Universidad Carlos III de Madrid
Escuela Politécnica Superior

Proyecto de fin de carrera de Ingeniero Informático

Herramienta de cálculo de mantenimiento cíclico basado en la fiablilidad

Autor: Alejandro Favieres Cuevas

Director: Dr. D. Jesús Carretero Pérez

Leganés, Febrero de 2011

Herramienta de cálculo de mantenimiento cíclico basado en la fiablilidad

Alejandro Favieres Cuevas

Resumen

Este trabajo propone diversos algoritmos para la creación de una planificación para tareas periodicas que se ejecutan dentro de un calendario real. Se realiza una comparación entre las distintas aproximaciones y se comprueba si consiguen el resultado deseado.

Abstract

This paper proposes several algorithms to create a plan for periodic tasks over a real calendar. A comparison between the aproaches taken is made and tests if the desirable results are reached.

Tabla de contenidos

1. Introducción 1
1.1. Marco de trabajo 1
1.2. Motivación 2
1.3. Objetivos 3
1.4. Trabajos relacionados 4
1.5. Estructura del documento 4
2. Estado de la técnica 6
2.1. Estructura y localización de los sistemas en una infraestructura ferroviaria 6
2.1.1. Circuitos de vía 6
2.1.2. Sistemas de señalización: Bloqueos 7
2.1.3. Enclavamientos 8
2.2. Problemas del mantenimiento 9
2.2.1. Tipos de mantenimiento 10
2.2.2. Metodología RCM 12
2.2.3. RCM aplicado al ferrocarril 17
2.3. Planificación de sistemas de tiempo real 20
2.3.1. Tipos de esquemas de planificación para tiempo real 21
2.3.2. Planificación factible 26
2.3.3. Planificación de tareas como búsqueda en un árbol 27
2.4. Técnicas de inteligencia artificial 29
2.4.1. Algoritmos de Simulated Annealing 30
3. Estudio del problema 35
3.1. Descripción del problema 35
3.2. Enfoques descartados 36
3.3. Solución por fuerza bruta 38
3.4. Armonización de períodos 39
3.5. Uso de algoritmos de recocido 41
4. Uso de Backtracking para planificar las tareas 42
4.1. Descripción de la técnica 42

4.2. Detalles de la implementación 47
4.3. Resultados 48
4.3.1. Influencia del número de tareas 48
4.3.2. Influencia del número de clases de equivalencia 53
4.4. Ejemplo de ejecución en un caso real 55
4.5. Evaluación de la solución 56
5. Armonización de períodos 59
5.1. Descripción de la técnica 59
5.1.1. Perturbaciones en los períodos 60
5.1.2. Introducción de una métrica 60
5.1.3. Seleccionar las mejores soluciones 61
5.1.4. Parámetros de la búsqueda 61
5.1.5. Detalles del algoritmo 62
5.2. Detalles de la implementación 63
5.3. Resultados 65
5.3.1. Influencia del número de tareas 65
5.3.2. Influencia del número de valores para cada período 67
5.3.3. Influencia del número de valores para el período principal..... 69
5.4. Ejemplo de ejecución en un caso real 71
5.5. Evaluación de la solución 71
6. Uso de simulated annealing para programar las tareas 73
6.1. Descripción de la técnica 73
6.2. Parámetros del algoritmo 77
6.3. Detalles de la implementación 79
6.4. Resultados 80
6.4.1. Comportamiento con respecto al número de tareas 80
6.5. Ejemplo de ejecución en un caso real 86
6.6. Evaluación de la solución 87
7. Conclusiones y trabajos futuros 89
7.1. Conclusiones del trabajo 89
7.2. Trabajos futuros 90
7.2.1. Mejoras sobre la definición del calendario 90

7.2.2. Mejoras sobre la gestión de los desplazamientos entre tareas 90
7.2.3. Mejoras sobre el proceso de armonización 91
7.2.4. Mejoras sobre el proceso de recocido 92
A. Presupuesto 93
A.1. Mano de obra 93
A.1.1. Tareas 94
A.2. Materiales 96
A.3. Resumen 97
Acrónimos 98
Bibliografía 99

Lista de figuras

1.1. Herramienta desarrollada 2
2.1. Curva de la bañera 13
2.2. Construcción de la planificación mediante asignación de tareas 28
2.3. Proceso de Simulated Annealing 33
4.1. Algoritmo por backtracking. 44
4.2. Partición del período total 45
4.3. Variación con el número de tareas (éxito) 49
4.4. Variación con el número de tareas (éxito) 50
4.5. Variación con el número de tareas (fallo) 51
4.6. Variación con el número de tareas (fallo) 52
4.7. Variación con el número de tareas (comparación) 53
4.8. Variación con el número de clases 54
4.9. Variación con el número de clases (logarítmico) 55
4.10. Ejecuciones sucesivas de la planificación obtenida 57
5.1. Variación con el número de tareas. 66
5.2. Variación con el número de tareas (escala logarítmica). 67
5.3. Variación con el número de valores por período. 68
5.4. Variación con el número de valores por período. 69
5.5. Variación con el número de valores para el período principal 70
6.1. Algoritmo de simulated annealing para la planificación de tareas 75
6.2. Comparación según el número de tareas (éxito) 81
6.3. Comparación según el número de tareas (fallo) 82
6.4. Comparación según el número de tareas (fallo) 83
6.5. Comparación según el número de tareas (fallo) 85
6.6. Duración de cada solución propuesta según el número de tareas 86

Lista de tablas

4.1. Variación con el número de tareas (éxito) 49
4.2. Variación con el número de tareas (fallo) 51
4.3. Variación con el número de clases de equivalencia 53
4.4. Tareas del trayecto 56
5.1. Multiplicadores según criticidad 65
5.2. Variación con el número de tareas 66
5.3. Variación con el número de valores por período 68
5.4. Variación con el número de valores para el período principal 70
6.1. Comparación de algoritmos en caso de éxito 80
6.2. Comparación según el número de tareas (fallo) 82
6.3. Tiempos de ejecución en simulated annealing 84
A.1. Fase de Análisis 94
A.2. Fase de Desarrollo 94
A.3. Fase de Pruebas 95
A.4. Fase de Documentación 95
A.5. Resumen por fases 95
A.6. Coste debido al software 96
A.7. Coste total del proyecto 97

Lista de ecuaciones

2.1. Distribución de Weibull. 14
2.2. Serie de disminución de frecuencias 20
2.3. Prueba de planificabilidad para tasa monótona. 22
2.4. Relación de recurrencia para tasa monótona. 23
2.5. Definición de tiempo de rodaja. 25
2.6. Prueba de planificabilidad para tiempo límite más corto. 25
2.7. Ecuación de Boltzmann 31
3.1. Demostración de que la relación es de equivalencia 38
A.1. Coste imputable por maquinaria 96

Capítulo 1. Introducción

1.1. Marco de trabajo

El presente trabajo se comenzó como parte del proyecto realizado por el grupo ARCOS de la Universidad Carlos III de Madrid para Renfe (hasta el 2005) y Adif (a partir del 2005).

El mencionado proyecto hereda su concepción del proyecto europeo RAIL (Reliability centered management Aproach for the Infraestructure and Logistics of railway operation). El proyecto RAIL supone un esfuerzo profesional por aplicar un sistema de mantenimiento más racional en las infraestructuras ferroviarias en toda Europa aplicando la metodología de revisiones RCM.

El trabajo desarrollado dentro del grupo para Renfe/ Adif se basa en aplicar los conceptos de RAIL a los métodos de operación y estructuración de las infraestructuras propias de dicha compañía española. El objetivo final consistió en proporcionar una herramienta para aplicar la metodología RCM al mantenimiento de las infraestructuras ferroviarias.

La herramienta, que ya está en uso en Adif, trabaja sobre los sistemas mantenibles del inventario de la compañía. Cada elemento de inventario se asigna a un nivel de criticidad según unos parámetros establecidos como pueden ser el impacto de un fallo en la explotación, el número de fallos o las condiciones del entorno.

La criticidad asociada al elemento modifica las frecuencias de revisión de tal modo que se minimizan las repercusiones de los fallos, no el número de los mismos.

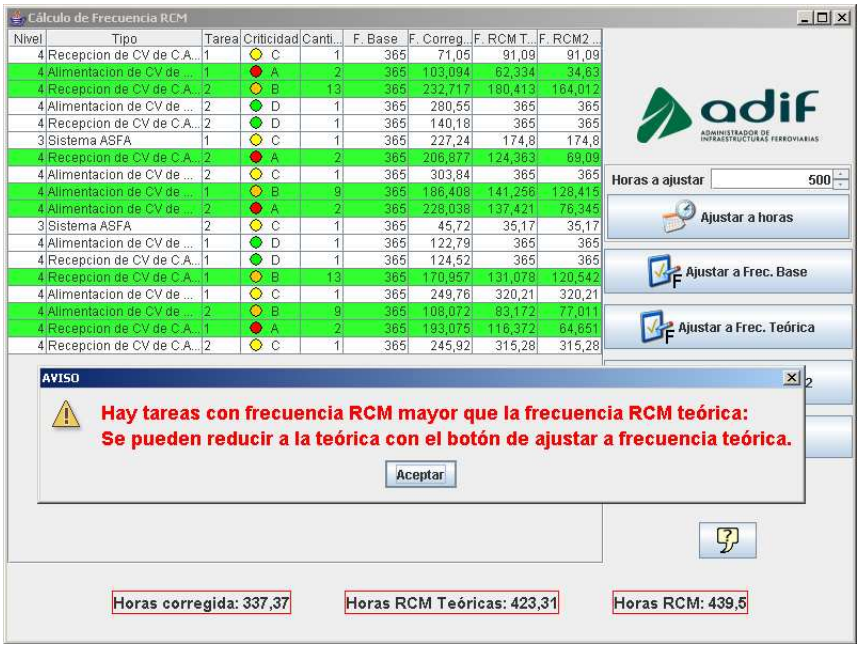

Figura 1.1. Herramienta desarrollada

Finalmente, una vez que se han fijado las frecuencias de revisión de los sistemas mantenibles, hay que programar las tareas de revisión propiamente dichas. Para la fuerza de trabajo de que se dispone hay que determinar qué sistemas hay que revisar cada día.

De ahí, surge la necesidad de planificar tareas periódicas con fuerza laboral limitada sobre calendarios reales.

1.2. Motivación

La planificación de las tareas de revisión, que son periódicas por naturaleza, dentro de calendarios reales forma parte de las funcionalidades de la herramienta mencionada. Dicho cálculo se realiza mediante backtracking como se detalla en capítulos posteriores. Sin embargo, el uso de esta funcionalidad introduce retardos considerables debido al gran costo computacional del algoritmo implementado.

Se toma como ejemplo de un inventario típico de Adif el tramo Villalba de Guadarrama - Cercedilla. Si se ejecuta el algoritmo basado en backtracking en una máquina típica de la época en que se realizó la implantación (Pentium III), tarda varias horas.

La funcionalidad queda más propia de un proceso batch que de un uso interactivo. Se pretende encontrar algoritmos más eficientes que permitan obtener respuestas más rápidamente.

1.3. Objetivos

El presente trabajo trata de aproximarse al problema de planificar la ejecución de tareas periódicas sobre calendarios laborales reales. En concreto, se intenta verificar si el uso de técnicas propias de la inteligencia artificial produce resultados mejores que el uso de algoritmos de backtracking.

Las tareas consideradas son aquellas propias de las revisiones de mantenimiento de las infraestructuras ferroviarias por parte de cuadrillas de mantenimiento.

El plan de revisiones debe ser establecido de tal modo que el tiempo entre revisiones de un sistema sea el adecuado según las técnicas RCM, que no se sobrepase la fuerza de trabajo diaria para poner en práctica el plan y que la distribución geográfica de los trabajos de mantenimiento sea lo más racional posible.

Si el objetivo anterior se cumple de forma satisfactoria, es razonable suponer que las mismas técnicas se pueden aplicar a problemas similares. Todos aquellos problemas que consistan en planificar tareas periódicas sobre calendarios reales en función de una jornada laboral de duración concreta serán fácilmente adaptables.

Por ejemplo, los planes de mantenimiento usados en otras industrias serán prácticamente idénticos, eliminando o sustituyendo por otro criterio la distribución geográfica.

1.4. Trabajos relacionados

Como se ha mencionado en un apartado anterior, el presente trabajo tiene sus orígenes en un proyecto realizado para Adif (anteriormente Renfe). El proyecto de fin de carrera "Herramienta para el análisis RCM aplicado al ferrocarril" hunde sus raíces en el mismo trabajo de investigación y, por tanto, la relación entre ambos es estrecha.

El objetivo de dicho proyecto de fin de carrera es determinar unas frecuencias (o períodos) de revisiones que optimicen el esfuerzo empleado [2]. Estos períodos, que se hallan aplicando la metodología RCM, determinan la frecuencia de las tareas que se tratan de planificar con este trabajo.

Desde ese punto de vista, se podría decir que la salida de el proyecto mencionado es el punto de partida de este proyecto. Desde luego que cada uno de estos trabajos se puede emplear por separado, pero cuando trabajan de forma coordinada pueden proporcionar un mejor resultado. Juntos proporcionan una planificación factible de las revisiones de mantenimiento tal que se tenga en cuenta el uso de los sistemas (al estilo RCM) y no se desperdicie, o no excesivamente, la fuerza de trabajo empleada.

1.5. Estructura del documento

A continuación se va a detallar cómo está estructurado el presente documento que refleja el trabajo realizado.

En este primer capítulo se ha recogido de dónde sale este proyecto y en qué circunstancias. Es un marco de referencia que permite ver que el trabajo realizado surge de una necesidad concreta.

En el segundo capítulo se sientan las bases necesarias para ver los cimientos sobre los que se va a construir este proyecto. Se va a realizar un recorrido sobre los elementos concretos sobre los que se va a realizar las planificaciones, los sistemas ferroviarios. Se pretende detallar los sistemas mantenibles concretos para que no se interponga

la terminología en la comprensión de los pasos realizados. En este capítulo también se realiza una explicación de qué es la metodología RCM, cómo se usa y porqué es una revolución en el mundo del mantenimiento. Por último, se fijan las bases necesarias para ver la relación del problema considerado, los sistemas de tiempo real y las búsquedas en un árbol, en concreto las búsquedas en árbol mediante técnicas de Inteligencia Artificial.

En el tercer capítulo la temática se va a centrar en los distintos enfoques planteados para resolver el problema. Se determinan los distintos procedimientos por los que se pretende hallar una solución y se da el motivo por el que se han elegido esos.

El cuarto capítulo se centra en detallar el algoritmo por backtracking usado y el comportamiento que ofrece cuando varía la carga.

El quinto capítulo plantea las ventajas e inconvenientes de realizar un proceso previo a la planificación que modifique ligeramente el conjunto de tareas a planificar de un modo tal que los períodos de las tareas sean armónicos.

El sexto capítulo introduce el cálculo de la solución mediante simulated annealing. Se comprueba el comportamiento que tiene este algoritmo con respecto al algoritmo por backtracking y su respuesta ante la variación de la carga computacional.

El séptimo capítulo reflexiona sobre las conclusiones extraídas del trabajo realizado. También se indican los posibles puntos dónde se pueden realizar mejoras o se pueden desarrollar evoluciones sobre lo expuesto.

El apéndice A contiene un presupuesto del trabajo realizado. El presente documento refleja un trabajo de investigación, pero conlleva una cierta cantidad de trabajo y ciertos materiales. El presupuesto presentado refleja el coste que supondría todo ello si fuera un trabajo remunerado.

Capítulo 2. Estado de la técnica

En este capítulo se introducirán diferentes temas que proporcionan el contexto necesario para comprender la relevancia de los conceptos desarrollados en este trabajo. A partir de estos bloques básicos se construye el presente trabajo.

2.1. Estructura y localización de los sistemas en una infraestructura ferroviaria

El presente trabajo se ha realizado sobre las tareas de revisión y mantenimiento de las infraestructuras ferroviarias. Para tener una comprensión clara de qué es lo que se está planificando y dónde, es extremadamente valioso tener un conocimiento claro de cuáles son los sistemas mantenibles de los que se está hablando y dónde se ubican.

La infraestructura ferroviaria incluye todas las instalaciones y edificaciones necesarias para el funcionamiento del ferrocarril: estaciones, vías puentes y túneles, sistema de señales y comunicaciones, infraestructura de bloqueo de trenes y guiado, agujas, etc.

A continuación se explica brevemente en qué consisten los sistemas mantenibles más importantes de la infraestructura ferroviaria [1].

2.1.1. Circuitos de vía

Son instalaciones eléctricas dispuestas de tal modo que, cuando el tren las alcanza, se cierra el circuito. Permiten situar a los trenes en determinados puntos de la vía y activar las señales adecuadas.

2.1.2. Sistemas de señalización: Bloqueos

El objeto del bloqueo es garantizar la seguridad en la circulación de los trenes, manteniendo entre ellos la distancia necesaria para que en su marcha no se choquen ni se alcancen entre ellos. Hay diferentes tipos de bloqueo.

2.1.2.1. Bloqueo telefónico (BT)

El bloqueo de los cantones se realiza mediante el envío de telefonemas entre los jefes de circulación.

2.1.2.2. Bloqueo eléctrico manual (BEM)

El bloqueo de vía se realiza por los jefes de circulación mediante un dispositivo eléctrico.

2.1.2.3. Control de circulación por radio (CCR)

En este bloqueo, usado en líneas de escaso tráfico, la petición y concesión de vía es realizada directamente por los maquinistas de los trenes en contacto con el Puesto de Control de Tráfico.

2.1.2.4. Bloqueo automático (BA)

Este bloqueo cuenta, en general, con cantones intermedios entre estaciones, los cuales quedan protegidos de modo automático por las señales.

En función de las condiciones de señalización y de la vía se distinguen los siguientes subtipos:

- Bloqueo automático de vía única (BAU).

  Solo hay circulación en uno de los sentidos.

- Bloqueo automático de vía doble (BAD).

  Solo hay secuencia automática de señales en cada vía en su sentido correspondiente.

- Bloqueo automático banalizado (BAB).

  La circulación se puede realizar en cada vía indistintamente en cualquiera de los sentidos.

2.1.2.5. Bloqueo de liberación automática (BLA)

Este bloqueo cuenta, en general, con un solo cantón entre estaciones, el cual es protegido por las señales de manera automática.

En función de las condiciones de señalización y de la vía se distinguen subtipos de manera análoga al BA.

2.1.2.6. Bloqueo de control automático (BCA)

En este bloqueo, la distancia de seguridad se mantiene indicando en la cabina de conducción una velocidad límite, una distancia meta y una velocidad meta.

2.1.2.7. Bloqueo de señalización lateral (BSL)

En este bloqueo la distancia de seguridad se garantiza cuando los trenes respetan las indicaciones de las señales.

2.1.3. Enclavamientos

Son dispositivos que restringen la apertura de señales ferroviarias o el movimiento de agujas, calces y semibarreras en función del estado de agujas, calces, semibarreras y señales, así como de la ocupación de los cantones. Se suele referir a los dispositivos que controlan los elementos de una estación ferroviaria y sus inmediaciones.

2.2. Problemas del mantenimiento

Los sistemas complejos forman parte de la vida diaria. Se pueden encontrar en el hogar (electrodomésticos), en la industria y, lo que resulta relevante para este proyecto, en los transportes. En concreto, los sistemas de señalización ferroviaria son sistemas complejos.

Los sistemas complejos fallan. Cuando un mecanismo es complejo, tiene diversas partes en las que puede haber un problema inutilizando, en la práctica, el sistema completo.

Las operaciones de mantenimiento tienen lugar para minimizar la probabilidad de un fallo en un sistema, maquinaria o equipo. Los fallos tienen diversos efectos negativos. Para empezar, impiden desarrollar la actividad del negocio y, por tanto, repercuten en la obtención de ingresos. Otro impacto negativo de los fallos es que pueden afectar a la imagen de marca, si se ha invertido un capital para crear imagen de marca, ese esfuerzo puede verse desperdiciado debido a determinados fallos muy visibles o repetidos. En el caso de sistemas de seguridad, como las señalizaciones ferroviarias, un fallo puede tener consecuencias más severas: daños personales o, incluso, la pérdida de vidas humanas.

El reto del mantenimiento consiste en mantener o mejorar la fiabilidad o la disponibilidad de los sistemas mantenibles reduciendo los costes. Los métodos para efectuar el mantenimiento han sufrido reiteradas revisiones y mejoras desde que se empezó a vislumbrar su importancia, a finales de la Segunda Guerra Mundial, hasta hoy en día.

La *fiabilidad* se define como la probabilidad, durante un período de tiempo determinado, de que el equipo en cuestión pueda realizar su función o su actividad, en las condiciones de utilización, o sin avería. La *disponibilidad* es el porcentaje de equipos o sistemas útiles en un determinado momento frente al parque total de equipos o sistemas.

