 10.1109/TAES.2015.140063

IEEE Portable Application Standards Committee, P., 2003. Standard for information technology-portable operating system interface (posix) realtime and embedded application support. std. 1003.13.

ISO/IEC, 2012. Ada 2012 reference manual. language and standard libraries - interna-tional standard iso/iec 8652:2012(e).

Jiang, W., Pop, P., Jiang, K., 2017. Design optimization for security-and safety-critical distributed real-time applications. Microprocessors and Microsystems 52, 401–415.
DOI: 10.1016/j.micpro.2016.08.002

John, R., 1999. Partitioning in avionics architectures: requirements, mechanisms, and assurance.

Kirkpatrick, S., 1984. Optimization by simulated annealing: Quantitative studies. Journal of statistical physics 34 (5-6), 975–986.
DOI: 10.1007/BF01009452

Klobedanz, K., Jatzkowski, J., Rettberg, A., Mueller, W., 2013. Fault-tolerant deployment of real-time software in autosar ecu networks. In: International Embedded Systems Symposium. Springer, pp. 238–249.
DOI: 10.1007/978-3-642-38853-8-22

Kopetz, H., 2011. Real-time systems: design principles for distributed embedded applications. Springer Science & Business Media.
DOI: 10.1007/978-1-4419-8237-7

Land, A. H., Doig, A. G., 1960. An automatic method of solving discrete programming problems. Econometrica: Journal of the Econometric Society, 497–520.
DOI: 10.2307/1910129

Lin, M., Karlsson, L., Yang, L. T., 2000. Heuristic techniques: Scheduling partially ordered tasks in a multi-processor environment with tabu search and genetic algorithms. In: Parallel and Distributed Systems: Workshops, Seventh International Conference on, 2000. IEEE, pp. 515–523.
DOI: 10.1109/PADSW.2000.884676

Liu, C. L., Layland, J. W., 1973. Scheduling algorithms for multiprogramming in a hard-real-time environment. Journal of the ACM (JACM) 20 (1), 46–61.
DOI: 10.1145/321738.321743

Liu, J., 2000. Real-time systems. Prentice Hall 48, 42.

Martí, R., Lozano, J. A., Mendiburu, A., Hernando, L., 2016. Multi-start methods. Handbook of Heuristics, 1–21.
DOI: 10.1007/0-306-48056-5-12

Mehiaoui, A., Wozniak, E., Tucci-Piergiovanni, S., Mraidha, C., Di Natale, M., Zeng, H., Babau, J.-P., Lemarchand, L., Gerard, S., 2013. A two-step optimization technique for functions placement, partitioning, and priority assignment in distributed systems. ACM SIGPLAN Notices 48 (5), 121–132.
DOI: 10.1145/2491899.2465572

Metzner, A., Herde, C., 2006. Rtsat–an optimal and efficient approach to the task allocation problem in distributed architectures. In: Real-Time Systems Symposium, 2006. RTSS'06. 27th IEEE International. IEEE, pp. 147–158.
DOI: 10.1109/RTSS.2006.44

Minaeva, A., Akesson, B., Hanzálek, Z., Dasari, D., 2018. Time-triggered co-scheduling of computation and communication with jitter requirements. IEEE Transactions on Computers 67 (1), 115–129.
DOI: 10.1109/TC.2017.2722443

Minoux, M., 1986. Mathematical programming: theory and algorithms. John Wiley & Sons.

Mitra, H., Ramanathan, P., 1993. A genetic approach for scheduling non-preemptive tasks with precedence and deadline constraints. In: System Sciences, 1993, Proceeding of the Twenty-Sixth Hawaii International Conference on. Vol. 2. IEEE, pp. 556–564.
DOI: 10.1109/HICSS.1993.284070

Monnier, Y., Beauvais, J.-P., Deplanche, A.-M., 1998. A genetic algorithm for scheduling tasks in a real-time distributed system. In: Euromicro Conference, 1998. Proceedings. 24th. Vol. 2. IEEE, pp. 708–714.
DOI: 10.1109/EURMIC.1998.708092