2.2.1. Tipos de mantenimiento

La norma AFNOR NFX 60-010 considera que existen tres tipos básicos de mantenimiento: correctivo, preventivo y predictivo. Se va a discutir a continuación, las peculiaridades de cada uno de estos tipos de mantenimiento.

2.2.1.1. El mantenimiento correctivo

Este tipo de mantenimiento fue el primero en aparecer y, prácticamente, el único existente hasta la explosión de la ingeniería posterior a la Segunda Guerra Mundial.

Este mantenimiento consiste en realizar las operaciones de mantenimiento exclusivamente cuando se detecta un fallo en el funcionamiento. Según la importancia del fallo se puede forzar la parada de la máquina hasta que se resuelva o se puede planificar para un futuro próximo.

Su ventaja más importante es que es el método más simple, la planificación se limita a la disponibilidad de repuestos. La desventaja principal es que los paros por avería son, en general, más frecuentes que con el resto de técnicas. Este tipo de mantenimiento puede ser adecuado cuando el sistema mantenido no es muy caro, los fallos no provocan que se pare el sistema o cuando dichas paradas no son un gran contratiempo.

2.2.1.2. El mantenimiento preventivo

Como se ha indicado antes, aproximadamente desde el final de la Segunda Guerra Mundial la industrialización tuvo un gran desarrollo en Europa debido a diversos factores. Como consecuencia de los avances de los sectores industriales fabriles y de la proliferación de la competencia, se produce una evolución en la concepción

del papel del mantenimiento industrial. Los equipos son cada vez más complejos y automatizan más tareas. Las paradas de la maquinaria son cada vez más costosas y las averías provocan resultados más destructivos Debido a las nuevas necesidades se desarrolla el mantenimiento de tipo preventivo [10].

Los objetivos que se plantean para el mantenimiento ya no son realizar la reparación lo más rápido y barato posible. Ahora lo que se desea es la disponibilidad de los medios de producción o transporte. Asimismo, se espera que los equipos duren lo máximo posible en condiciones operativas idóneas. Por supuesto, se pretende que los costes sean lo más bajos posible.

Este tipo de mantenimiento consiste en revisar de forma cíclica los equipos, instalaciones y medios en general. El plan de revisión, que incluye la duración de los ciclos para todos los sistemas mantenibles, intenta reemplazar los sistemas justo antes de que se produzca el fallo.

Al aplicar este tipo de mantenimiento se evitan muchos de los fallos que se hubiesen producido usando un mantenimiento correctivo. Esto se consigue sin modificar los sistemas mantenibles, solo se modifica la metodología. Como veremos más adelante, el mantenimiento predictivo requiere equipamiento adicional que supone una mayor inversión inicial. Ya que el mantenimiento preventivo no requiere realizar modificaciones sobre los equipamientos existentes para implantarlo, es ventajoso frente al predictivo cuando los sistemas mantenibles son muy caros o están poco amortizados.

El mantenimiento predictivo es más efectivo realizando las revisiones precisamente cuando hacen falta, ni antes ni después. Se realizaron numerosos ajustes a los ciclos de revisiones. Pronto se llego a un equilibrio, o estancamiento si se quiere ver así, entre las variables fiabilidad, disponibilidad y costes. Una mejora en una de ellas produce un empeoramiento de una, o ambas, de las restantes variables.

2.2.1.3. El mantenimiento predictivo

En la década de los años ochenta, era evidente que las técnicas de mantenimiento preventivo estaban estancadas, como ya se ha comentado antes. No se conseguía salir del equilibrio entre fiabilidad, disponibilidad y costes. Debido a esta limitación se comenzó a plantearse alternativas. Sumado a esto, la "curva de la bañera", en la que se basaban las técnicas de mantenimiento preventivo en aquel entonces, resultó que representaba tan solo a un pequeño porcentaje de los equipos modernos. Por otro lado, el aumento de la reglamentación normativa imponía criterios cada vez más estrictos en los que operar, desde cuestiones de seguridad a medioambientales.

La solución adoptada en el mantenimiento predictivo consistió en reducir las revisiones periódicas a aquellos casos en que fuera necesario u obligase la normativa. Los sistemas se mantienen o se sustituyen, no según un plan preconcebido, si no según su estado real. Para ello es necesario conocer en todo momento el estado de todos los sistemas mantenibles. En realidad, se controlan ciertas variables del sistema que se ha comprobado que son indicativas de los modos de fallo. En general, se controlan dichas variables mediante sensores u otros dispositivos de medida.

Su ventaja fundamental es que no se realizan revisiones innecesarias. La mano de obra se aplica donde es útil. Además, se pueden evitar todos los fallos excepto los repentinos o inesperados. Como desventaja se puede apuntar el alto coste de los sensores y el equipamiento de aviso necesarios para realizar el seguimiento, sobre todo si se trata de realizar reingeniería sobre los procesos existentes.

2.2.2. Metodología RCM

Desde los comienzos del mantenimiento hasta hoy en día, los paradigmas en que se apoya han variado mucho, como ya se ha indicado en apartados anteriores. Tanto los supuestos, como los objetivos que se manejan han evolucionado en gran medida desde los orígenes del mantenimiento hasta los que se tienen actualmente.

La experiencia y la comprensión de los procesos implicados, han permitido eliminar supuestos erróneos y enfocar objetivos más cercanos a lo que se espera de los sistemas.

Como ejemplo se puede tomar la frecuencia del fallo. Anteriormente se pensaba que la frecuencia del fallo con respecto al tiempo tenía una zona con alta probabilidad en los primeros momentos de la vida útil del equipo, que luego se estabilizaba con una frecuencia de fallo baja y que, con el desgaste, volvía a crecer hacia el final de la vida útil media del mismo. Es lo que se conoce como la curva de la bañera.

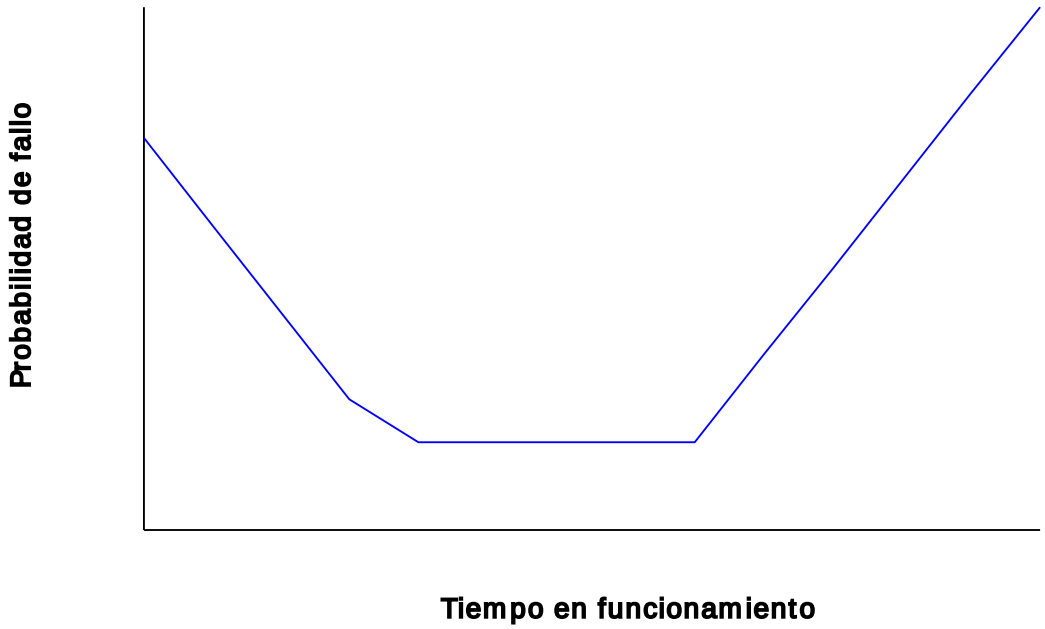

Figura 2.1. Curva de la bañera

Sin embargo, la frecuencia de fallo en los equipos sigue en realidad una distribución de Weibull. En algunos casos, la distribución de Weibull sí se adapta a la curva de la bañera, pero, en general, no tiene por que ser así. La consecuencia de este hallazgo es que la revisión cíclica de los componentes es útil para aquellos en los que la probabilidad de fallo aumente con el tiempo pero es inútil para aquellos en los que la probabilidad de fallo disminuye con el tiempo.

$$F(t) = 1 - e^{-(\frac{t-t_0}{T-t_0})^{\beta}}$$

Ecuación 2.1. Distribución de Weibull.

Donde las variables que aparecen en la formula representan lo siguiente:

La letra t. Es la variable de duración. Se suele considerar que representa el tiempo de uso, pero puede ser representada en forma de kilómetros (para transportes), servicios (para sistemas que son de uso puntual y no continuo, como las armas) o cualquier otra que sea apropiada.

La letra T. Duración característica media. Se pueden considerar varios valores distintos pero, en general, se usa el valor para el cual han fallado el 63,2% de los sistemas (distribución normal).

La letra t_0. Si interesa el tiempo medio entre fallos se puede usar como valor de t_0 el momento en que ha ocurrido el fallo anterior. Sin embargo, si lo que se quiere estudiar es toda la vida del sistema hay que usar el valor 0 para esta variable.

La letra β. Es un parámetro siempre mayor que 0. Según el valor que tome describe el grado de variación de la tasa de fallos. Cuando es menor que la unidad, la tasa de fallos disminuye con el tiempo; cuando es exactamente la unidad, la tasa de fallos permanece constante en el tiempo y cuando es mayor que la unidad, la tasa de fallos crece con el tiempo.

El desarrollo en los objetivos y supuestos ha permitido que se creen técnicas y metodologías novedosas que integran los hallazgos realizados. Una de las metodologías más exitosas que ha surgido de esta evolución es RCM.

RCM es el acrónimo de Reliability Centered Maintenance (Mantenimiento Centrado en la Fiabilidad). Esta metodología proporciona un marco de trabajo que permite determinar las técnicas más adecuadas para tratar cada tipo de fallo de la forma mas duradera y económica. La idea de la metodología RCM es mejorar el plan de mantenimiento para aumentar la fiabilidad y, además, reducir los costes del mantenimiento aplicando el esfuerzo allí donde resulte más provechoso. [3].

Las ventajas que puede aportar RCM repercuten en diversos ámbitos. Algunas de las ventajas que se pueden obtener de aplicarlo se muestran a continuación.

- Costes
  - Reducir los costes del mantenimiento preventivo rutinario.
  - Definir directrices concretas para sustituir preventivos por predictivos.
  - Reducir las paradas en producción mediante reingenieria.
- Servicio
  - Conocer mejor los requerimientos de servicio del cliente.
  - Definir de forma consensuada niveles de calidad de servicio.
  - Reducir las averías, sobre todo las que repercuten en el servicio.
- Calidad
  - Incremento de la disponibilidad.
  - Eliminación de fallos crónicos.
  - Mejor documentación del cambio y sistema auditable.
- Tiempo
  - Reducción de las paradas programadas para grandes revisiones.
  - Intervalos más largos entre paradas.

    - Tiempo de intervención más corto por mejor conocimiento del sistema.

- Riesgos

    - Mayor aseguramiento de la integridad de la seguridad y entorno.

    - Análisis de fallos ocultos y sus causas.

    - Reducción de la probabilidad de fallos múltiples.

Esta metodología se basa en el principio de que todo equipo en una instalación industrial está pensado para cumplir una función. Por tanto, no se desea mantener el estado del equipo, sino la función que cumple dentro del conjunto.

El primer paso para aplicar RCM consiste en determinar para cada equipo cuál es la función que los usuarios desean que realice. Adicionalmente se debe comprobar que el equipo es capaz de realizar dicha tarea.

Después hay que estudiar cuáles son los fallos funcionales (estados en los que el equipo es incapaz de cumplir la función a unos niveles aceptables para el usuario) para cada equipo.

Un tercer paso consiste en determinar los modos de fallo, que son las diferentes formas en que un equipo puede, de forma sensata, llegar a un fallo funcional. Dentro de los modos de fallo hay que incluir los que se han producido alguna vez, los que se han evitado gracias al mantenimiento actual y aquellos que no han ocurrido todavía pero tienen probabilidad de ocurrir.

El siguiente paso consiste en enumerar los efectos del fallo que describe las consecuencias de que ocurra cada uno de los modos de fallo. Estas descripciones deben contener toda la información necesaria para ayudar a la evaluación de las consecuencias. Esto incluye las consecuencias de que se produjera de forma simultáneamente un modo de fallo asociado.

El quinto paso consiste en evaluar las consecuencias del fallo en la operación, la calidad, el entorno, la seguridad o el servicio al cliente. El esfuerzo dedicado al mantenimiento de los equipos está muy influenciado por la importancia de las consecuencias del fallo. Puede haber equipos que no merezca la pena mantener, si no esperar a que falle para hacer el mantenimiento.

El último paso consiste en considerar si es posible prevenir el fallo. Si es posible, determinar cómo se puede conseguir. En cualquier caso, hay que determinar el coste de las tareas de prevención y el coste que la detección precoz del fallo ahorra.

A partir de las consecuencias se asignan criticidades a los sistemas. Según la criticidad de cada sistema se modifican sus períodos de revisión teóricos, de modo que los sistemas más críticos sean revisados más a menudo que los sistemas idénticos pero menos críticos.

2.2.3. RCM aplicado al ferrocarril

Sobre los sistemas mantenibles pertenecientes a la infraestructura ferroviaria que se han mencionado en un apartado anterior, se quiere aplicar un mantenimiento según la metodología RCM.

En el caso de Adif, ya existe un inventario de los sistemas existentes y en uso que incluye su ubicación dentro de la red de transporte. Si no existiera, sería necesario realizarlo como paso previo a la aplicación de la metodología.

La red, en el contexto de Adif, tiene distintos ámbitos. Se pueden considerar como una serie de subconjuntos, donde cada elemento se encuentra dentro de un elemento del nivel superior y contiene uno o más elementos del nivel inferior. De más general a más concreto son los siguientes:

- Red

- Gerencia
- Jefatura
- Línea
- Trayecto
- Tramo (Bloqueos o enclavamiento)
- Agrupación de sistemas
- Sistema
- Equipo

El ámbito base sobre el que se va a trabajar es el trayecto. Se va a considerar que el mantenimiento de cada trayecto es independiente del de los demás. Dentro de un trayecto, los sistemas mantenibles RCM van a ser los elementos que dentro de la jerarquía de Adif son los llamados sistemas y los equipos.

Los bloqueos y enclavamientos del trayecto considerado reciben una criticidad según las técnicas aplicadas [2]. Esta criticidad puede aplicarse a todos los sistemas mantenibles que pertenecen a ese trayecto. También se pueden definir valores concretos para dichos sistemas modificando sus parámetros. Por último, se pueden marcar sistemas como no mantenibles.

Para sistemas individuales, se pueden asignar valores concretos de los parámetros definidos para determinar la criticidad. A partir de los parámetros que tenga dicho sistema, se calcula la criticidad que va a tener el sistema. La criticidad será un valor que va desde el tipo A, el más crítico , hasta el tipo D, el menos crítico.

Se pueden asignar criticidades a elementos de un nivel distinto de los sistemas, por ejemplo a las agrupaciones. También se puede heredar la criticidad de elementos de un nivel distinto de tramo (bloqueo y enclavamiento).

Una vez que los sistemas tienen asignada una criticidad, se calcula para cada uno de ellos las frecuencias RCM teóricas. Esta frecuencia es la resultante de aplicar el multiplicador asociado a la criticidad a la frecuencia corregida del sistema mantenible sin tener en cuenta la fuerza laboral disponible.

Finalmente, se introduce en los cálculos la fuerza laboral disponible. El modo de llevar a cabo esto depende de si las horas de trabajo disponibles son más numerosas que las horas de revisión mediante RCM calculadas de forma teórica o si, por el contrario, las horas reales son menos que las teóricas, En el caso en el que ambos valores coincidan, no sería necesario realizar ningún ajuste posterior.

Para el caso en el que hay más horas laborales disponibles que las horas teóricas calculadas usando las técnicas que usan RCM, se realiza un aumento lineal en la frecuencia de las revisiones usando como pendiente un valor distinto para cada valor de la criticidad. Es decir, el crecimiento de las frecuencias sigue una tasa distinta para cada valor de la criticidad.

Cuando las horas disponibles no son suficientes para las frecuencias teóricas calculadas, el proceso es algo más complicado. Hay que cumplir una serie de condiciones:

- Se debe empezar a reducir las frecuencias por aquellos elementos que tengan menor criticidad.
- Se debe reducir más la frecuencia de los elementos menos críticos que la de los de mayor criticidad
- La reducción de frecuencias ha de ser equitativa

La solución seleccionada consiste en un proceso iterativo. Se van considerando distintas cuaternas de valores. Cada uno de esos valores es un multiplicador asociado a una criticidad. La frecuencia teórica encontrada se multiplica por el factor correspondiente a la iteración para encontrar la frecuencia ajustada. Si las horas no son suficientes con las frecuencias ajustadas en la iteración actual, se pasa a la siguiente, hasta que la suma de horas sea igual o menor que las disponibles.

El multiplicador que se usa en cada iteración para las distintas criticidades se calcula mediante la serie que aparece más abajo. Existe un parámetro k que es constante para una determinada criticidad del elemento. El valor de k es más pequeño cuanto mayor sea la criticidad del elemento.

$$S(n) = S(n-1) - \frac{S(n-1)k}{\log_2(\frac{S(0)}{S(n-1)+2})}$$

Ecuación 2.2. Serie de disminución de frecuencias

Una vez se termina el proceso se dispone de las tareas a realizar sobre los sistemas mantenibles de la infraestructura con sus frecuencias calculadas aplicando RCM y ajustando el esfuerzo a las horas de trabajo disponibles. Sin embargo, ésto no indica cuándo hay que realizar cada tarea. Para poder realizar las tareas encontradas en las frecuencias calculadas, hay que construir un plan de revisión.

2.3. Planificación de sistemas de tiempo real

Formalmente, la planificación de tareas de mantenimiento es un problema idéntico al de planificación de tareas en un sistema de tiempo real. Los dos problemas consisten en ejecutar tareas, asignando los recursos disponibles, de modo que se cumplan los plazos límite de todas ellas.

Es pues, natural, el fijarse en cómo se resuelve el problema en el ámbito de los sistemas de tiempo real. Si las similitudes son suficientes, se puede aplicar al problema actual el conocimiento existente en ese campo.

2.3.1. Tipos de esquemas de planificación para tiempo real

En un programa concurrente no es preciso determinar el orden concreto de ejecución de los procesos. Si se han diseñado bien los mecanismos de sincronización mediante zonas de exclusión mutua, el programa se comportará como se espera.

Cuando se trata de sistemas de tiempo real, no es suficiente con asegurar que el programa realiza la tarea que le corresponde. Es necesario, además, que realice dicha tarea en los plazos de tiempo establecidos. Por eso, es preciso predecir el orden de ejecución para confirmar que se pueden cumplir todas las restricciones temporales del sistema.

El método por el que se garantiza que se cumplen los plazos temporales del sistema es realizando una planificación de cómo va a ser la ejecución de los procesos. Sobre la planificación realizada se puede comprobar directamente si se van a cumplir los plazos o no.

La planificación de tareas en un sistema de tiempo real puede entrar dentro de dos categorías distintas según cuando se realiza la planificación de las tareas. Si la planificación se realiza antes de que empiecen a ejecutarse las tareas, se considera que la planificación es estática. Si la planificación se realiza según se van ejecutando las tareas, se considera que la planificación es dinámica. [7]

Cada una de las divisiones anteriores tiene dos variantes según si hay expulsión de tareas o no. La planificación es preemtiva si, una vez que una tarea ha comenzado a realizarse, puede verse interrumpida y sustituida por otra. Si, en cambio, una vez que una tarea comienza su ejecución debe terminar de ejecutarse para que otra puede tomar su lugar, se le llama no preemtiva.

A continuación, se comenta brevemente cómo se construye una planificación mediante algunos de los planificadores para sistemas de tiempo real más importantes. Los esquemas de planificación que se comentan son modelos teóricos, por lo que se supone que la carga adicional debida a los cambios de contexto y a los cálculos para determinar la siguiente tarea a ejecutar no se tienen en cuenta.

2.3.1.1. Planificación de prioridad estática

En este tipo de esquemas de planificación, cada proceso tiene asignado una prioridad. La prioridad se asigna antes de comenzar la planificación y permanece constante durante toda el período planificado. En los sistemas de tiempo real estricto la prioridad refleja las necesidades de temporización, no su importancia en el funcionamiento del sistema global.

- **Planificador de tasa monótona (Rate-monotonic).** Este planificador se usa cuando el tiempo límite de ejecución del proceso coincide con su período. En este esquema se asignan prioridades a los procesos según su período. Cuanto menor es el período, mayor es la prioridad.

  Esta estrategia se puede demostrar que es óptima, es decir, puede planificar cualquier problema que se pueda resolver por cualquier otro esquema de prioridades estático. Por otro lado, cualquier problema que cumpla la ecuación de abajo es planificable mediante esta técnica.

$$\sum_{i=1}^{n} \frac{C_i}{T_i} \leq n(2^{\frac{1}{n}} - 1)$$

Ecuación 2.3. Prueba de planificabilidad para tasa monótona.

Donde C_i es el tiempo máximo de ejecución de la tarea i, T_i es el período de la tarea i y n es el número de tareas.

La utilización máxima del recurso (en general el procesador) que garantiza la planificabilidad según la formula anterior disminuye asintóticamente hacia el 69,3% al aumentar el número de procesos.

Al aplicar el esquema de tasa monótona, se puede calcular el tiempo de respuesta para una tarea concreta en un instante crítico. El tiempo de retardo de una tarea en ese instante está influenciado por las tareas que tienen mayor prioridad que ella y no dependen de aquellas tareas cuya prioridad sea mayor que la suya. El modo de calcularlo es mediante una relación de recurrencia[9].

$$w_i^0 = C_i$$
$$w_i^{n+1} = C_i + \sum_{j \in h(i)} \lceil \frac{w_i^{n+1}}{T_i} \rceil C_j$$

Ecuación 2.4. Relación de recurrencia para tasa monótona.

Donde C_i es el tiempo máximo de ejecución de la tarea i, C_j es el tiempo máximo de ejecución de la tarea j, T_i es el período de la tarea i, n es el número de tareas y h(i) representa el conjunto de tareas con prioridad mayor que la tarea i.

Si se itera la relación anterior las veces necesarias ocurrirá que o bien el valor calculado para la w es mayor que el período o el valor calculado de la iteración n + 1 tiene el mismo valor que en la iteración n.

En el primer caso, cuando el retardo de la tarea supera el valor del período de esa tarea, el sistema no es planificable. Se está considerando que el tiempo límite de ejecución de la tarea coincide con su período. Si el retardo supera el tiempo límite, se puede asegurar que la tarea incumplirá su plazo por la interferencia de las tareas más prioritarias.

En el segundo caso, cuando se alcanza un punto estable, se ha calculado el retardo. Es el punto donde las tareas más prioritarias permiten que se ejecute la tarea considerada. Si este retardo es menor que el tiempo límite de ejecución de la tarea, se puede asegurar que la tarea cumplirá con sus plazos. La prueba consistente en calcular el retardo es necesaria y suficiente.

- **Planificador monótono en plazos limite (Deadline-monotonic).** Es una variación sobre el anterior esquema de planificación que se usa cuando los tiempos límite de ejecución de las tareas son menores o iguales al período de ejecución. En este caso, el planificador asigna prioridades según el plazo límite de las tareas. Cuanto más corto es el plazo límite, menor es la prioridad que se le asigna a la tarea.

  En este caso, el cálculo de los retardos sigue la misma fórmula que en el caso anterior. Sin embargo, para saber si el conjunto de tareas es planificable hay que comparar el retardo con el tiempo límite de ejecución de la tarea en vez de con el período de la misma.

2.3.1.2. Planificación de prioridad dinámica

Este tipo de esquemas de planificación, tienen la característica de que las tareas tienen asignadas prioridades que varían con el tiempo. Es decir, la prioridad depende del instante de tiempo en que se mida.

- **Planificador con menor tiempo de rodaja (Least slack time).** Este algoritmo selecciona, entre las tareas que están listas para ejecutar, aquellas que tengan un menor tiempo de rodaja, es decir, intenta ejecutar las tareas lo más tarde posible. Funciona mucho mejor con expulsión, llegando a alcanzar en este caso una utilización del 100%. Sin embargo, cuando se produce una sobrecarga transitoria, el comportamiento se degrada mucho, produciendo un efecto cascada que hace que los retrasos se encadenen. Es muy útil en sistemas donde hay una proporción considerable de procesos aperiódicos.

$$s = (d - t) - c'$$

Ecuación 2.5. Definición de tiempo de rodaja.

Donde d es el plazo límite de la tarea, t es el tiempo desde que comenzó el ciclo y c' es el tiempo de ejecución restante.

- **Primero el tiempo límite mas corto (Earliest Deadline First).** Este planificador determina cuál va a ser el siguiente proceso en ejecutarse en función del plazo máximo de ejecución. El proceso que tenga el tiempo límite más próximo, será el próximo en ejecutarse con independencia de cualquier otro parámetro. Este planificador dinámico es capaz de cumplir todos los tiempos límite siempre que la ocupación del procesador sea menor del 100%.

La prueba necesaria y suficiente de planificabilidad para este esquema es, simplemente, la que se muestra a continuación.

$$\sum_{i=1}^{n} \frac{C_i}{T_i} \leq 1$$

Ecuación 2.6. Prueba de planificabilidad para tiempo límite más corto.

Donde C_i es el tiempo máximo de ejecución de la tarea i, T_i es el período de la tarea i y n es el número de tareas.

- **Atropos.** El planificador Atropos esta basado en el planificador anterior, pero tiene control de admisión que permiten que se comporte mejor cuando la carga del sistema supera el 100%. Ha sido desarrollado por la Universidad de Cambridge.

2.3.2. Planificación factible

Para saber si un conjunto de tareas es planificable o no, se realiza una prueba de planificabilidad. Una prueba de planificabilidad puede requerir condiciones suficientes, condiciones necesarias o condiciones necesarias y suficientes [6] . La complejidad de una prueba de planificabilidad necesaria y suficiente es, en casi todos los casos, NP-Completo y por tanto nada práctico [8].

Se pueden usar condiciones necesarias o suficientes que sean computacionalmente ligeras para descartar o aceptar algunos casos. Por ejemplo, una condición necesaria, que es especialmente ligera, consiste en comprobar el uso de los recursos. Si se da el caso de que los recursos tienen una ocupación mayor que el 100%, las tareas no son planificables.

Una condición suficiente que indica que el problema es planificable, consiste en construir un ejecutivo cíclico que resuelva el problema. Se considera que es una condición suficiente por construcción.

La prueba por construcción tiene una ventaja. Al mismo tiempo que se obtiene la prueba de que la planificación es factible, se obtiene simultáneamente la planificación. También tiene la desventaja de que la prueba, obviamente, suele ser bastante pesada computacionalmente hablando.

La mayor diferencia entre el problema considerado y los sistemas de tiempo real radica en el uso que se realiza del tiempo. En los sistemas de tiempo real, el tiempo se considera homogéneo. Sin embargo, cuando se considera la planificación de tareas en un calendario real, aparece una heterogeneidad.

En efecto, no todas las horas son iguales entre si. En general, las jornadas laborales se reducen a determinadas horas del día y, en los casos que no es así, se prefieren las horas del día a las de la noche, ya que estas últimas son más caras. Si además se tienen en cuenta los días festivos, se incrementa la diferencia entre ambos.

Sin embargo, la prueba por construcción sigue siendo válida. Si se encuentra una planificación factible, el problema es planificable. Por tanto, si se encuentra un método para construir una planificación para las tareas sobre un calendario real, se puede discutir sobre la factibilidad del problema.

2.3.3. Planificación de tareas como búsqueda en un árbol

El proceso de encontrar una planificación factible por construcción consiste en ir añadiendo las tareas una a una a la planificación hasta que, o bien es imposible introducir ninguna tarea más o bien se han planificado todas las tareas.

Como se ha mencionado en un apartado anterior, hay múltiples algoritmos para realizar este proceso de forma guiada en los sistemas de tiempo real. Sin embargo, los principios en que se apoyan dichos algoritmos, no se adaptan bien a los calendarios reales. Por tanto, no hay, en principio, un orden predilecto que ofrezca más garantías de éxito que los demás. Es decir, hay que probar todas las posibilidades en que se se pueden ordenar las tareas.

Este proceso puede equipararse a la búsqueda dentro en un árbol. Se puede considerar un nodo raíz que contenga una planificación vacía. Se puede seleccionar cualquiera de las n tareas como primera tarea a planificar. De esa forma, del nodo raíz surgen n ramas correspondiente a las n posibles tareas que se han podido seleccionar.

De forma general, si en el nivel i hay i tareas planificadas, quedan n - i tareas que se pueden seleccionar para planificar a continuación. De cada nodo del nivel i cuelgan n - i nodos que representan cada una de las posibles elecciones de la siguiente tarea a planificar. En el nivel i + 1 se encuentran planificadas las i tareas del nodo considerado y, además, la tarea que se ha seleccionado entre las n - i disponibles[6].

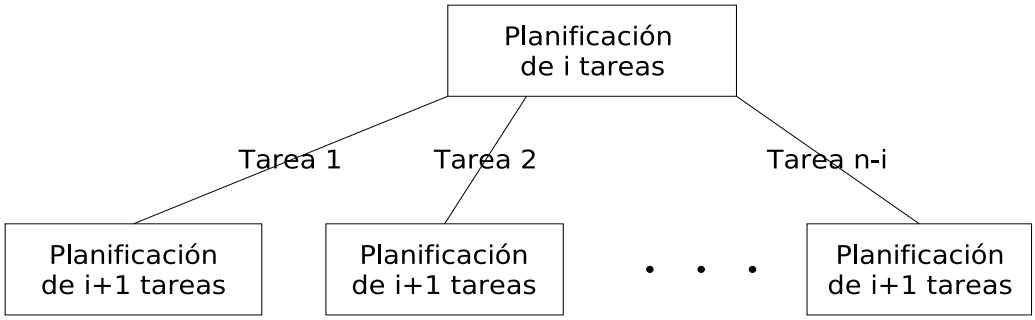

Figura 2.2. Construcción de la planificación mediante asignación de tareas

Es decir, el nodo raíz tendrá n nodos que cuelgan de él, uno por cada tarea que se planifica. Cada nivel tiene menos nodos dependientes que los que están por encima en el árbol. De tal forma que los nodos que son padres de los nodos hoja solo tienen un nodo dependiente.

Por construcción, la altura del árbol será, exactamente, el número de tareas y en los nodos hoja se encuentran las planificaciones posibles. Si existe una planificación factible se encuentra entre los nodos-hoja cuyo camino desde el nodo raíz tiene como longitud el número de tareas.

Hallar una planificación a partir del número de tareas es lo mismo que hallar una ruta entre la raíz y los nodos hoja tal que se cumpla esa restricción. Hay que tener en cuenta que el número de nodos que sean solución puede ser uno, muchos o ninguno, según lo restrictivo que sea el problema.

Dado que el problema, en el fondo, consiste en realizar una búsqueda en un árbol de decisiones, se pueden aplicar técnicas de inteligencia artificial para resolverlo. Una heurística adecuada puede disminuir el orden de la complejidad algorítmica de forma notable.

2.4. Técnicas de inteligencia artificial

La inteligencia artificial es una disciplina compleja en su definición y ambiciosa en su alcance. Se puede decir que tiene dos vertientes principales. Aun así, hay veces que se difumina la distinción. También hay ramas que se engloban dentro de la IA pero no pertenecen claramente a ninguno de los dos grupos.

Una de las vertientes que se engloban dentro del campo de la inteligencia artificial trata de emular el proceso de pensamiento humano. Esta vertiente se puede considerar la más científica de las dos. Su objetivo es producir maquinas que puedan pasar el test de Turing. Con cada avance, se aumenta la comprensión de cómo funciona realmente el cerebro humano. Dentro de este grupo se puede enmarcar la visión artificial y el aprendizaje automático, por ejemplo.

La otra vertiente trata de resolver problemas de forma heurística. Es una parte que se puede considerar más orientada a la ingeniería. Con este enfoque se intenta resolver problemas que requieren de sentido común o razonamiento abstracto mediante ordenadores.

Esta última modalidad se puede considerar, en general, como técnicas para evitar buscar a ciegas dentro del espacio de soluciones. Cuanta más información sobre el problema considerado o sobre las características de la solución se introduzcan en estos métodos, más rápidamente se obtendrán mejores soluciones.

El diseño de una heurística entraña dos peligros: en primer lugar, si la heurística no ofrece suficiente información sobre la idoneidad de una rama del árbol frente a otra, la búsqueda se vuelve aleatoria; en segundo lugar, si la heurística es excesivamente compleja no se obtiene beneficios sobre una búsqueda exhaustiva. Es decir, no se reduce la complejidad computacional.

2.4.1. Algoritmos de Simulated Annealing

La mecánica estadística es la rama de la física que se ocupa de la descripción microscópica de los fenómenos termodinámicos. La termodinámica clásica se puede considerar como el resultado del comportamiento medio de las moléculas que forman los cuerpos. Las propiedades de moléculas aisladas es objeto de estudio de la mecánica cuántica, pero si el número de moléculas es muy elevado (objetos macroscópicos), su comportamiento medio sigue los resultados hallados por la mecánica estadística.

Hallar una solución a un problema mediante la optimización de una función de coste, tiene grandes similitudes con los procesos que estudia la mecánica estadística[4]. Los distintos valores que pueden tomar las variables de las que depende la solución pueden recordar a los distintos estados de energía de las moléculas de un material. Se busca una solución óptima y la naturaleza tiende al estado de menor energía.

El calculo de una solución por medio de simulated annealing se inspira, en concreto, en el proceso industrial de recocido. Este proceso consiste en calentar un material hasta una temperatura adecuada al proceso que se desea realizar (varía según el objetivo a conseguir) y, posteriormente, dejarlo enfriar lentamente.

Mediante el proceso de recocido se eliminan los defectos y tensiones que se encuentran en los sólidos, sobre todo se usa con el acero. Al calentar el sólido, se aumenta la libertad de las moléculas, que pueden desplazarse ligeramente. De tal forma que, al enfriarse, adoptan una estructura cristalina más perfecta.

Durante este proceso las moléculas cambian de nivel energético mediante transiciones electrónicas. En la primera fase, durante el calentamiento del material, los electrones son promocionados a niveles de energía elevados. Cuando la temperatura del cuerpo disminuye, la configuración de los estados electrónicos de las moléculas del material van cambiando mediante transiciones entre niveles de energía.

El comportamiento de un conjunto de moléculas en equilibrio a una determinada temperatura viene determinado por la función de Boltzmann. Es una función estadística que determina la probabilidad del cambio de estado de energía de una molécula.

$$P(E_2 - E_1) = e^{\frac{E_1 - E_2}{k \cdot T}}$$

Ecuación 2.7. Ecuación de Boltzmann

Esta ecuación determina una cierta probabilidad al cambio de nivel energético desde el valor E_1 al valor E_2. La letra T representa la temperatura del sistema y k es la constante de Boltzman. Por tanto, la probabilidad del cambio de nivel energético depende de la diferencia de energía entre los niveles y de la temperatura.

Una de las propiedades de esta función consiste en que cuando el incremento es positivo, la probabilidad es mayor que uno. Lo que significa que, si la transición es a un nivel energético inferior, siempre ocurre. Cuando la transición es a un nivel energético superior, ocurre con cierta probabilidad. La probabilidad de que se dé la transición es mayor cuanto más alta sea la temperatura y disminuye con la misma.

Kirkpatrick y otros se dieron cuenta de que se podía aprovechar este mecanismo de optimización de la naturaleza [4]. El método de optimización (Simulated Annealing) que simula el proceso de recocido se adapta directamente del proceso físico.

Cuando en el proceso real se considera una configuración molecular, en el proceso simulado se refiere a una posible solución del problema. Cuando se realiza una transición entre niveles de energía, se esta considerando una posible solución distinta pero similar a la anterior. La transición sigue una cadena de Markov.

Cada solución candidata tiene asociado un coste que, en el proceso físico, sería la energía del sistema. Para calcularlo en el algoritmo, se crea una función de coste que, dada una posible solución, lo calcula de tal forma que represente la bondad de esa solución.

Donde en física aparece la temperatura del sistema, en computación se introduce un parámetro que parte de un cierto valor y va disminuyendo según avanza el proceso. Este parámetro condiciona la probabilidad de pasar de una configuración del sistema a otra de mayor energía.

El sistema simulado va recorriendo sucesivas cadenas de Markov. Cada cadena progresa mediante las transiciones de energía que dicta la ecuación de Boltzman. Al final del proceso, queda un estado del sistema que se presenta como la solución. El algoritmo completo por el que se busca la solución al problema queda representado en el diagrama que se muestra a continuación.

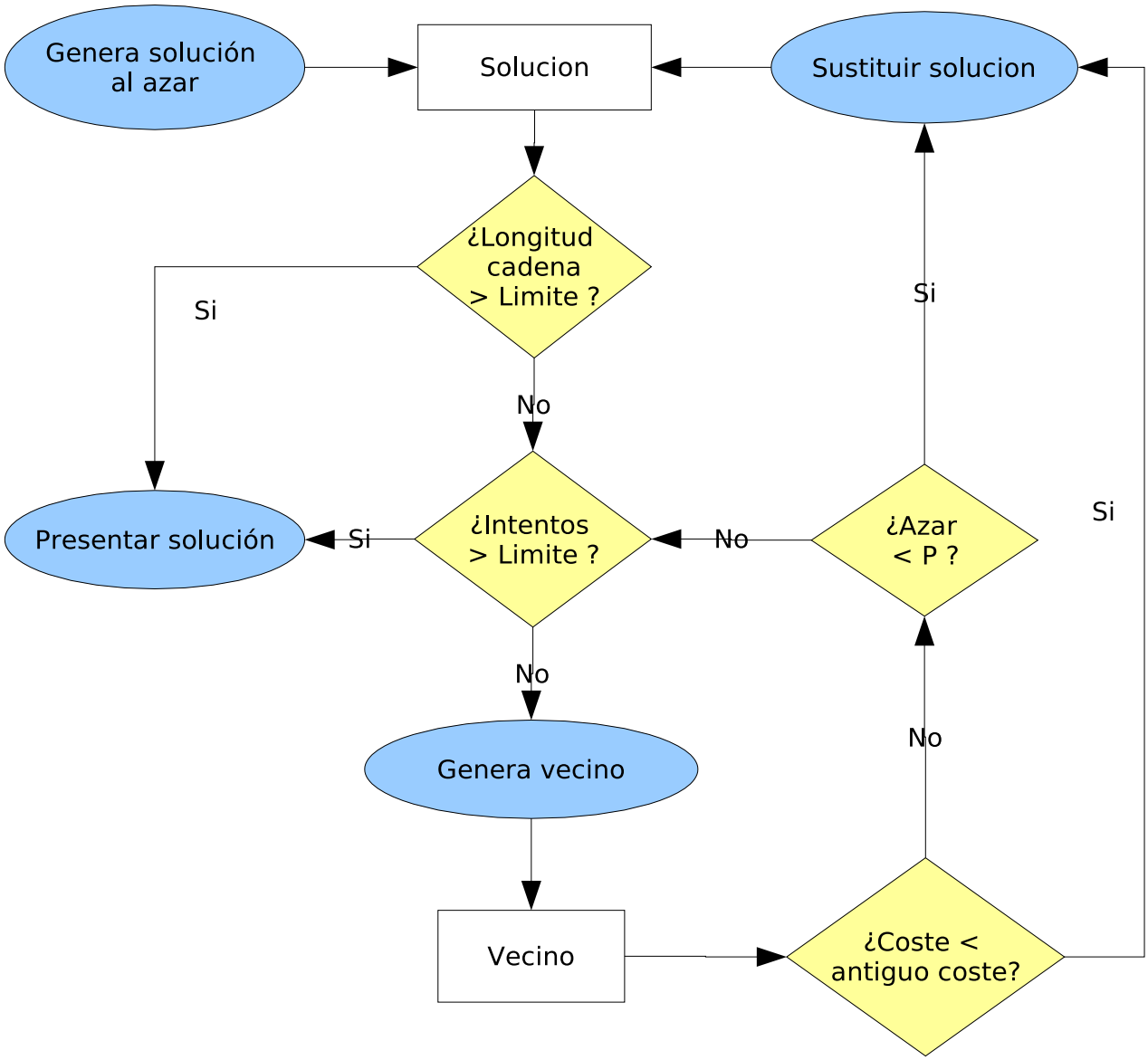

Figura 2.3. Proceso de Simulated Annealing

El origen de la cadena es una solución generada al azar. Para calcular el siguiente elemento de la cadena se parte del elemento actual. Se genera un vecino cercano y se evalúa la solución que representa. En caso de que el coste energético de ese vecino sea menor que el de la solución considerada, se adopta la nueva solución como el siguiente paso de la cadena. En el caso contrario, cuando el coste energético

de la nueva solución es mayor, se adapta como nuevo paso con la probabilidad de Boltzman. Si se rechaza la nueva solución se genera una nueva. El proceso continua hasta que se han recorrido un número determinado de cadenas sin que ninguna de ellas haya mejorado la solución anterior.