Neukirchner, M., Stein, S., Ernst, R., 2011. A lazy algorithm for distributed priority assignment in real-time systems. In: Object/Component/Service-Oriented Real-Time Distributed Computing Workshops (ISORCW), 2011 14th IEEE International Symposium on. IEEE, pp. 126–132.
DOI: 10.1109/ISORCW.2011.22

Oh, J., Wu, C., 2004. Genetic-algorithm-based real-time task scheduling with multiple goals. Journal of systems and software 71 (3), 245–258.
DOI: 10.1016/S0164-1212(02)00147-4

Palencia, J. C., Harbour, M. G., Gutiérrez, J. J., Rivas, J. M., 2017. Response-time analysis in hierarchically-scheduled time-partitioned distributed systems. IEEE Transactions on Parallel and Distributed Systems 28 (7).
DOI: 10.1109/TPDS.2016.2642960

Pearl, J., 1984. Heuristics: intelligent search strategies for computer problem solving. Addison-Wesley Pub. Co., Inc., Reading, MA.

Pedreiras, P., Almeida, L., 2004. Message routing in multi-segment ftt networks: The isochronous approach. In: Parallel and Distributed Processing Symposium, 2004. Proceedings. 18th International. IEEE, p. 122.
DOI: 10.1109/IPDPS.2004.1303082

Peng, D.-T., Shin, K. G., Abdelzaher, T. F., 1997. Assignment and scheduling communicating periodic tasks in distributed real-time systems. IEEE Transactions on Software Engineering 23 (12), 745–758.
DOI: 10.1109/32.637388

Pishdar, M. A., Akkasi, A., 2015. Task scheduling and idle-time balancing in homogeneous multi processors: A comparison between ga and sa. International Journal of Computer Applications 123 (13).
DOI: 10.5120/ijca2015905656

Pop, P., Eles, P., Peng, Z., 2000. Bus access optimization for distributed embedded systems based on schedulability analysis. In: Proceedings of the conference on Design, automation and test in Europe. ACM, pp. 567–575.
DOI: 10.1109/DATE.2000.840842

Pop, P., Eles, P., Peng, Z., 2004a. Analysis and synthesis of distributed real-time embedded systems. Springer Science & Business Media.
DOI: 10.1007/978-1-4020-2873-1

Pop, P., Eles, P., Peng, Z., 2004b. Schedulability-driven communication synthesis for time triggered embedded systems. Real-Time Systems 26 (3), 297–325.
DOI: 10.1109/RTCSA.1999.811257

Pop, P., Eles, P., Peng, Z., Izosimov, V., 2004c. Schedulability-driven partitioning and mapping for multi-cluster real-time systems. In: Real-Time Systems, 2004. ECRTS 2004. Proceedings. 16th Euromicro Conference on. IEEE, pp. 91–100.
DOI: 10.1109/EMRTS.2004.1311010

Pop, P., Eles, P., Pop, T., Peng, Z., 2001a. An approach to incremental design of distributed embedded systems. In: Proceedings of the 38th annual Design Automation Conference. ACM, pp. 450–455.
DOI: 10.1145/378239.378557

Pop, P., Eles, P., Pop, T., Peng, Z., 2001b. Minimizing system modification in an incremental design approach. In: Proceedings of the ninth international symposium on Hardware/software codesign. ACM, pp. 183–188.
DOI: 10.1145/371636.371718

Pop, T., 2007. Analysis and optimisation of distributed embedded systems with heterogeneous scheduling policies. Ph.D. thesis, Institutionen för datavetenskap.

Pop, T., Eles, P., Peng, Z., 2003a. Design optimization of mixed time/event-triggered distributed embedded systems. In: Proceedings of the 1st IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis. ACM, pp. 83–89.
DOI: 10.1109/CODESS.2003.1275264

Pop, T., Eles, P., Peng, Z., 2003b. Schedulability analysis for distributed heterogeneous time/event triggered real-time systems. In: Real-Time Systems, 2003. Proceedings. 15th Euromicro Conference on. IEEE, pp. 257–266.
DOI: 10.1109/EMRTS.2003.1212751

Pop, T., Pop, P., Eles, P., Peng, Z., 2005. Optimization of hierarchically scheduled heterogeneous embedded systems. In: Embedded and Real-Time Computing Systems and Applications, 2005. Proceedings. 11th IEEE International Conference on. IEEE, pp. 67–71.
DOI: 10.1109/RTCSA.2005.67