Capítulo 3. Estudio del problema

3.1. Descripción del problema

Se parte de un conjunto de tareas de revisión. Cada una de ellas se refiere a un sistema mantenible dentro de la infraestructura que hay que conservar. Estas tareas tienen un período máximo entre revisiones. Dichos plazos se deben cumplir para garantizar el estado óptimo de la infraestructura. Cada tarea de revisión tarda un tiempo determinado en llevarse a cabo, ese tiempo es conocido y se supone constante. Los sistemas pueden tener una criticidad distinta entre ellos aunque sean del mismo tipo. Cada sistema está ubicado en un determinado punto geográfico que se considera un punto kilométrico dentro de un tramo de vía.

El problema, en detalle, consiste en el modo de organizar en un calendario real las diversas revisiones de tal modo que se cumplan las siguientes restricciones:

- Se cumplan los plazos límite de las tareas.
- En un día no se necesiten más horas de las disponibles.
- Dentro de un mismo día las tareas sean tan próximas como sea posible.

El calendario en el que quieren incluir las tareas es un calendario laboral ordinario. Se puede considerar que consta de dos tipos de días: laborables y festivos. Los días laborables constan de un determinado número de horas-hombre disponibles de trabajo que se supone el mismo para todos ellos. Los festivos son días que carecen de horas de trabajo.

Hay que descontar de cada día laborable el tiempo necesario para los desplazamientos que, en realidad, no se emplea para realizar las tareas de mantenimiento. Tener en cuenta ese factor aumenta la complejidad de la planificación sin aportar mucho a la precisión de la solución.

Para solventar esta dificultad, se consideran varios supuestos. En un día de trabajo se viaja hasta el punto de la primera tarea del día, cuando se termina con ella se dirige a la siguiente y así hasta que se termina con la última y se vuelve al punto de partida. El plan del día se ordena de modo que no se retroceda. Dado que el plan obtenido tiende a juntar en el mismo día las tareas cercanas y separar en diferentes días las tareas más alejadas geográficamente, se puede considerar que las distancias entre tareas del mismo día son pequeñas. Por otro lado, la planificación se hace en tramos no muy largos (del orden de 40 Km) por lo que la ida y la vuelta se pueden realizar en menos de una hora en la mayoría de los casos. Así pues, se supone que la jornada laboral de cada miembro de la cuadrilla de mantenimiento tiene una hora menos de su duración teórica. De este modo, se puede evitar el aumento de complejidad del problema sin efectos especialmente adversos.

La duración de las tareas viene dada en horas-hombre. No se tiene en cuenta que una tarea sea de una naturaleza tal, que permita un número máximo de trabajadores simultáneamente. Así, una mayor cantidad de personas no supondrían una ayuda extra o, incluso, podría ser contraproducente. Se podría añadir este nivel de complejidad suponiendo que un día no dispone de ciertas horas-hombres, sino de un conjunto de hombres, cada uno con sus horas laborables. Además, la descripción de la tarea debería incorporar, aparte de su duración, el número máximo de personas que pueden participar en ella. No es necesario mencionar, que esto añade complejidad al problema, ya que multiplica el número de posibles soluciones.

3.2. Enfoques descartados

Se va a intentar una aproximación al problema de la planificación del mantenimiento desde la óptica de un problema de planificación de procesos en un sistema de tiempo real. Esto es, plantear el problema como la ubicación en una línea temporal de diferentes tareas periódicas de modo que se cumplan los períodos de las tareas.

Se han comentado anteriormente las similitudes entre este problema y los sistemas de tiempo real, pero también hay diferencias. La mayor discrepancia, como ya se ha mencionado, reside en que, en el mantenimiento, hay jornadas laborales mientras que en los sistemas de tiempo real el tiempo es contínuo.

Esta distinción se puede minimizar, ya que no eliminar, si se hace una equiparación lógica entre los períodos secundarios y las jornadas laborales. Los dos elementos pueden verse como contenedores de tareas con las siguientes propiedades:

- Pueden contener una o más tareas.
- Una ejecución de una tarea está en uno y solo uno de dichos contenedores.
- Un período principal contiene un número entero de contenedores.

Un plan de mantenimiento tiene en cuenta unas determinadas tareas que hay que realizar. Normalmente, una vez que se realiza el plan no se añaden ni se eliminan tareas. Por tanto, las técnicas dinámicas no son adecuadas para este caso.

En general, no tiene sentido que se interrumpan las tareas de mantenimiento. Si, por ejemplo, una tarea consiste en desmontar, limpiar y volver a montar un determinado sistema, no es razonable que un día se desmonte y limpie el sistema y montarlo de nuevo al día siguiente. Por ello, los métodos que utilizan expulsión no son los más apropiados.

Por tanto, los métodos de planificación estáticos sin expulsión son los más adecuados para el problema en cuestión. Sin embargo, ninguno funciona sin modificaciones debido al problema de la heterogeneidad del tiempo y la interpretación que se ha adoptado de los ciclos secundarios. Realizar modificaciones a estos métodos podría ser una opción válida. Sin embargo, la opción escogida es buscar una solución mediante una aproximación diferente.

3.3. Solución por fuerza bruta

La solución más evidente es realizar la composición del plan probando todas las combinaciones posibles que respeten los períodos. Si se considera el árbol mencionado en el capítulo anterior, se trataría de recorrer todo el árbol en busca de una solución idónea. La dificultad que se encuentra es que, como se ha mencionado en dicho capítulo, el problema de construir una planificación por fuerza bruta es NP-completo.

Sin embargo, de este planteamiento surge un concepto interesante: si en una propuesta se intercambian dos tareas con el mismo período y la misma duración, la planificación es idéntica en cuanto a planificabilidad. Es decir, si uno es planificable el otro también y si el primero no lo es, el segundo tampoco. Por tanto, no hay que probar todas las combinaciones, solo aquellas que no sean equivalentes.

La relación "tiene el mismo período y la misma duración" es una relación de equivalencia. La demostración consiste, sencillamente, en comprobar que cumple las tres propiedades requeridas: reflexiva, simétrica y transitiva.

$$\text{Sea:Tareas}(p,d): A(\text{pa},\text{da}) \equiv_{\text{pd}} B(\text{pb},\text{db}) \Leftrightarrow \text{pa} = \text{pb} \wedge \text{da} = \text{db}$$
$$\forall\, A(\text{pa},\text{da}), B(\text{pb},\text{db}), C(\text{pc},\text{dc}) \in \text{Tareas}$$

$$\text{reflexiva:}$$
$$\text{pa} = \text{pa} \wedge \text{da} = \text{da} \Rightarrow A \equiv_{\text{pd}} A$$

$$\text{simetrica:}$$
$$\text{Si} A \equiv_{\text{pd}} B \Rightarrow \text{pb} = \text{pa} \wedge \text{db} = \text{da} \Rightarrow B \equiv_{\text{pd}} A$$

$$\text{transitiva:}$$
$$\text{Si} A \equiv_{\text{pd}} B \wedge B \equiv_{\text{pd}} C \Rightarrow \text{pa} = \text{pb} = \text{pc} \wedge \text{da} = \text{db} = \text{dc} \Rightarrow \text{pa} = \text{pc} \wedge \text{da} = \text{dc} \Rightarrow A \equiv_{\text{pd}} C$$

Ecuación 3.1. Demostración de que la relación es de equivalencia

Como la relación "tiene el mismo período y la misma duración" es una relación de equivalencia, divide el espacio de tareas en clases de equivalencia. El número de clases de equivalencia en cualquier problema siempre sera menor o igual que el número de tareas, con lo cual el problema, generalmente, se simplifica.

Un problema cualquiera estará compuesto por n tareas. Los períodos de dichas tareas provienen de aplicar los cálculos RCM a un número limitado de elementos mantenibles. Por tanto, es razonable que el número de clases de equivalencia sea menor que el número de tareas en, prácticamente, todos los casos reales. Se espera que el rendimiento de esta técnica esté relacionada con el número de clases de equivalencia y con el número de tareas.

3.4. Armonización de períodos

Cuando se trata de sistemas de tiempo real, supone una gran ventaja el hecho de que los períodos sean armónicos. En ese caso, el problema es mucho más sencillo.

Si se consideran ahora las tareas de mantenimiento sobre jornadas laborales limitadas en calendarios reales, se ve que parte de las ventajas proporcionadas sobre períodos armónicos desaparecen. Una concentración inoportuna de días de fiesta puede descabalar una distribución creada con técnicas para tareas armónicas.

Sin embargo, el efecto de la armonización de períodos sobre las clases de equivalencia es notable. Después de efectuar los cálculos RCM, los períodos de las tareas pueden presentar mucha dispersión. Esto provoca que las clases de equivalencia sean numerosas, pero si se realiza un proceso de armonización se consigue reducir el número de clases de equivalencia a unos pocos conjuntos representativos.

Los períodos asignados a las tareas después de armonizar deben cumplir tres condiciones:

- **Deben ser armónicos.** En caso contrario, carece de sentido haber realizado el proceso de armonización. Se podría relajar esta condición para que agrupe tareas similares bajo una misma clase de equivalencia. Para ello, habría que dar dos pasos que pueden estar relacionados: encontrar el número de clases que se pretende conseguir y determinar en qué clase entra cada tarea. Las técnicas de agrupamiento, como por ejemplo k-medias, pueden ser adecuadas para este fin. Una posible ampliación podría consistir en estudiar el comportamiento de estos tipos de enfoques en la solución.

- **El nuevo problema debe consumir como mucho los recursos disponibles.** Si no es así, la planificación es imposible y no sirve de nada que los períodos estén armonizados. Hay que tener en cuenta que se supone que las tareas originales están definidas de modo que el uso de los recursos está bien aprovechado [2]. Por tanto, este factor es significativamente limitante ya que el margen de maniobra es pequeño. Por otro lado, no se aceptan aquellas soluciones que apuren demasiado los recursos, ya que es necesario tener holgura para evitar problemas en fases posteriores.

- **La variación de los períodos debe ser pequeña y tender a acortar los períodos más que a alargarlos.** Cuanto más crítico sea un sistema, mayor debe de ser la resistencia que presenta a que se alarguen sus períodos. Este efecto no debe ser muy pronunciado ya que la criticidad es, en parte, responsable del período que presenta el sistema en un principio.

Estas condiciones no garantizan que el nuevo problema sea planificable, pero sí aumentan en gran medida las posibilidades de éxito. Si el problema original era planificable, el problema modificado muy probablemente lo sea. En el caso de que el problema original no fuera planificable, el nuevo problema tiene posibilidades de serlo. Hay que tener cuidado en este punto, ya que si se permiten fuertes modificaciones del problema original se pueden conseguir problemas planificables

a partir de algunos que no lo eran, pero la relación entre ambos problemas puede ser ténue. La solución tomada es encontrar el problema más similar que cumpla las condiciones y dejar en manos del hipotético usuario evaluar si el problema es lo bastante similar al original o no. Para ello, se proporciona una medida cuantitativa de la diferencia y la descripción de cada tarea antes y después.

3.5. Uso de algoritmos de recocido

Si se considera la descripción del problema en forma de un árbol, la solución se puede considerar desde el punto de vista de una búsqueda del nodo-hoja idóneo. En realidad, no tiene por qué haber un único nodo-hoja idóneo. Hay dos tipos de nodo-hoja, aquellos que consiguen una planificación y aquellos que no. Dentro de aquellos que son planificables se diferencian por las distancias geográficas entre las ubicaciones de las tareas que se realizan en el mismo día.

La búsqueda dentro de un árbol de soluciones es un problema muy conocido en el que las técnicas de inteligencia artificial se desenvuelven muy bien. Se podrían usar numerosas técnicas distintas: A*, búsqueda en haz, etc. Sin embargo, se ha optado por utilizar búsquedas locales iterativas. En concreto, se ha trabajado con algoritmos de recocido.

El recocido es una técnica computacional que imita el proceso físico por el cual, durante el proceso de solidificación de un material, éste cae en un mínimo de energía. De la misma forma, con una heurística que simule el nivel de energía se consigue evitar los mínimos del sistema debido al selectivo uso del azar que realiza el algoritmo.

Capítulo 4. Uso de Backtracking para planificar las tareas

4.1. Descripción de la técnica

En esta aproximación al problema se parte de un conjunto de tareas iniciales. El conjunto contiene las tareas que se van a planificar en el tramo considerado. Cada una de dichas tareas tiene, aparte de un identificador unívoco, estos datos:

- **Duración.** Es el trabajo que hay que realizar para completar la tarea expresado en horas-hombre.
- **Período.** Representa el intervalo de tiempo entre distintas ejecuciones de la tarea.
- **Criticidad.** Una medida de la importancia de la tarea desde el punto de vista de las consecuencias del fallo. Se usan cuatro categorías etiquetadas desde la A (la más crítica) a la D (la menos crítica).
- **Localización.** La localización se refiere al punto del tramo donde se encuentra el sistema a revisar. Cada conjunto de tareas está referida a un tramo de vía, por lo que, indicando el punto kilométrico, queda ubicado el punto de la revisión.

Como se ha explicado anteriormente, dos planificaciones que intercambian tareas equivalentes entre si son, esencialmente, la misma planificación. Se utiliza la partición del espacio de soluciones que realizan las clases de equivalencia definidas por la relación para evitar esa redundancia. Para cada problema concreto se definen las clases de equivalencia correspondientes. Cada tarea del problema inicial se asocia con una y solo una clase de equivalencia.

Para realizar la planificación se realiza backtracking sobre las clases de equivalencia. Es decir, se van añadiendo tareas a la planificación de uno en uno. Mientras la planificación sea factible y queden tareas por planificar, se siguen añadiendo tareas. Si al añadir una nueva tarea no se consigue que sea factible se elimina la anterior y se prueba con otra perteneciente a otra clase de equivalencia. La búsqueda se detiene cuando se encuentra una solución o se han probado todas las combinaciones de clases de equivalencia.

En el diagrama de más abajo, se ve con más detalle el proceso que realiza el algoritmo para seleccionar la tarea que se va a añadir a la planificación en cada momento. La selección se realiza según la clase de equivalencia a la que pertenece. Cuando se selecciona una clase de equivalencia solo se puede escoger entre aquellas que aún tengan alguna tarea asociada sin asignar. Dentro de una clase de equivalencia es recomendable que las tareas estén ordenadas por algún criterio concreto.

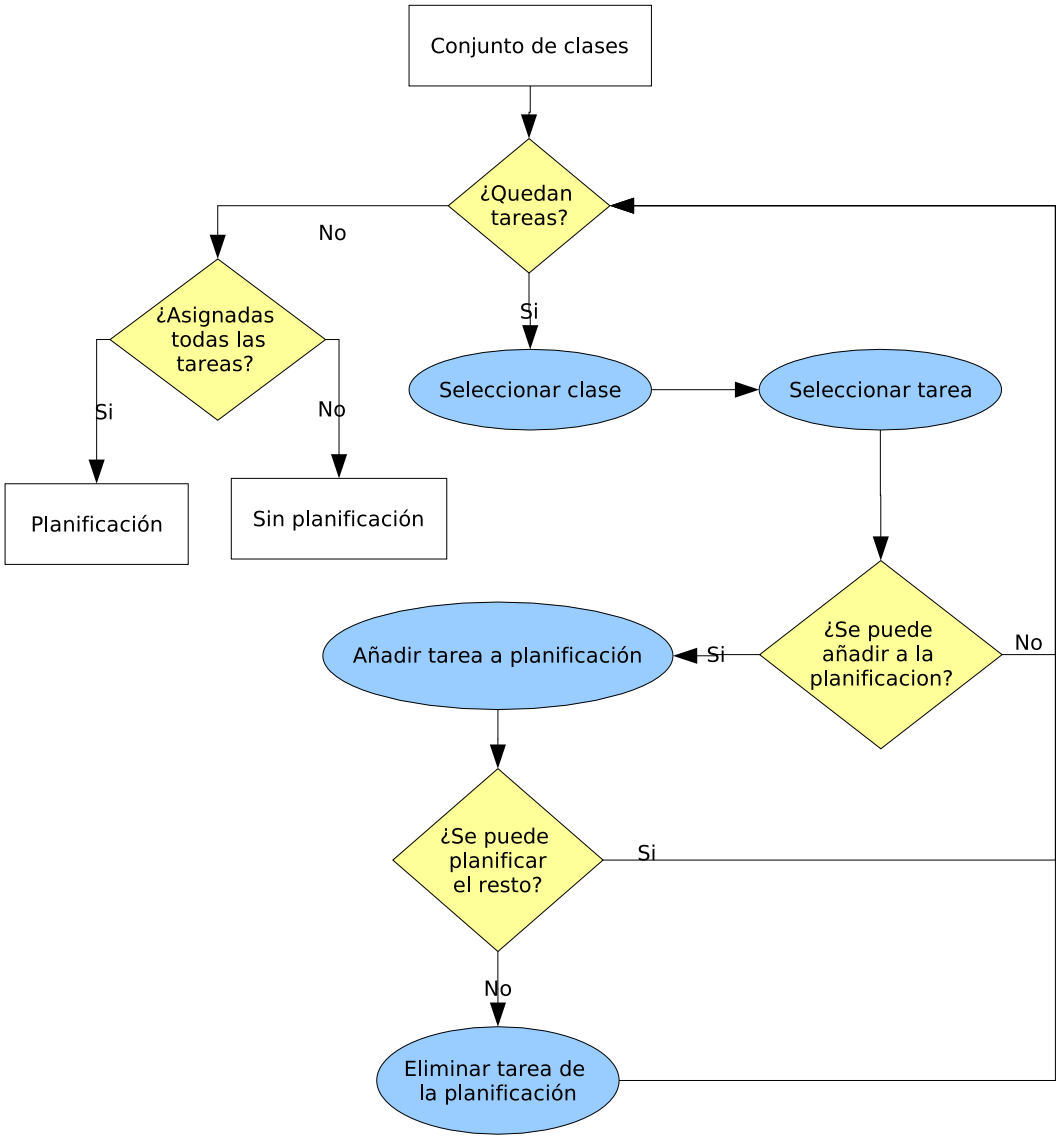

Figura 4.1. Algoritmo por backtracking.

La asignación de tareas a días concretos se realiza simultáneamente a lo largo del período total a planificar. El período total a planificar se divide en períodos del tamaño que tiene el período de la tarea. Si el período total no es múltiplo exacto del período de la tarea, el conjunto de días entre el ultimo período completo de ejecución de la tarea y el final del período a planificar se trata de forma especial.

En cada uno de los períodos de ejecución de la tarea se busca un día con el suficiente tiempo restante como para acomodarla. La tarea se asigna a dicho día y se descuenta el tiempo empleado del tiempo productivo restante de dicho día.

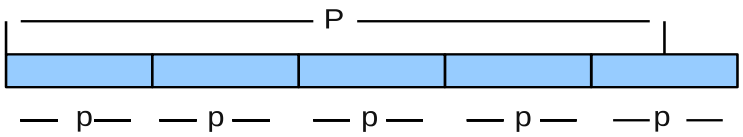

Figura 4.2. Partición del período total

Si el final del período a planificar (La distancia marcada P) no coincide con el final del período de la tarea (marcado con las distintas p), se permite que en ese último período no se lleve a cabo la tarea. Salvo esta excepción, es necesario que cada período de la tarea tenga una ejecución de la tarea para que se considere que la tarea ha sido planificada.

El día de ejecución de una tarea dentro del período de ejecución impacta en la bondad del algoritmo. Para seleccionar el día más adecuado, se utilizan los conceptos de próximo y lejano que se han definido anteriormente. El día para la ejecución de una tarea se escoge aplicando las reglas que se detallan a continuación y en el orden en que aparecen.

- El primer día cuyas tareas estén físicamente cercanas (localización próxima) a la tarea actual.
- Si no cumple el criterio anterior, se escoge el primer día sin otras tareas.
- Si no se dan los criterios anteriores, se selecciona el primer día que no tenga tareas en puntos kilométricos muy alejados de la tarea considerada.
- Si los criterios anteriores no son satisfechos, se asigna la tarea al primer día con suficiente tiempo como para aceptarla.

La cuantificación de *cerca* y *lejos* se realiza como un porcentaje de la distancia máxima entre tareas a realizar. Se ha descubierto que, si cerca se considera que es el 10% de esta distancia máxima y lejos aquellos que se encuentren a más del 60% de esta distancia, los resultados ofrecen distancias entre tareas sucesivas lo bastante pequeñas como para que no supongan un tiempo extra considerable el desplazarse de una a la siguiente.

Si no se consigue una planificación mediante este método, se puede permitir que alguna de las tareas menos prioritarias se salten algún plazo. De ese modo, en determinadas fechas con gran concentración de festivos se puede sacrificar alguna revisión de un sistema poco prioritario a cambio de tener una planificación.

Pseudocódigo del método por backtracking.

```
procedimiento planificar( numeroColocados )
Si numeroColocados = numeroTareas
devuelve éxito
si no
para cada clase de equivalencia
para cada tarea de la clase
si la tarea no esta asignada
actual:= tarea
fin si
fin para
si actual se puede asignar a la planificación
se asigna actual a la planificación
encontrado := llamada a planificar( numeroColocados + 1 )
si encontrado
devuelve éxito
si no
desasigna actual de la planificación
fin si
fin si
fin para
devuelve fracaso
fin si
```

```
fin procedimiento
```

4.2. Detalles de la implementación

Se ha realizado una implementación del algoritmo explicado anteriormente para realizar las pruebas y mediciones. Al realizar la implementación se han tomado ciertas decisiones que pueden afectar al rendimiento.

La implementación realizada es recursiva. Una implementación iterativa podría tener un perfil de eficiencia diferente.

Tal y como está diseñado el algoritmo, hay una gran diferencia entre las ejecuciones que finalizan con una planificación factible y las que no encuentran una planificación. Cuando se encuentra una planificación, se detiene la búsqueda, mientras que para tener la seguridad de que no existe una planificación factible hay que agotar el espacio de búsqueda. En la implementación esta divergencia entre las dos posibilidades se mantiene.

Cuando una planificación está en un estado tal que no se puede asignar ninguna de las tareas restantes, se vuelve atrás. De este modo, si un problema no tiene una planificación factible, el tiempo que se tarda en dar una respuesta depende, en parte, de en que lugar del árbol estén los puntos de vuelta atrás. Las clases de equivalencia se han ordenado por su nivel de exigencia, de modo tal que las tareas más exigentes se intenten asignar primero.

El control de cómo se organizan las tareas según el punto kilométrico donde se realicen, queda en manos de la parte del programa que asigna una tarea a los días en que se efectúa. Por tanto, no interviene directamente en el algoritmo de búsqueda de la planificación. Aun así, dentro de cada clase de equivalencia se ordenan las tareas por punto kilométrico. De ese modo, dos problemas idénticos se ejecutan siempre de la misma manera independientemente del orden en que estén las tareas que lo componen.

4.3. Resultados

El algoritmo explicado anteriormente se ha ejecutado sobre varios conjuntos de tareas. Se va a discutir los resultados obtenidos.

El período a planificar en estas pruebas se ha fijado a un valor concreto, en este caso es un año (365 días entre laborales y festivos).

Las horas consideradas son las horas laborales reales, es decir, se ha descontado previamente el tiempo invertido en desplazamientos, como se mencionó anteriormente. La duración de la tarea más corta en estas pruebas es de media hora, por tanto no se consideran fracciones de tiempo inferiores a media hora.

Para cada conjunto se selecciona una cantidad de horas diarias disponibles lo bastante pequeña como para que sea necesario probar varias combinaciones para poder determinar si existe una solución o no.

El algoritmo no explora las ramas que ya ha descartado, por lo que diferentes problemas con el mismo número de tareas, de clases de equivalencia y de horas-hombre diarias pueden tener diferentes tiempos de ejecución según la distribución de las tareas en las distintas clases de equivalencia. Aunque el mismo conjunto de tareas tardará siempre lo mismo con independencia del orden en el que estén definidas.

4.3.1. Influencia del número de tareas

Para comprobar la influencia del número de tareas a planificar, se realizan cinco ejecuciones de diferentes problemas manteniendo fijo el número de clases de equivalencia existentes, en concreto, tres clases. En cada fila se indica el número de tareas, el tiempo medio de ejecución y la desviación media de la medida.

Primero se realizan mediciones para problemas para los que se encuentra una solución factible. Los resultados obtenidos se reflejan en la tabla que aparece a continuación.

Tabla 4.1. Variación con el número de tareas (éxito)

Número de tareas	Tiempo	Desviación
6	0,06	0,02
8	0,25	0,06
10	0,4	0,2
12	0,6	0,5
14	2	1,8

Si se representa de forma gráfica estos datos, se puede apreciar una cierta correlación entre el número de tareas y el tiempo medio de ejecución. La desviación típica es considerablemente alta, lo que no es de extrañar, pues el tiempo que tarda depende de en que punto de los casos considerados encuentre la solución.

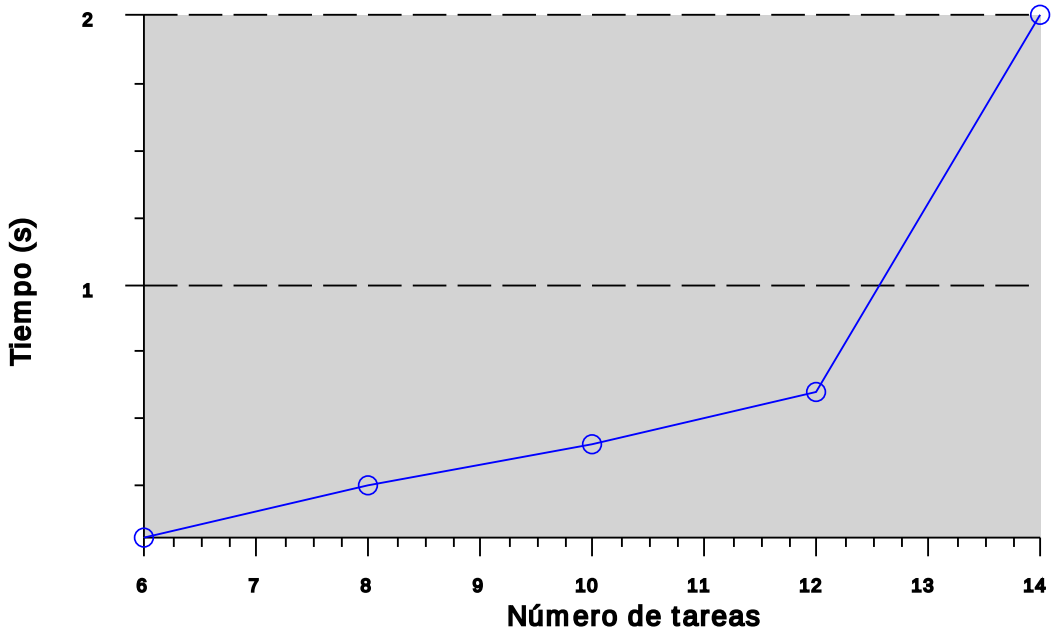

Figura 4.3. Variación con el número de tareas (éxito)

En la gráfica anterior, la parte izquierda se asemeja a un crecimiento lineal y la parte derecha recuerda a un crecimiento exponencial. A continuación, se muestran los mismos datos, pero en escala logarítmica. Se aprecia que hay un crecimiento constante, pero no se ajusta bien ni a una función lineal ni a una exponencial.

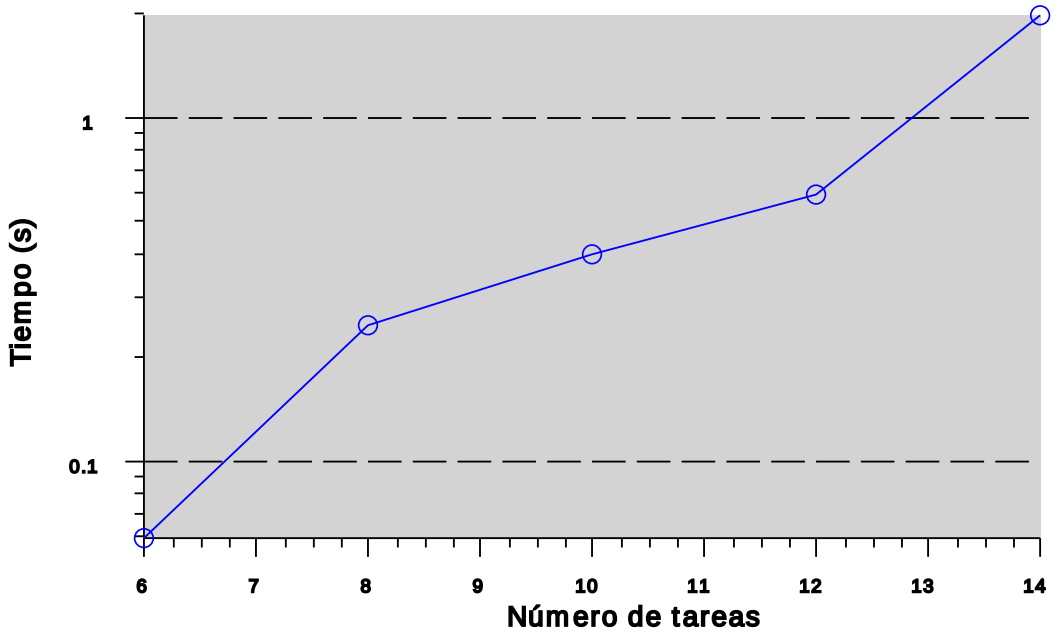

Figura 4.4. Variación con el número de tareas (éxito)

Las ejecuciones anteriores se detienen cuando encuentran una solución. Pero el peor caso, aquel que tarda más en terminar el algoritmo, es cuando no se encuentra una solución. El proceso de búsqueda es igual al caso anterior, excepto porque, en este caso, no se detiene hasta que ha agotado el espacio de búsqueda. De ahí proviene la enorme diferencia de tiempo entre los casos en los que se encuentra la solución con respecto a aquellos en los que no. Además, como el algoritmo termina de explorar todas las posibilidades, las desviaciones típicas de las medidas no son tan grandes, relativamente, como en el caso anterior.

En la tabla siguiente se muestran los resultados obtenidos en casos análogos al anterior, pero en los que no se encuentra ninguna solución factible.

Tabla 4.2. Variación con el número de tareas (fallo)

Número de tareas	Tiempo	Desviación
6	0,23	0,05
8	0,9	0,3
10	3	2
12	20	10
14	100	70

Se puede observar que hay una correlación entre el tiempo medio de resolución de un problema y el número de tareas de que consta dicho problema. A continuación se representan estos datos de forma gráfica.

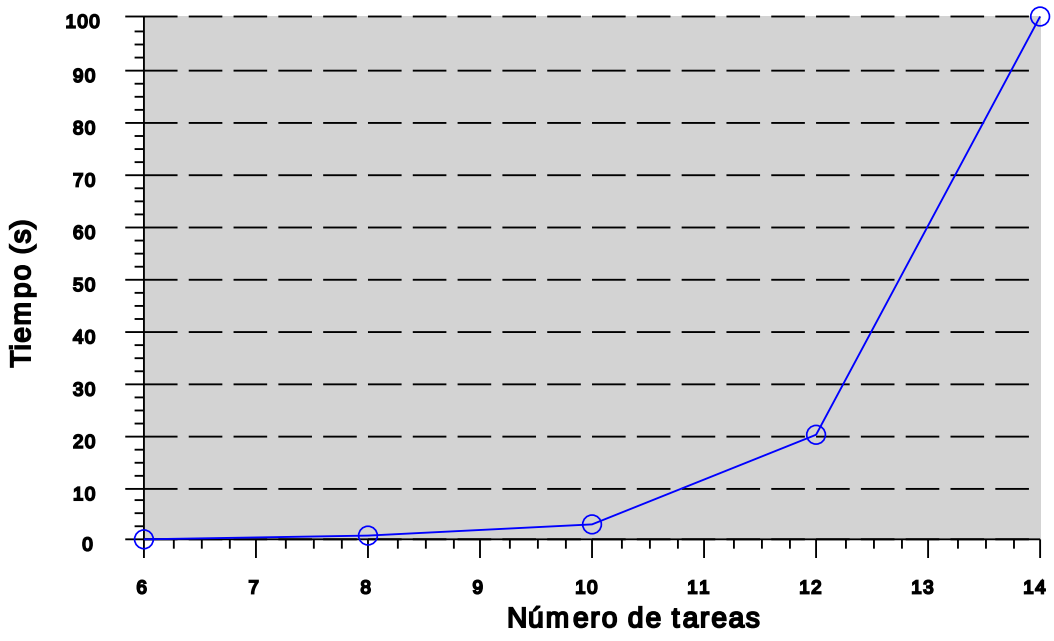

Figura 4.5. Variación con el número de tareas (fallo)

Si se representa en escala logarítmica se ve más claramente el comportamiento. La tendencia es claramente exponencial.

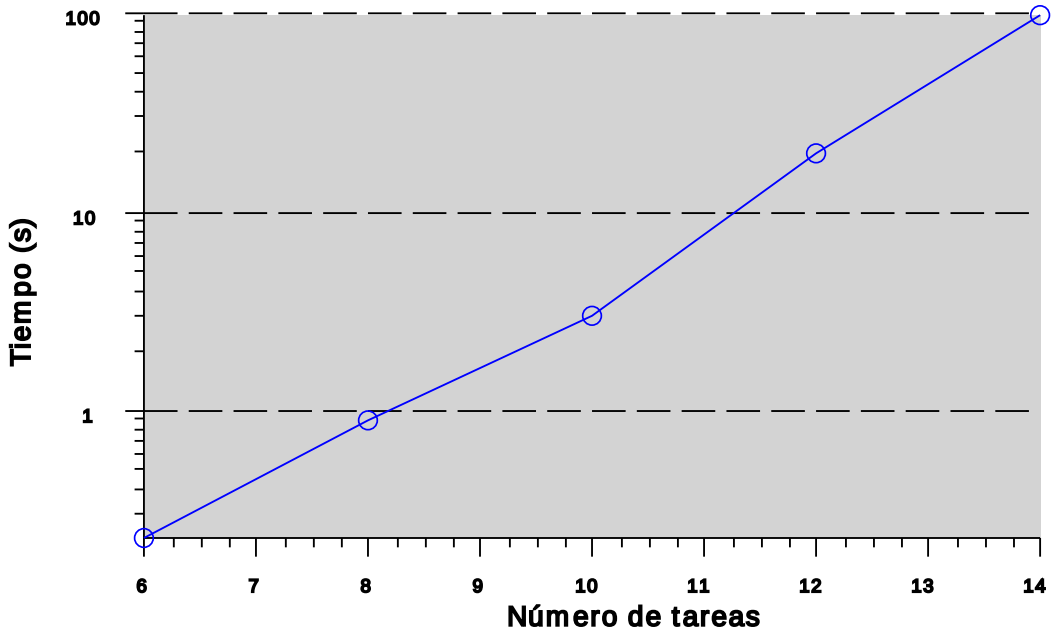

Figura 4.6. Variación con el número de tareas (fallo)

Para la serie en la que el algoritmo encuentra solución no se ve una tendencia clara, pero para los casos en los que se agota el espacio de búsqueda, el comportamiento es de crecimiento exponencial.

A continuación, se muestra una comparación gráfica del comportamiento cuando encuentra una solución en contraste con aquellos casos en los que no se haya una solución factible.

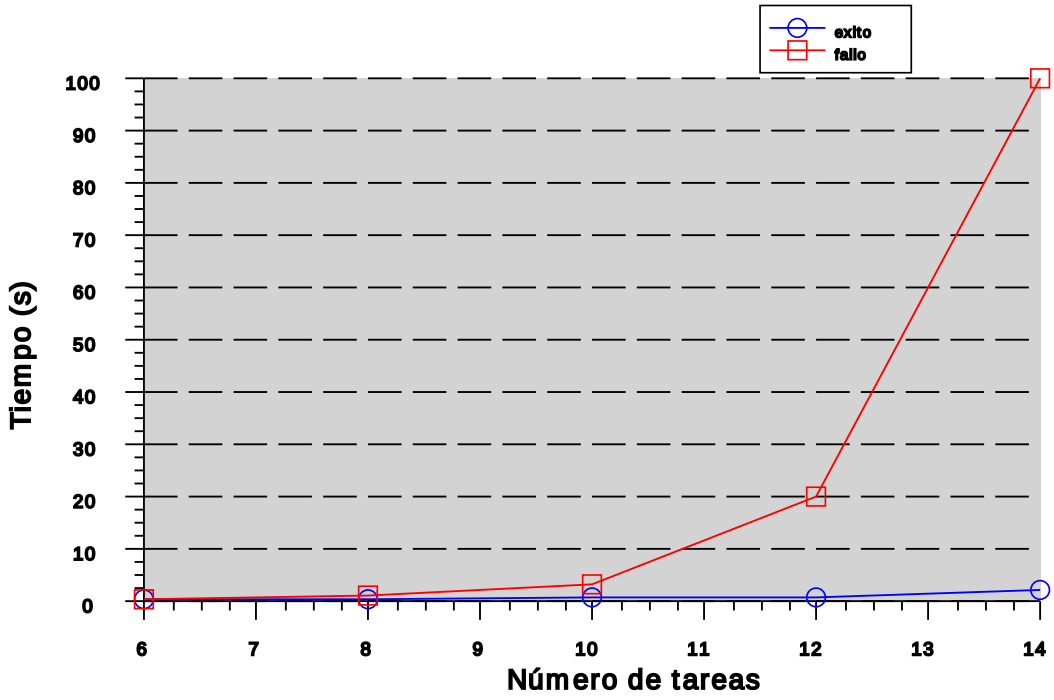

Figura 4.7. Variación con el número de tareas (comparación)

4.3.2. Influencia del número de clases de equivalencia

Ahora se va a comprobar cómo afecta la variación del número de clases de equivalencia manteniendo constante el resto de factores. Se construyen diferentes problemas con ocho tareas cada uno. Se resuelven cinco problemas para cada valor del número de clases y se muestra el tiempo medio y la desviación típica.

Tabla 4.3. Variación con el número de clases de equivalencia

Número de clases	Tiempo	Desviación
3	1	0,5

Número de clases	Tiempo	Desviación
4	2	1
5	5	1
6	11	4
7	27	7
8	43	15

Se puede comprobar en la gráfica siguiente que el crecimiento es también exponencial. Es decir, que dados dos problemas cualesquiera con el mismo número de tareas, el que tenga un número de clases de equivalencia menor es mucho más sencillo de resolver.

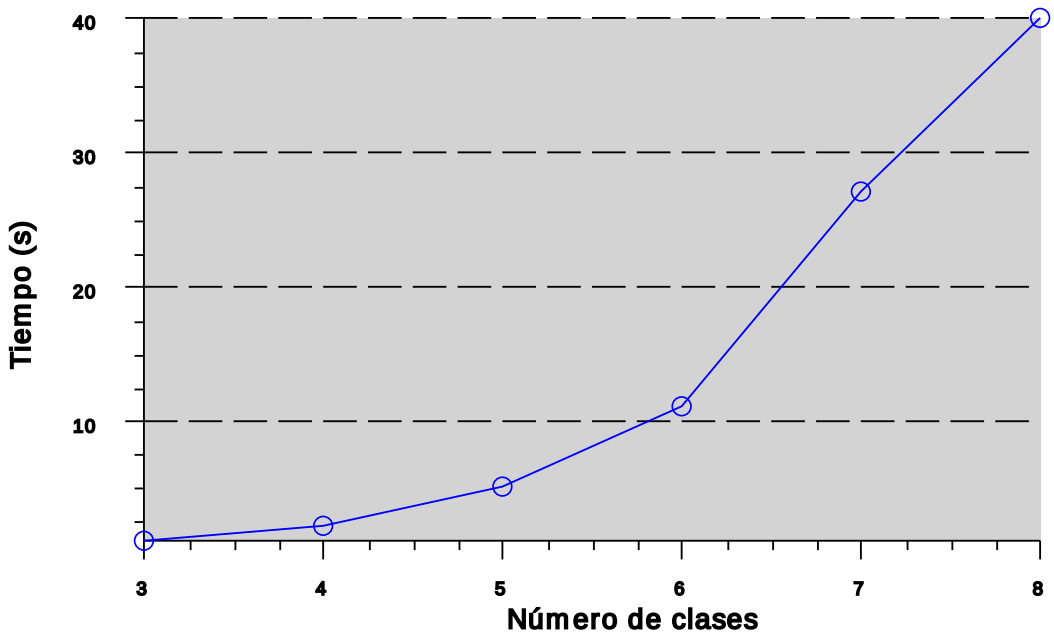

Figura 4.8. Variación con el número de clases

Si se representa en escala logarítmica se ve más claramente el comportamiento.