Porto, S. C., Kitajima, J. P. F., Ribeiro, C. C., 2000. Performance evaluation of a parallel tabu search task scheduling algorithm. Parallel Computing 26 (1), 73–90.
DOI: 10.1016/S0167-8191(99)00096-4

Qin, X., Jiang, H., 2006. A novel fault-tolerant scheduling algorithm for precedence constrained tasks in real-time heterogeneous systems. Parallel Computing 32 (5-6), 331–356.
DOI: 10.1016/j.parco.2006.06.006

Ramamritham, K., 1995. Allocation and scheduling of precedence-related periodic tasks. IEEE Transactions on Parallel and Distributed Systems 6 (4), 412–420.
DOI: 10.1109/71.372795

Richard, M., Richard, P., Cottet, F., 2003. Allocating and scheduling tasks in multiple fieldbus real-time systems. In: Emerging Technologies and Factory Automation, 2003. Proceedings. ETFA'03. IEEE Conference. Vol. 1. IEEE, pp. 137–144.
DOI: 10.1109/ETFA.2003.1247699

Rivas, J. M., Gutiérrez, J. J., Palencia, J. C., et al., 2011. Schedulability analysis and optimization of heterogeneous edf and fp distributed real-time systems. In: Real-Time Systems (ECRTS), 2011 23rd Euromicro Conference on. IEEE, pp. 195–204.
DOI: 10.1109/ECRTS.2011.26

Samii, S., Yin, Y., Peng, Z., Eles, P., Zhang, Y., 2009. Immune genetic algorithms for optimization of task priorities and flexray frame identifiers. In: Embedded and Real-Time Computing Systems and Applications, 2009. RTCSA'09. 15th IEEE International Conference on. IEEE, pp. 486–493.
DOI: 10.1109/RTCSA.2009.58

Schrijver, A., 1998. Theory of linear and integer programming. Wiley.

Sha, L., Abdelzaher, T., Årzén, K.-E., Cervin, A., Baker, T., Burns, A., Buttazzo, G., Caccamo, M., Lehoczky, J., Mok, A. K., 2004. Real time scheduling theory: A historical perspective. Real-time systems 28 (2-3), 101–155.

Shang, L., Dick, R. P., Jha, N. K., 2007. Slopes: hardware–software cosynthesis of low-power real-time distributed embedded systems with dynamically reconfigurable fpgas. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 26 (3), 508–526.
DOI: 10.1109/TCAD.2006.883909

Smith, D. J., Simpson, K. G., 2004. Functional Safety: A straightforward guide to applying IEC 61508 and related standards. Routledge.

Szymanek, R., Krzysztof, K., 2003. Partial task assignment of task graphs under heterogeneous resource constraints. In: Proceedings of the 40th annual Design Automation Conference. ACM, pp. 244–249.
DOI: 10.1145/775832.775895

Szymanek, R., Kuchcinski, K., 2001. A constructive algorithm for memory-aware task assignment and scheduling. In: Proceedings of the ninth international symposium on Hardware/software codesign. ACM, pp. 147–152.
DOI: 10.1109/HSC.2001.924666

Tămaş-Selicean, D., Pop, P., 2011a. Optimization of time-partitions for mixed-criticality real-time distributed embedded systems. In: Object/Component/Service-Oriented Real-Time Distributed Computing Workshops (ISORCW), 2011 14th IEEE International Symposium on. IEEE, pp. 1–10.
DOI: 10.1109/ISORCW.2011.11

Tămaş-Selicean, D., Pop, P., 2011b. Task mapping and partition allocation for mixed-criticality real-time systems. In: Dependable Computing (PRDC), 2011 IEEE 17th Pacific Rim International Symposium on. IEEE, pp. 282–283.
DOI: 10.1109/PRDC.2011.42

Tămaş-Selicean, D., Pop, P., 2015. Design optimization of mixed-criticality real-time embedded systems. ACM Transactions on Embedded Computing Systems (TECS) 14 (3), 50.
DOI: 10.1145/2700103