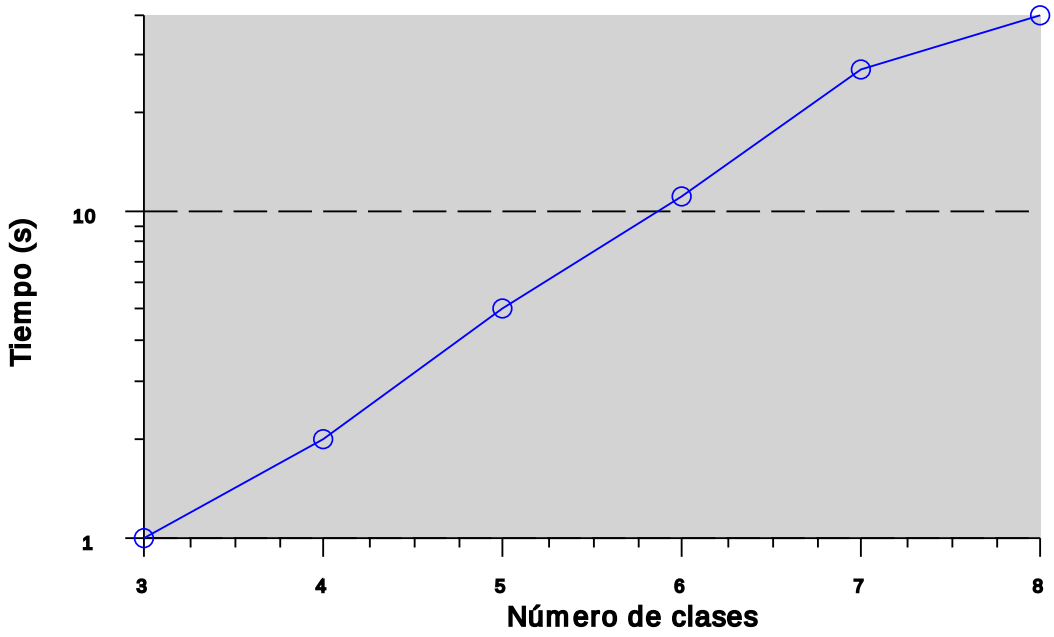

Figura 4.9. Variación con el número de clases (logarítmico)

4.4. Ejemplo de ejecución en un caso real

Se va a considerar el comportamiento real del algoritmo con un ejemplo real. Se va a considerar el caso en que se quiera planificar las tareas de mantenimiento del trayecto Villalba de Guadarrama - Cercedilla.

Como se ha visto antes, el rendimiento del algoritmo depende de varios factores. Entre los factores se encuentra el hecho de que exista una solución factible o no. Se va a considerar el caso en que existe la planificación, ya que es el caso deseable.

Este trayecto consta de 62 tareas de mantenimiento repartidas como se aprecia en la siguiente tabla.

Tabla 4.4. Tareas del trayecto

Número de tareas	Criticidad
8	A
44	B
6	C
4	D

Se ejecuta la planificación con 3 horas laborables diarias y el calendario laboral del año 2006. El factor de utilización de los recursos obtenido con estos datos es del 91,9%. El sistema encuentra una solución factible en 15400 segundos aproximadamente, lo que equivale a cerca de cuatro horas y media

4.5. Evaluación de la solución

A la vista de los resultados obtenidos en los apartados anteriores, se puede decir que esta aproximación al problema tiene ciertas desventajas.

El algoritmo usado tiene complejidad computacional exponencial. Por tanto, el método no escala bien. El tiempo de cómputo, con el número de instrucciones por segundo constante, crece de forma exponencial.

Cuando termina el período planificado, no se puede comenzar de nuevo por el día 1 de la planificación anterior. Hay dos motivos principales por lo que ocurre esto. En primer lugar, la planificación tiene en cuenta los días que son laborables y los que no. Aparte de este motivo, hay problemas con los plazos de ejecución de las tareas debido a efectos de borde hacia el final del plazo planificado.

Tal y como se ha definido el algoritmo, se garantiza que se cumplen los plazos de todos los períodos desde la primera ejecución de la tarea hasta la penúltima. Como las tareas no tienen por qué ser armónicas entre si, puede haber tareas cuyo período no sea divisor entero del período a planificar. Es decir, pueden existir tareas para las cuales el período a planificar no sea un número entero de veces la duración de su período.

En el período de tiempo comprendido desde el último día múltiplo del período de la tarea (instante de tiempo C en el dibujo) y el final del período a planificar (instante D del dibujo) no se asegura que exista una ejecución de la tarea. Si la tarea se ejecuta al final del primer período dentro de la repetición del período planificado (representado por la E del dibujo), el período total entre ejecuciones es la distancia entre C y E que es mayor que el período de la tarea que es la distancia entre D y E.

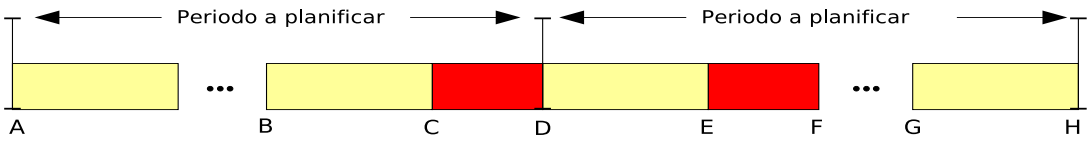

Figura 4.10. Ejecuciones sucesivas de la planificación obtenida

Para paliar este problema se puede realizar un ajuste. Si el período de tiempo en el que se desea planificar es T, se puede contar con la posibilidad de ampliar el período en el que existe una programación. Para ello se planifica para un tiempo T+P, donde P es el período de la tarea que tenga el período mayor. Si cuando se cumple T, se sustituye la planificación por una nueva, los plazos incumplidos serán menos numerosos que si se espera a T+P.

Este método tiene la ventaja de que es determinista y exhaustivo. Se tiene la certeza de que, si existe la solución, tarde o temprano el algoritmo dará con ella. Además, para el mismo conjunto de tareas, el tiempo que tardará en encontrar la solución o indicar que la solución no existe será la misma (siempre que no se modifique el hardware).

Recapitulando, las principales características de este método son: problemas de escala, mal ajuste entre planificaciones y determinismo. Ésto refleja claramente los entornos en los que es apropiado, entornos donde se quiera mantener un conjunto estable y conocido de tareas durante un tiempo limitado y también conocido. Si es más importante saber si hay una planificación factible o no que el hecho de encontrarla, este método es más deseable que otros no deterministas.

El tiempo que tarda para relativamente pocas tareas es elevado, pero si se está planificando las tareas de todo un año no es descabellado que los tiempos de cómputo se eleven a uno o más días. La capacidad del hardware influye mucho en el tiempo de cálculo y es fácilmente paralelizable. Hay que tener en cuenta que si se considera el problema como una búsqueda en árbol, la paralelización es prácticamente trivial enviando distintas ramas del árbol a distintas unidades de proceso.

Capítulo 5. Armonización de períodos

5.1. Descripción de la técnica

Pasar de un determinado problema a otro donde los períodos sean armónicos proporciona varias ventajas en los sistemas de tiempo real. Por ejemplo, una vez realizada la planificación para el período principal se puede mantener para siempre. Sin embargo, también tiene sus desventajas. El nuevo problema, aunque sea muy parecido, no es el problema original. Por otro lado, encontrar el problema armónico lleva un coste computacional añadido que, en principio, puede ser NP-completo.

Cuando se trata de adoptar las tareas a un calendario real las cosas cambian. Una vez realizada la planificación sobre un período principal no se puede suponer que esa misma planificación sea valida para el intervalo de tiempo siguiente. Hay fiestas que cambian de ubicación cada año como, por ejemplo, la Semana Santa. Además, si el período principal no es múltiplo de siete los fines de semana pierden sincronía.

En este caso, la ventaja radica en que el problema se simplifica. La dispersión de períodos cercanos pero diferentes del conjunto original se concentran en un menor número de clases de equivalencia. En el capitulo anterior ya se discutió el efecto de la reducción del número de clases de equivalencia en el rendimiento del algoritmo.

Por otro lado, si los períodos de las tareas que forman el problema son armónicos, se puede asegurar que existen instantes críticos. Hay momentos en los que todas las tareas se encuentran al final de su ciclo. Este punto tiene importancia a la hora de extender la duración de la planificación.

Los períodos de partida no son realmente arbitrarios ya que surgen a partir de aplicar determinadas transformaciones a las frecuencias base de revisión[2]. Sin embargo, no se pueden realizar suposiciones sobre los períodos resultantes, ya que dependen, en gran medida, de las horas de trabajo disponibles, el número de tareas y su criticidad.

5.1.1. Perturbaciones en los períodos

Una opción para encontrar un conjunto de períodos armónicos podría consistir en introducir pequeñas perturbaciones en los períodos originales y comprobar si resultaban armónicos o no. Esta linea de trabajo se puede considerar como fuerza bruta y, por tanto, cualquier otro sistema razonable resulta más eficiente.

Si se invierte el razonamiento anterior, se encuentra un método más efectivo. Se puede fijar un valor concreto para el mayor de los períodos (que debe coincidir con el período principal, ya que se fuerza que los períodos sean armónicos). En ese caso, los valores permitidos del resto de los períodos están restringidos a los divisores enteros de dicho período. Este esquema es mucho más práctico que el anterior, pero aún quedan algunos cabos sueltos: determinar cuál debe ser el valor del período principal y si un período en concreto debe ser modificado a la alza o a la baja.

5.1.2. Introducción de una métrica

El factor principal, aparte de si es armónico o no, es cómo de diferente es el nuevo problema del original. Para ello, hay que establecer una métrica. Una posibilidad podría ser simplemente la suma de las diferencias entre el período original y el modificado. Pero eso podría provocar que resultase más atractivo aumentar considerablemente los períodos de algún sistema crítico y dejar el resto tal cual que modificar muchos sistemas no críticos mientras se modifica poco o nada los sistemas más críticos. Una métrica más acertada es ponderar los cambios de período según si se trata de reducir los períodos, aumentar los sistemas de criticidad A, los sistemas de tipo B, los de criticidad C o los de criticidad D.

Una vez que se cuenta con esta métrica se puede decidir cuál es la mejor solución entre un grupo de ellas. Se puede, por tanto, fijar un período principal, hallar diferentes soluciones y determinar cual está más cerca del problema original. De entre las soluciones halladas hay que descartar todas aquellas que consuman más recursos de los disponibles, ya que éstos no serán planificables.

5.1.3. Seleccionar las mejores soluciones

Para hallar esas diversas soluciones se procede por backtracking. Los períodos no se modifican de unidad en unidad. Si fuera así, para encontrar una combinación que fuese armónica habría que comprobar una inmensa cantidad de posibilidades para encontrar una sola que fuera armónica. Prácticamente, se volvería al método por fuerza bruta. En vez de eso, se fuerza que los períodos sean divisores del período principal. En este caso, el numero de combinaciones en las que las tareas tienen períodos armónicos es del mismo orden que el número de ensayos.

Si se proponen varios períodos principales y para cada uno de ellos se obtienen varios resultados distintos, es posible averiguar cuál es el que más se acerca al problema inicial.

5.1.4. Parámetros de la búsqueda

De la exposición anterior solo resta concretar los límites de la búsqueda. El número de divisores diferentes que se prueban para cada período afecta en gran medida al rendimiento. Además, cuanto más lejos del valor inicial menos probable es que la solución encontrada sea buena. Si se impone un máximo de tres o cuatro valores para cada período se obtienen armonizaciones muy similares a las obtenidas cuando se permite más diversidad.

Finalmente queda determinar los valores que se van a probar como período principal. Dichos valores deben ser cercanos al mayor de los períodos para que el nuevo problema sea cercano al original. La diferencia entre el período principal fijado y el valor del mayor de los períodos es, en todo caso, la perturbación de ese período. Por tanto, si dicha diferencia es grande las penalizaciones serán también considerables y la calidad de las soluciones disminuye.

5.1.5. Detalles del algoritmo

Para resumir y concretar toda la exposición anterior se muestra a continuación el pseudocódigo del algoritmo. En este fragmento se encuentra recogido el núcleo del algoritmo, es decir, la parte que realmente marca el procedimiento para obtener los nuevos periodos.

Pseudocódigo del método de armonización.

```
procedimiento armonizar()
Si período[] es armónico
terminar
fin si
Para cada períodoPrincipal entre períodoMínimo y períodoMáximo
candidato := llamada a buscarArmónicos( períodoPrincipal, 1 );
impacto:= calcularImpacto( candidato )
Si impacto < menorImpacto
menorImpacto := impacto
mejor := candidato
fin si
fin para
fin procedimiento

procedimiento buscarArmónicos( períodoPrincipal, índicePeríodo )
Si períodoConsiderado > númeroPeríodos
Para cada índice entre 1 y númeroPeríodos
nuevosPeríodos[índice] := período[índice]+modificación[índice]
fin para
Si nuevosPeríodos[] es armónico
devuelve nuevosPeríodos[]
sino
devuelve nada
fin si
sino
Para cada paso entre 1 y pasoMáximo
modificación[índicePeríodo] := llamada a calcularDesviación(
```

```
períodoPrincipal, período[índicePeríodo], paso )
candidato := llamada a buscarArmónicos(
períodoPrincipal, índicePeríodo + 1)
impacto:= calcularImpacto( candidato )
Si impacto < menorImpacto
menorImpacto := impacto
mejor := candidato
fin si
fin para
fin si
fin procedimiento

procedimiento calcularDesviación ( períodoPrincipal, períodoInicial, paso )
máximoValor := |períodoPrincipal - períodoInicial|
Para cada índice entre 0 y el mayor entre períodoInicial y máximoValor
resta := períodoInicial - índice
Si resta > 0 y períodoPrincipal%resta = 0
paso := paso - 1
Si paso = 0
devuelve (-índice)
fin si
fin si
suma := períodoInicial + índice
Si suma < períodoPrincipal y períodoPrincipal%suma = 0
paso := paso - 1
Si paso = 0
devuelve (índice)
fin si
fin si
fin para
fin procedimiento
```

5.2. Detalles de la implementación

Se ha realizado una implementación del algoritmo explicado anteriormente para realizar las pruebas y mediciones. Al realizar la implementación se han tomado ciertas decisiones que pueden afectar al rendimiento.

El algoritmo se ha diseñado recursivo y en la implementación se ha mantenido así. En este caso, la creación de una versión iterativa es más desafiante que en el caso anterior.

En la implementación se ha optado por pasar como parámetros los objetos con que se opera (períodos antiguos, solución candidata, mejor impacto, etc) en vez de usar instancias accesibles. No hay una gran diferencia pero deja abierta la oportunidad de usar un mismo objeto para resolver diferentes problemas.

Como ya se ha dado a entender en apartados anteriores, se ha preferido un enfoque tal que el período principal que se escoge recorra una cantidad de valores predeterminada. Se podría haber implementado el algoritmo de modo que fuera ampliando la búsqueda hasta que el impacto bajase de un cierto umbral. En los casos en que no se alcanzase dicho umbral, habría que añadir un mecanismo extra para detener la búsqueda. Por otro lado, cuando se alcanzase dicho umbral el tiempo de ejecución dependería en gran medida de dónde se encuentre la primera solución que cumpla la condición dentro del espacio de soluciones.

El problema de la propuesta de fijar un umbral es el valor que se le asigna. El valor adecuado varía con cada caso y, prácticamente, el único modo de calcularlo (aparte de prueba y error, que resulta mucho más costoso que la solución anterior) es conociendo la solución que se quiere encontrar.

En la implementación realizada, además de comprobar que sea armónico, se comprueba que la solución no use más recursos de los disponibles. Es decir, el número de horas necesarias para satisfacer todas las tareas debe ser menor que el número de horas disponibles. La holgura usada para este cálculo es de un 10%.

Para implementar el cálculo del impacto se procede en dos etapas. La primera es calcular para cada período su impacto. Ello se obtiene multiplicando la cantidad que se le añade o resta por un factor que tiene en cuenta la restricción que se está forzando. Si se reduce un período, el impacto global es menor que si se amplia el período de un sistema crítico. Los multiplicadores usados se muestran en la tabla siguiente:

Tabla 5.1. Multiplicadores según criticidad

Tipo de modificación	Multiplicador
Sistema de tipo A	1
Sistema de tipo B	0,9
Sistema de tipo C	0,7
Sistema de tipo D	0,6
Reducción de período	0,3

La segunda parte consiste en sumar el impacto parcial calculado en el paso anterior para todos los períodos. Esta suma representa el impacto total de la perturbación aplicada sobre el problema inicial. Cuanto mayor sea el impacto calculado, más alejado está el problema perturbado del problema inicial.

5.3. Resultados

En los apartados que vienen a continuación se va detallando cómo afectan en los resultados obtenidos los cambios que se realizan en los parámetros que se han comentado previamente.

5.3.1. Influencia del número de tareas

Se han realizado diferentes ejecuciones manteniendo constantes todos los parámetros del problema menos el número de tareas.

Se hizo oscilar el período principal entre el período máximo - 10 y el período máximo + 10, es decir, tomó veintiún valores diferentes. Se consideraron para cada período los tres posibles valores más próximos al inicial. Los resultados están recogidos en la tabla que aparece a continuación.

Tabla 5.2. Variación con el número de tareas

Número de tareas	Tiempo
10	12
11	36
12	105
13	410
14	1050

Al representar los valores recogidos en una gráfica, como la que se puede ver más abajo, se puede comprobar que el crecimiento del tiempo empleado se dispara al aumentar el número de tareas.

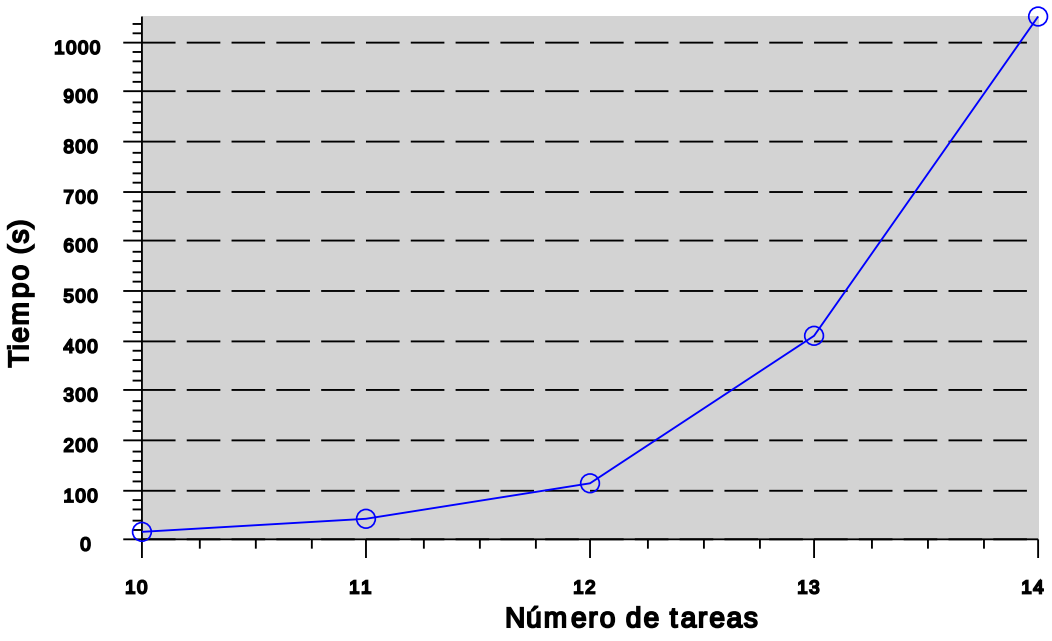

Figura 5.1. Variación con el número de tareas.

Si esos mismos datos se representan en escala logarítmica, se puede apreciar más claramente la tendencia. En la gráfica que se incluye abajo, se puede ver como el crecimiento tiene un carácter claramente exponencial.

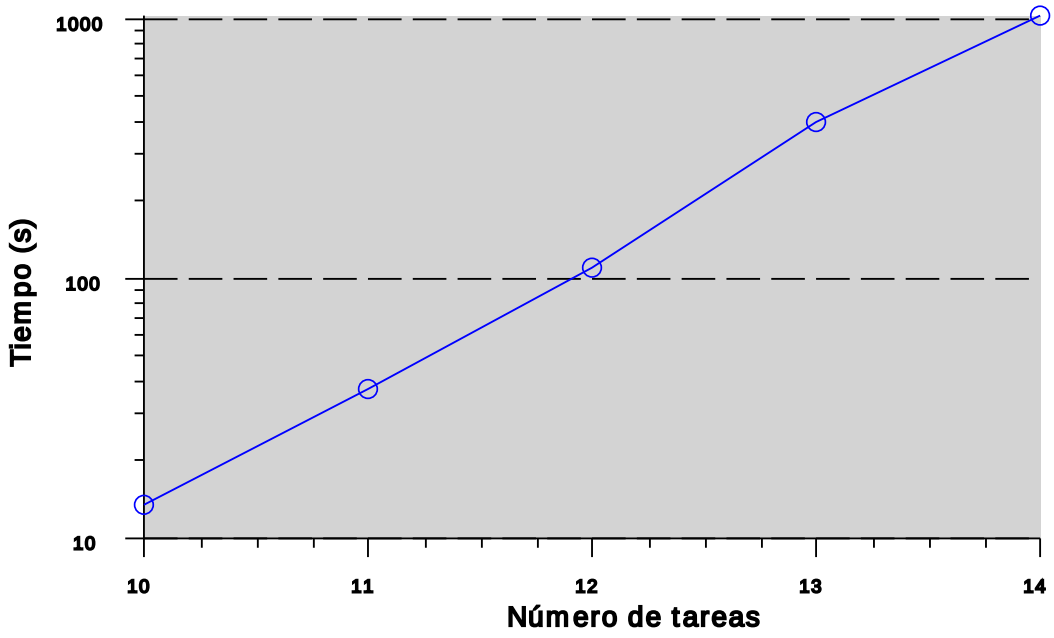

Figura 5.2. Variación con el número de tareas (escala logarítmica).