Tindell, K., Clark, J., 1994. Holistic schedulability analysis for distributed hard real-time systems. Microprocessing and microprogramming 40 (2-3), 117–134.
DOI: 10.1016/0165-6074(94)90080-9

Tindell, K. W., Burns, A., Wellings, A. J., 1992. Allocating hard real-time tasks: an np-hard problem made easy. Real-Time Systems 4 (2), 145–165.
DOI: 10.1007/BF00365407

Trujillo, S., Crespo, A., Alonso, A., Pérez, J., 2014. Multipartes: Multi-core partitioning and virtualization for easing the certification of mixed-criticality systems. Microprocessors and Microsystems 38 (8), 921–932.
DOI: 10.1016/j.micpro.2014.09.004

Tsang, E., 2014. Foundations of constraint satisfaction: the classic text. BoD–Books on Demand.

Vargas, L. M. F., de Oliveira, R. S., 2005. Empirical study of tabu search, simulated annealing and multi-start in fieldbus scheduling. In: Emerging Technologies and Factory Automation, 2005. ETFA 2005. 10th IEEE Conference on. Vol. 2. IEEE, pp. 8–pp.
DOI: 10.1109/ETFA.2005.1612668

Vestal, S., 2007. Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance. In: Real-Time Systems Symposium, 2007. RTSS 2007. 28th IEEE International. IEEE, pp. 239–243.
DOI: 10.1109/RTSS.2007.47

WindRiver, 2016. Wind river vxworks 653 platform.

Wozniak, E., Mehiaoui, A., Mraidha, C., Tucci-Piergiovanni, S., Gerard, S., 2013. An optimization approach for the synthesis of autosar architectures. In: Emerging Technologies & Factory Automation (ETFA), 2013 IEEE 18th Conference on. IEEE, pp. 1–10.
DOI: 10.1109/ETFA.2013.6647952

Xie, G., Zeng, G., Liu, L., Li, R., Li, K., 2016. High performance real-time scheduling of multiple mixed-criticality functions in heterogeneous distributed embedded systems. Journal of Systems Architecture 70, 3–14.
DOI: 10.1016/j.sysarc.2016.04.008

Yoo, M., 2009. Real-time task scheduling by multiobjective genetic algorithm. Journal of Systems and Software 82 (4), 619–628.
DOI: 10.1016/j.jss.2008.08.039

Yoon, H., Ryu, M., 2015. Guaranteeing end-to-end deadlines for autosar-based automotive software. International Journal of Automotive Technology 16 (4), 635–644.
DOI: 10.1007/s12239-015-0065-7

Zhang, L., Goswami, D., Schneider, R., Chakraborty, S., 2014. Task-and network-level schedule co-synthesis of ethernet-based time-triggered systems. In: Design Automation Conference (ASP-DAC), 2014 19th Asia and South Pacific. IEEE, pp. 119–124.
DOI: 10.1109/ASPDAC.2014.6742876

Zheng, W., Di Natale, M., Pinello, C., Giusto, P., Vincentelli, A. S., 2007a. Synthesis of task and message activation models in real-time distributed automotive systems. In: Design, Automation & Test in Europe Conference & Exhibition, 2007. DATE'07. IEEE, pp. 1–6.
DOI: 10.1109/DATE.2007.364573

Zheng, W., Zhu, Q., Di Natale, M., Vincentelli, A. S., 2007b. Definition of task allocation and priority assignment in hard real-time distributed systems. In: Real-Time Systems Symposium, 2007. RTSS 2007. 28th IEEE International. IEEE, pp. 161–170.
DOI: 10.1109/RTSS.2007.40

Zhu, Q., Yang, Y., Scholte, E., Di Natale, M., Sangiovanni-Vincentelli, A., 2009. Optimizing extensibility in hard real-time distributed systems. In: Real-Time and Embedded Technology and Applications Symposium, 2009. RTAS 2009. 15th IEEE. IEEE, pp. 275–284.
DOI: 10.1109/RTAS.2009.37

Zhu, Q., Zeng, H., Zheng, W., Natale, M. D., Sangiovanni-Vincentelli, A., 2012. Optimization of task allocation and priority assignment in hard real-time distributed systems. ACM Transactions on Embedded Computing Systems (TECS) 11 (4), 85.
DOI: 10.1145/2362336.2362352