5.3.2. Influencia del número de valores para cada período

Se realizaron experiencias para aislar los efectos debidos a la variación del número de valores distintos que se consideran para cada período durante el proceso de armonización para un período principal concreto.

En las pruebas realizadas se mantuvo el número de tareas constante. Se utilizaron problemas que constaban de doce tareas. Se hizo oscilar el período principal entre el período máximo - 10 y el período máximo + 10, es decir, tomó veintiún valores diferentes.

Tabla 5.3. Variación con el número de valores por período

Valores diferentes por período	Período principal	Tiempo
1	169	0,11
2	168	2,4
3	168	105
4	168	2030

Como se podía esperar, la influencia de este parámetro sobre el tiempo de ejecución es muy grande. En cada iteración del algoritmo hay que repetir ese número de veces los cálculos para hallar el período armónico correspondiente.

Si se muestra de forma gráfica, se ve un crecimiento pronunciado.

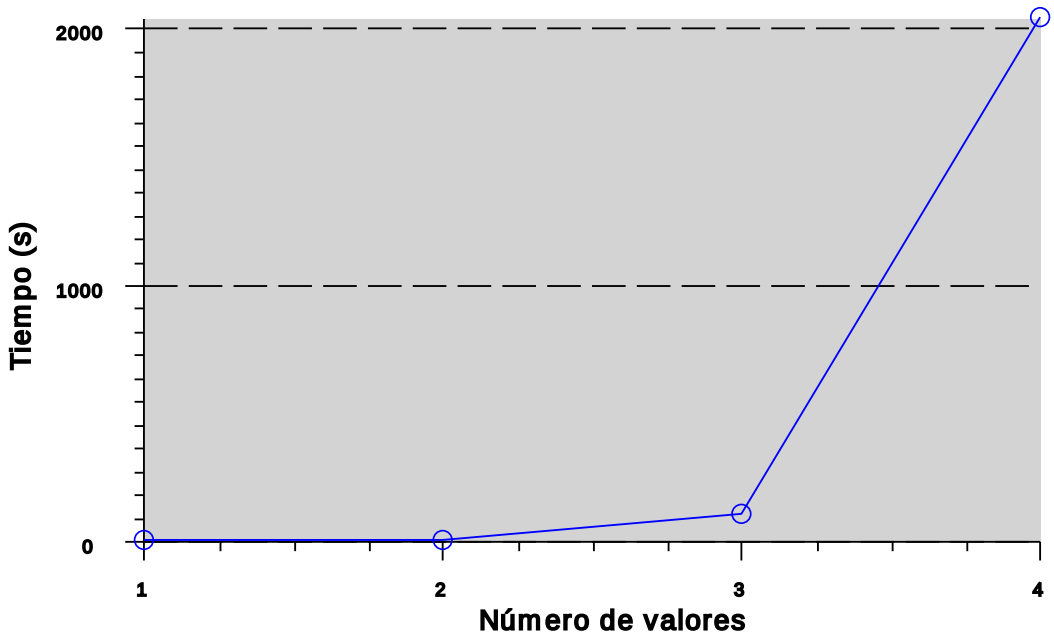

Figura 5.3. Variación con el número de valores por período.

De nuevo, la naturaleza exponencial de la influencia encontrada salta a la vista.

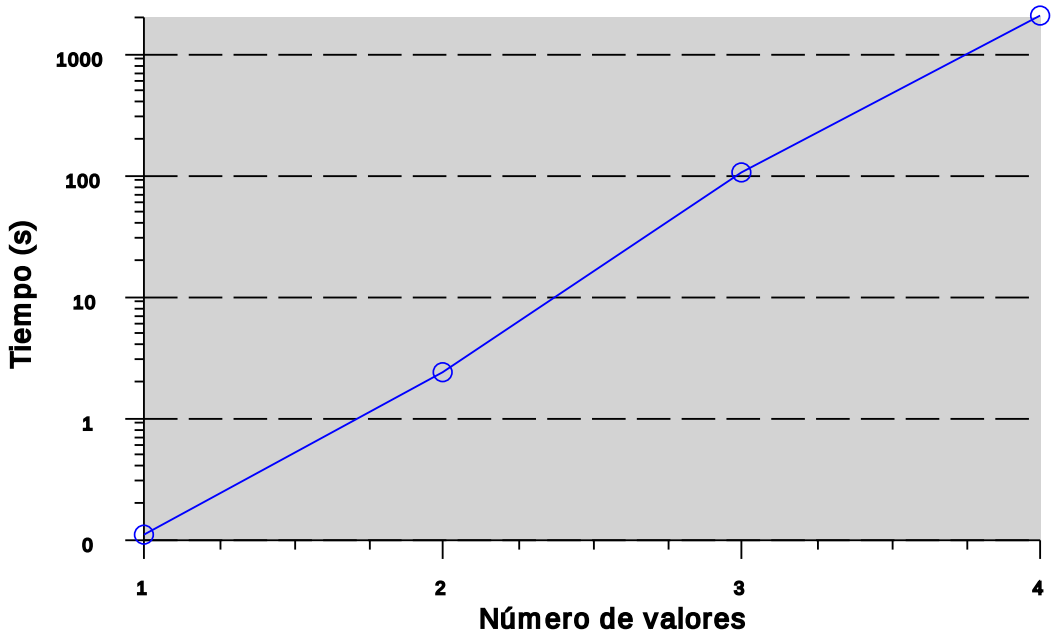

Figura 5.4. Variación con el número de valores por período.

5.3.3. Influencia del número de valores para el período principal

En las pruebas realizadas se mantuvo el número de tareas constante. Se utilizaron problemas que constaban de doce tareas. Se consideraron para cada período los tres posibles valores más próximos al inicial. Los resultados están recogidos en la tabla que aparece a continuación.

Tabla 5.4. Variación con el número de valores para el período principal

Valores diferentes para el período principal	Período principal	Tiempo
1	173	0,23
11	168	69
21	168	105
31	168	160

A continuación se pueden visualizar de forma gráfica los resultados recopilados. Se puede observar que el crecimiento sigue un comportamiento aproximadamente lineal.

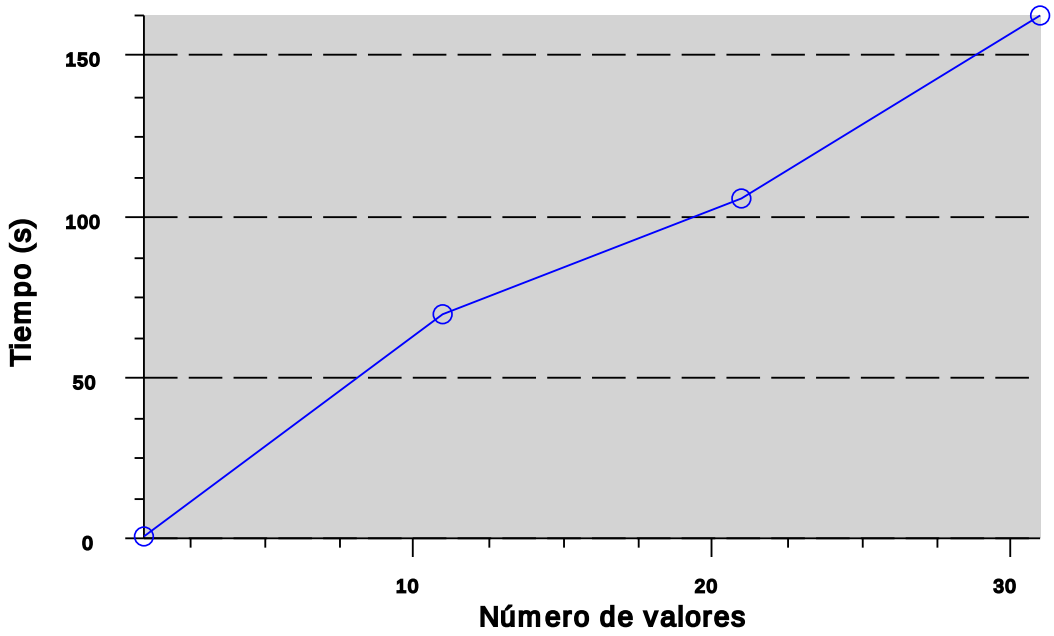

Figura 5.5. Variación con el número de valores para el período principal

5.4. Ejemplo de ejecución en un caso real

Al igual que en el capitulo anterior, se va a realizar una ejecución con datos reales para tener una apreciación de cómo funcionaría este algoritmo en producción. Se va a realizar la armonización de las tareas del trayecto Villalba de Guadarrama - Cercedilla.

En un primer paso, se realiza una transformación de las tareas contenidas en el problema inicial. Los parámetros usados en esta ejecución coinciden con los empleados en las pruebas anteriores. Los valores probados para el período principal oscilan entre diez por encima del período máximo de las tareas y diez por encima de dicho período máximo. Para cada tarea se prueban tres valores.

Con esa configuración, el algoritmo encuentra un problema equivalente en 8400 segundos aproximadamente (unas dos horas y media).

El problema equivalente encontrado se planifica con 3 horas laborables diarias y el calendario laboral del año 2006. El factor de utilización de los recursos obtenido con estos datos es del 96,2%. El sistema encuentra una solución factible en 6300 segundos aproximadamente, lo que equivale a una hora y tres cuartos.

5.5. Evaluación de la solución

En los apartados anteriores, se discutió cómo y por qué realizar una armonización de los períodos de las tareas que conforman el problema para hallar un problema similar pero con características más deseables.

Se comprobó que los problemas de escala planteados por el método anterior no son menores en éste que en aquel. El aumento de la complejidad con el número de tareas es exponencial y algunos parámetros del algoritmo que determinan los problemas equivalentes que se consideran afectan también radicalmente al rendimiento.

Su principal bondad estriba en que las transiciones entre planificaciones son perfectas. Ésto es una gran diferencia con respecto al problema sin armonizar, que ya se vio en el capitulo anterior que tiene problemas cerca del final del período planificado. Si al terminar una planificación obtenida mediante esta técnica empieza otra, ninguno de los plazos del problema modificado se incumplen.

Debido a como se ha obtenido, es posible que algunos de los plazos del problema original sí que sufran retrasos, pero serán tan solo aquellos de baja prioridad y los retrasos serán lo menores que sea posible. El valor de esta desviación del comportamiento deseado depende de lo cerca que este el problema original de ser armónico y del tiempo de trabajo disponible frente al tiempo necesario (factor de utilización).

Este método es más adecuado cuando se trata de planificar un conjunto no muy grande de tareas durante un tiempo indefinido y los períodos originales han sido calculados de modo aproximado. Si la exactitud de los períodos de ejecución de las tareas es crítica, no se aconseja utilizar este método.

Capítulo 6. Uso de simulated annealing para programar las tareas

6.1. Descripción de la técnica

Los casos de prueba realizados usando el procedimiento por backtracking de encontrar una planificación factible arrojan un resultado interesante. Para un mismo número de tareas y clases de equivalencia, el hecho de que se alcance una solución o que se tenga que agotar el espacio de búsqueda supone una gran diferencia en el tiempo de ejecución. Esto apunta a que, en la mayoría de los casos, basta con recorrer una pequeña porción del espacio de búsqueda para encontrar una solución factible.

La dificultad consiste en seleccionar una porción adecuada del espacio de búsqueda. La aproximación elegida para llevar esto a cabo es realizar búsquedas locales iterativas. En este caso, se usaran técnicas de simulated annealing. En el capítulo tres ya se discutió las razones de esta elección, ahora el discurso se centrará en cómo adaptar esta técnica al problema concreto.

En primer lugar, se va a determinar la forma que va a adoptar la solución para admitir el concepto de vecindad. Esta forma debe permitir calcular un valor único para la energía del sistema para cada solución. Se ha decidido que la representación de los candidatos a solución, consista en una lista ordenada que contenga a todas las tareas una y solo una vez.

La planificación correspondiente consiste en añadir a una planificación vacía las tareas en el orden en el que se encuentran en la lista ordenada, hasta que se hayan añadido todas las tareas a la planificación o hasta que una tarea no pueda ser asignada. Si se vuelve a la imagen del árbol que representa el espacio de búsqueda, cada solución candidata se puede considerar una ruta desde la raíz del árbol hasta uno de los nodos hoja.

Dado un candidato, sus vecinos se encuentran mediante el procedimiento de intercambiar la posición de dos de las tareas dentro del orden de asignación. Las tareas intercambiadas deben pertenecer a clases de equivalencia diferentes. Si las tareas intercambiadas pertenecieran a la misma clase de equivalencia no se produciría cambio alguno en la energía del sistema, ni en la posición dentro del espacio de soluciones.

Una vez que ya se ha definido lo que es un estado del sistema, hay que determinar qué nivel de energía le corresponde. En realidad, es suficiente con determinar la diferencia de energía entre dos estados pero, en este caso, se puede fijar una cantidad para cada estado. La energía del sistema está, en cierta medida, relacionada con el desorden del sistema. En un estado de mínima energía, el sistema tiene las tareas distribuidas de modo que el tiempo se aprovecha al máximo. Por tanto, la medida de la energía que se usa es el número de horas libres diarias que hay de media. Cuantas más horas libres tiene el sistema, más energía tiene el sistema, ya que las tareas están peor colocadas.

En el diagrama de más abajo se puede observar como se desarrolla el algoritmo. Hay que recordar que la probabilidad p que se usa para determinar si aceptar soluciones con mayor energía que el actual paso de la cadena de Markov depende de la temperatura.

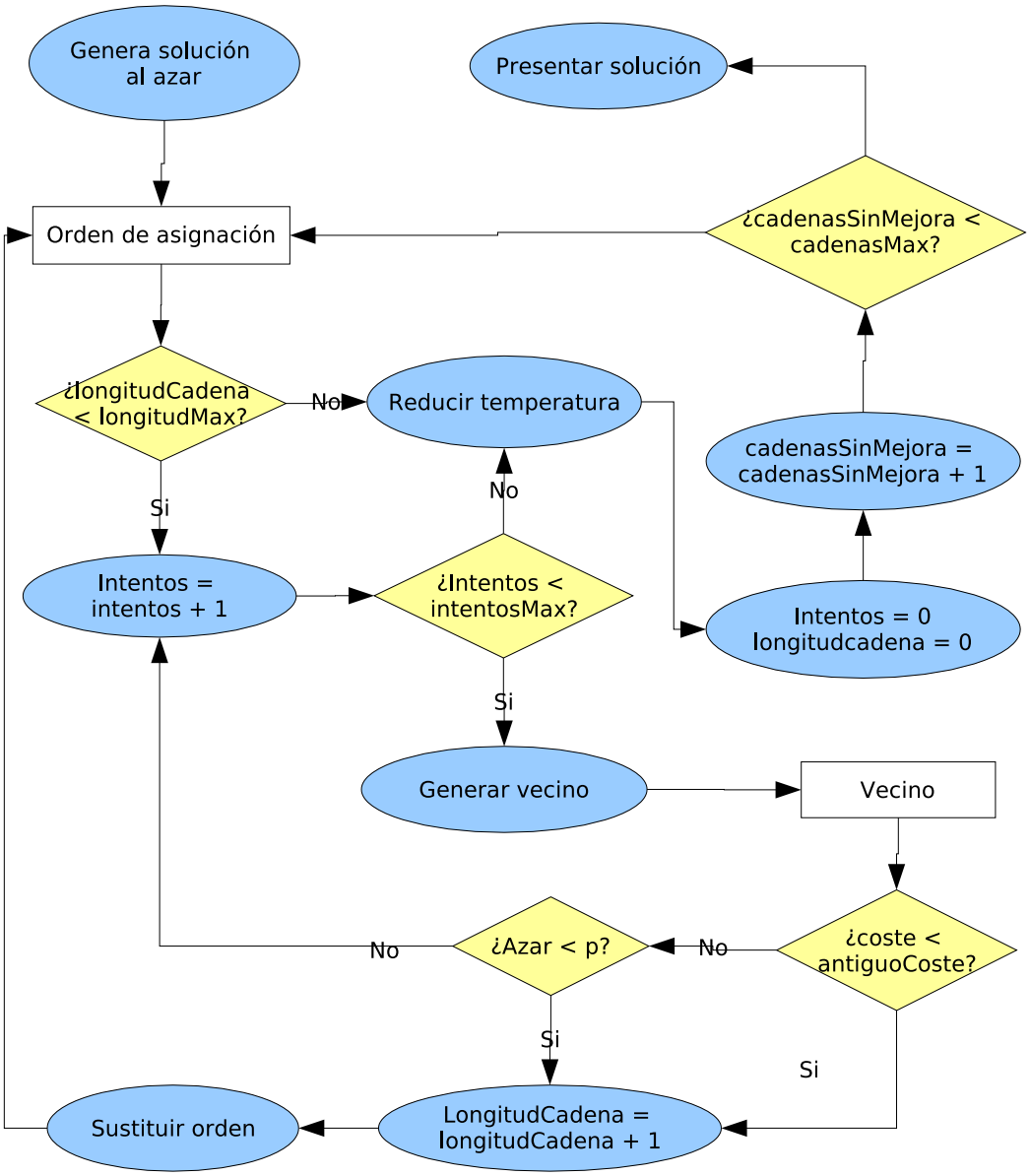

Figura 6.1. Algoritmo de simulated annealing para la planificación de tareas

A continuación, se muestra el pseudocódigo del núcleo del algoritmo. Se puede dividir el proceso en la repetición de una secuencia básica con temperaturas sucesivamente más frías. Cada uno de esos procesos consiste básicamente en crear una cadena de Markov de una determinada longitud o con un determinado número de intentos, lo que ocurra antes.

Pseudocódigo del método por simulated annealing.

```
procedimiento enfriamientoLento( soluciónInicial )
temperatura := TEMPERATURA_INICIAL
cadenasSinMejora := 0
mejorSolución := soluciónInicial
hacer
candidato := cadenaDeMarkov( temperatura , soluciónInicial )
si candidato.coste < mejorSolución.coste
mejorSolución := candidato
cadenasSinMejora := 0
sino
cadenasSinMejora := cadenasSinMejora + 1
fin si
temperatura := temperatura * VELOCIDAD_ENFRIAMIENTO
mientras cadenasSinMejora < CADENAS_MÁXIMAS
devuelve mejorSolución
fin procedimiento

procedimiento cadenaDeMarkov( temperatura, soluciónInicial )
intentos := 0
aceptados := 0
soluciónActual := soluciónInicial
mientras intentos < INTENTOS_MÁXIMOS Y aceptados < LONGITUD_CADENA
candidato := generaVecino( soluciónActual )
intentos := intentos + 1
si aceptaVecindad( candidato, soluciónActual, temperatura )
soluciónActual := candidato
aceptados := aceptados + 1
fin si
fin mientras
fin procedimiento
```

6.2. Parámetros del algoritmo

El comportamiento del algoritmo viene regido por la elección que se realice de los valores de ciertos parámetros. Este método es probabilístico y, por tanto, dos ejecuciones con idénticos parámetros y sobre el mismo problema pueden comportarse de forma diferente. A pesar del papel del azar en los resultados que ofrece el algoritmo, la capacidad para encontrar una solución válida dentro del espacio de soluciones y el tiempo empleado para encontrarla están muy relacionados con dichos parámetros del algoritmo.

Velocidad de enfriamiento. Este parámetro consiste en la velocidad con la que desciende la temperatura del sistema. Un enfriamiento demasiado rápido puede impedir que se exploren las suficientes rutas en las que la solución no mejora como para que se evite caer en un mínimo local. En cambio, si la temperatura decae con excesiva lentitud, se prueban un número excesivo de rutas, con lo que se pierde parte de la ventaja de no tener que explorar totalmente el espacio de soluciones.

Temperatura inicial. No es tan importante cuál es la temperatura inicial del sistema como cuál es la relación entre la temperatura inicial del sistema y el incremento de energía medio entre soluciones vecinas. La temperatura del sistema indica cuál es la probabilidad de aceptar un determinado paso de un estado del sistema a otro y la temperatura máxima señala un techo de probabilidad. Si el techo es demasiado restrictivo, la probabilidad de que el sistema evite el mínimo local y encuentre la solución sera menor. Por otro lado, si la temperatura inicial es muy alta, el algoritmo pierde mucho tiempo intentando soluciones al azar ya que casi cualquier vecino sera aceptado como parte de la cadena de Markov.

Longitud de la cadena. La cadena de Markov para una determinada temperatura debe terminar en algún punto. Puede acabar de dos formas, cuando la cadena tiene la longitud especificada en este parámetro o cuando se han probado un determinado número de vecinos. Este parámetro es más relevante a altas temperaturas que a bajas. Cuanto mayor sea, más larga es cada una de las búsquedas iterativas que componen este método. Debe ser lo bastante grande como para permitir que la diferencia entre el primer elemento de la cadena y el último sea significativa.

Número de intentos por cadena. Este parámetro junto con el anterior, determinan el final de cada una de las cadenas de Markov que se utilizan en el algoritmo. La importancia de este parámetro es mayor cuando la tasa de aceptación de vecinos disminuye. Si este número es excesivamente pequeño, no se explora de forma exhaustiva la región donde se espera encontrar la solución. En cambio, si es demasiado grande, el sistema carece de dirección a altas temperaturas.

Número de cadenas sin mejora. Como la temperatura tiende de forma asintótica a cero, el proceso puede ser infinito. Para evitar este problema, la ejecución del algoritmo se detiene cuando después de que se hayan probado este número de cadenas, ninguna de ellas haya mejorado la solución obtenida. Si este número resulta ser demasiado bajo, el sistema puede acabar en un mínimo local. Si el valor de este parámetro es excesivamente elevado se puede continuar la ejecución un tiempo notable sin que, por ello, la solución obtenida mejore.

Como se ve, el ajuste de los parámetros se mueve, en muchos casos, entre la pérdida o degradación de la solución obtenida por un lado y el desperdicio de tiempo de ejecución por el otro. Parámetros muy restrictivos pueden evitar que se explore la parte del espacio de búsqueda donde se haya la solución y, por contra, parámetros muy laxos hacen que la búsqueda no se dirija de forma más o menos directa a la solución, sino que vague por el resto del espacio de búsqueda.

No existe ninguna técnica para encontrar los valores de los parámetros adecuados a un problema concreto. Hay que fijarlos con algo de sentido común y realizando ensayos previos para afinar los valores según el comportamiento apreciado.

6.3. Detalles de la implementación

Para la realización de las pruebas que se detallan en el próximo apartado se ha realizado una implementación del algoritmo presentado. Dicha implementación tiene detalles que bien no se han especificado en el caso general o bien difieren ligeramente de lo expuesto anteriormente.

La generación de vecinos de una solución se realiza mediante un proceso aleatorio. En primer lugar, se seleccionan al azar dos tareas. Si las dos tareas pertenecen a clases de equivalencia diferentes, se intercambian sus posiciones dentro del orden de asignación a la planificación. En caso de que pertenezcan a la misma clase de equivalencia, se conserva una de ellas y se selecciona una tarea diferente hasta que tengan clases distintas. Hay que recordar que intercambiar dos tareas de la misma clase de equivalencia entre si deja la planificación intacta.

En la implementación usada se ha normalizado tanto la energía del sistema como la temperatura. En principio, no se conoce la energía máxima del sistema, lo que puede dificultar la normalización. Este problema se ha solventado tomando como referencia la energía del sistema antes del cambio de estado para cada transición. Ésto puede provocar que el gasto energético de dos transiciones iguales en términos absolutos sean diferentes calculados bajo esta regla, pero permanece incólume la propiedad de que las transiciones entre estados con una diferencia de energía mayor son menos probables que aquellos que suponen un aumento de energía menor.

Dado que se opera en una maquina con precisión finita, se ha establecido un umbral para el cociente entre la variación de energía y la temperatura del sistema. A partir de ese umbral, se entiende que la probabilidad de que se acepte el vecino es cero. El umbral se ha establecido cuando dicho cociente alcanza el valor 100, cuyo exponencial es aproximadamente $3.72*10^{-44}$.

6.4. Resultados

Se realizaron pruebas previas del algoritmo y se determinaron unos valores adecuados de los distintos parámetros que controlan su funcionamiento. Estos valores se usaron en el resto de las pruebas.

- Velocidad de enfriamiento: 0,9
- Temperatura inicial: 1
- Longitud de la cadena: 50
- Número de intentos por cadena: 100
- Número de cadenas sin mejora: 20

6.4.1. Comportamiento con respecto al número de tareas

Se compara el comportamiento de este algoritmo con el de planificación por backtracking para diferentes números de tareas. Primero se realizaran pruebas para obtener una idea general de cómo influye en el tiempo medio que se puede tardar en encontrar una solución el número de tareas. En la tabla siguiente se muestran los valores obtenidos.

Tabla 6.1. Comparación de algoritmos en caso de éxito

Tareas	**Tiempo backtracking (s)**	**Tiempo simulated annealing (s)**
6	0,06	0,13
10	0,4	0,26

Tareas	Tiempo backtracking (s)	Tiempo simulated annealing (s)
12	0,6	2
14	2,4	0,7
20		1
24		7

Los resultados no siguen una tendencia clara. Aún así, se puede ver que el crecimiento (aunque puntualmente sufre bruscos aumentos) es más lento que el de uno de tipo exponencial. A continuación se representa de forma gráfica los datos obtenidos.

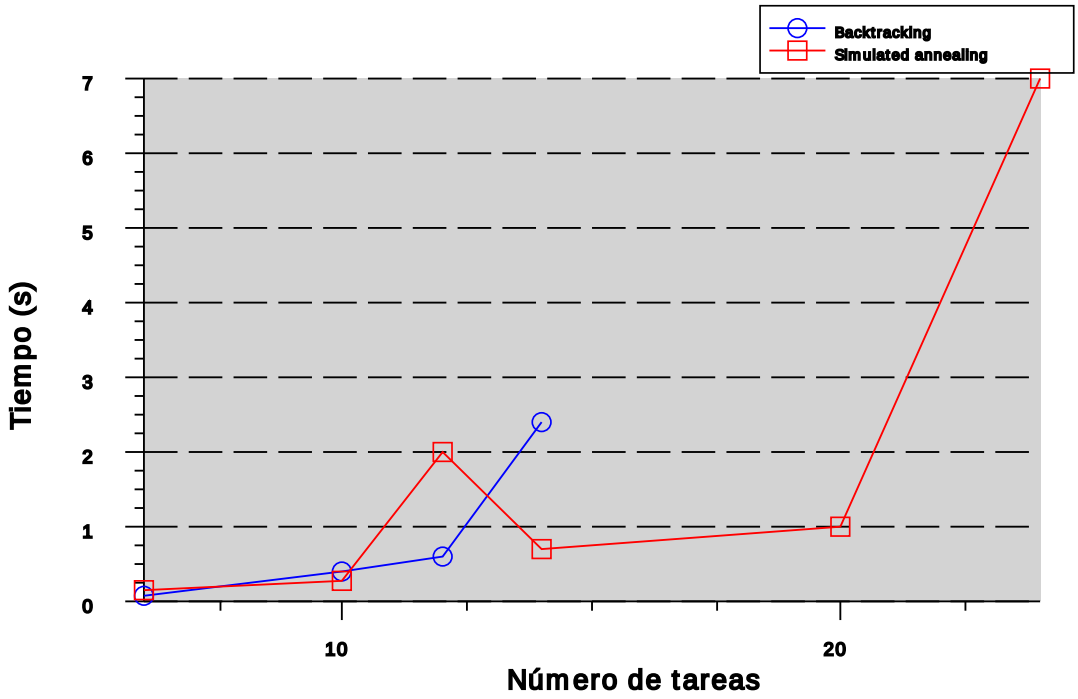

Figura 6.2. Comparación según el número de tareas (éxito)

Cuando el algoritmo encuentra una solución factible se detiene. Por ese motivo, el peor caso ocurre cuando no se puede planificar el conjunto de tareas. Se realiza, pues, una batería de pruebas donde no se encuentra la solución. Los resultados se encuentran en la tabla siguiente.

Tabla 6.2. Comparación según el número de tareas (fallo)

Tareas	Tiempo backtracking (s)	Tiempo simulated annealing (s)
6	0,23	20
10	3	40
14	104	40
20	8700	80
24		120
48		400

En la imagen que aparece a continuación se muestra, de forma gráfica, la diferencia entre los dos algoritmos en el caso de que no se encuentre una planificación factible.

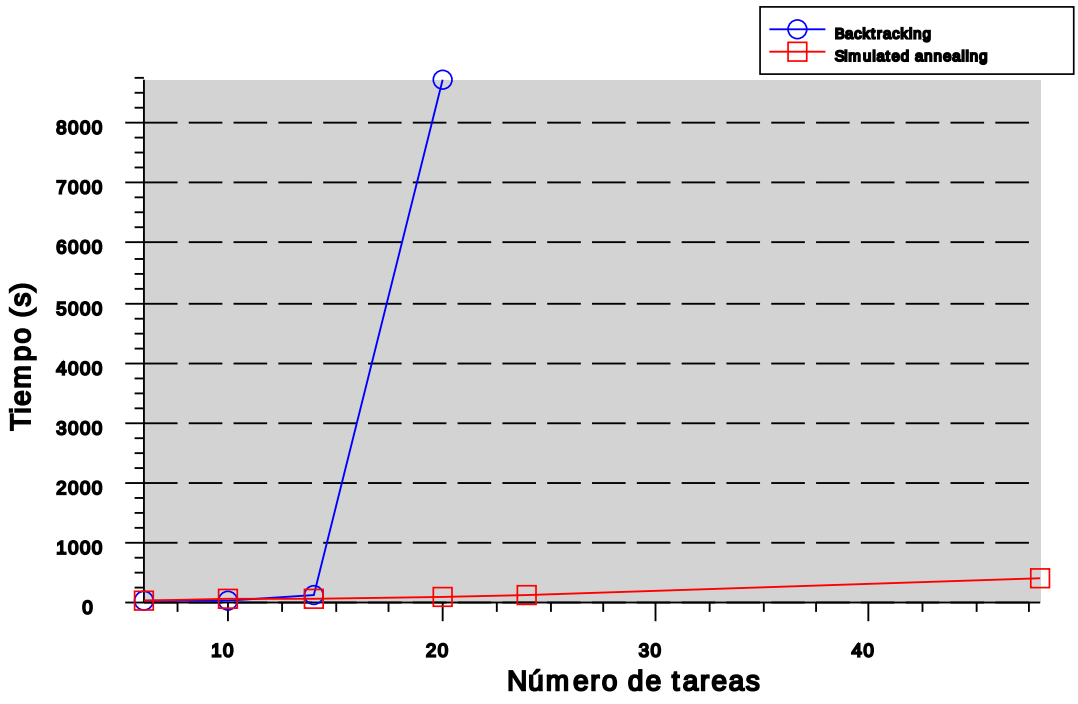

Figura 6.3. Comparación según el número de tareas (fallo)

Tal vez la gráfica que se muestra a continuación sea más ilustrativa. En esta imagen los datos se han representado con una escala logarítmica.

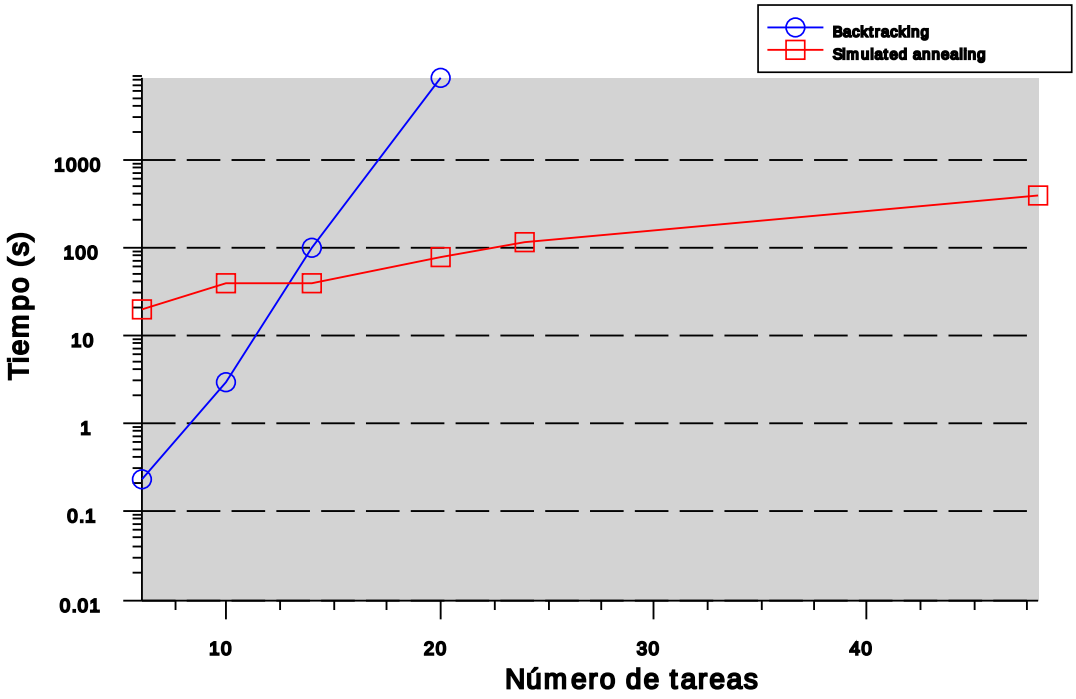

Figura 6.4. Comparación según el número de tareas (fallo)

En los casos propuestos se encuentra una solución en todos aquellos casos que contienen una. Ésto no tiene por que ser así, ya que al ser un método probabilístico siempre existe la posibilidad de que ninguna de las cadenas de Markov pase por la solución. Si los parámetros están bien dimensionados, dicha probabilidad sera muy pequeña, aunque nunca cero.

En el caso de que encuentren una solución factible, ambos métodos tienen un rendimiento similar. Ésto es así porque cuando cualquiera de los dos algoritmos encuentra una solución factible se detiene. La diferencia consiste en que el comportamiento del primero depende de donde se encuentre la solución dentro del espacio de búsqueda, ya que es secuencial y el segundo depende de la heurística y del azar.

Si se considera ahora el otro caso, cuando no existe una solución factible dentro del espacio de búsqueda, la diferencia es notable. El algoritmo de simulated annealing es capaz de discriminar que probablemente no hay solución examinando una pequeña parte del espacio de soluciones. Esto provoca que, cuando el algoritmo de fuerza bruta tarda tiempos desesperantes para una respuesta interactiva, el simulated annealing da respuestas en tiempos cómodos para un ser humano.

El algoritmo de simulated annealing escala de una forma poco predecible. El número de soluciones consideradas y, por tanto, el tiempo de ejecución, depende de varios factores. Entre ellos se encuentra el número de veces que puede descender la energía cuando la temperatura comienza a ser baja. En efecto, cada vez que una cadena obtiene un resultado mejor que el mejor histórico, se reinicia la cuenta de las cadenas que no proporcionan mejoras. Este valor está relacionado con el número de niveles de energía del sistema, que a su vez depende del problema concreto. Al estar las transiciones basadas en la probabilidad, diferentes ejecuciones del mismo problema pueden tener rendimientos sensiblemente diferentes.

A continuación se dan tiempos típicos de respuesta para conjuntos de tareas con tamaños más reales. Se han usado en todos los casos los factores de utilización más próximo al 90% de utilización como ha sido posible.

Tabla 6.3. Tiempos de ejecución en simulated annealing

Número de tareas	**Utilización (tanto por uno)**	**Soluciones probadas**	**Tiempo (s)**	**Tiempo por solución (s)**
100	0.95	4793	65.35	0.014
150	0.94	5814	173.045	0.030
200	0.90	6685	322.14	0.048
250	0.92	9232	614.00	0.066
300	0.91	12604	1357.98	0.108

Número de tareas	Utilización (tanto por uno)	Soluciones probadas	Tiempo (s)	Tiempo por solución (s)
350	0.89	10088	1212.63	0.120

En la imagen que se puede ver a continuación se representa el comportamiento del algoritmo para este número de tareas.

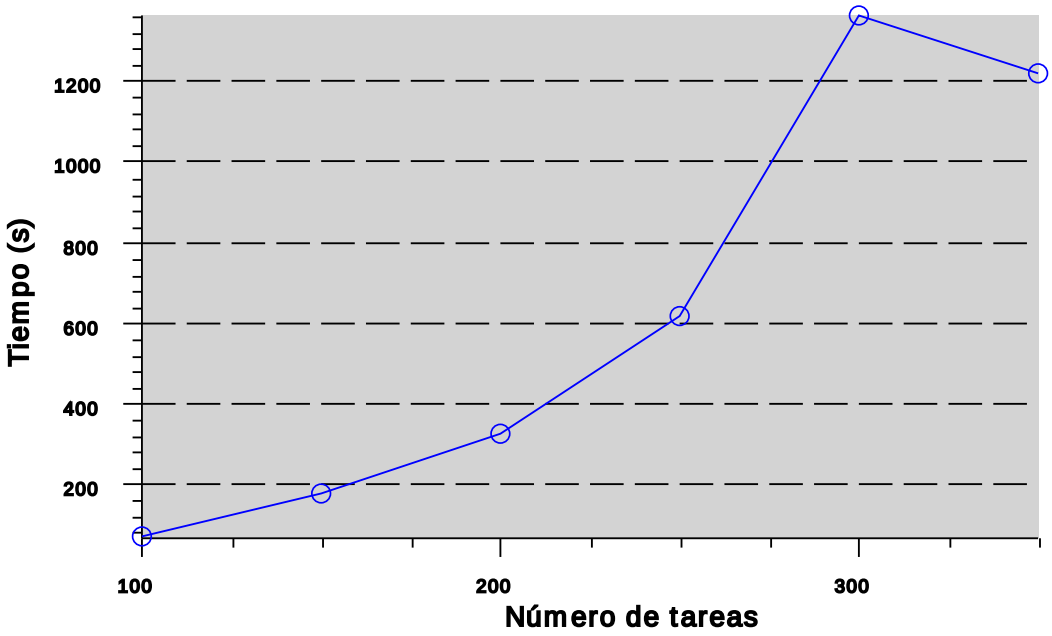

Figura 6.5. Comparación según el número de tareas (fallo)

Para el mismo problema, en diferentes ejecuciones, se pueden obtener resultados diferentes. El número de pasos que lleva alcanzar la solución tiende a crecer, aunque no con una tendencia clara. Se puede decir que dados dos problemas, el que tenga una complejidad mayor (un número mayor de tareas y clases de equivalencia) tiene mayor probabilidad de explorar más soluciones.

Sin embargo, si se centra la atención sobre el tiempo medio que lleva explorar cada una de las soluciones, se obtiene la gráfica que se muestra a continuación.

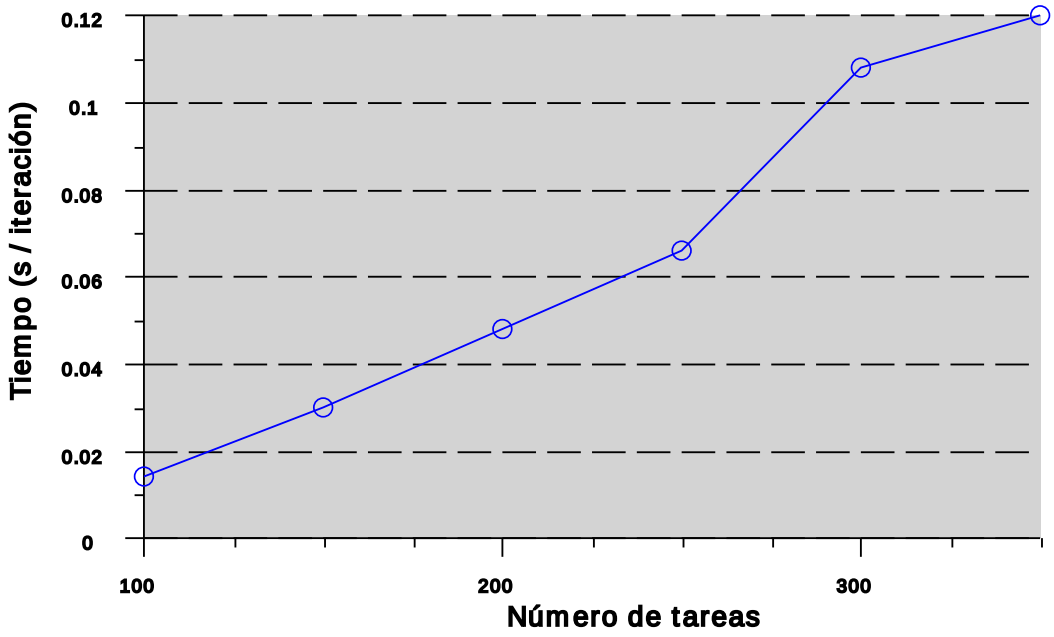

Figura 6.6. Duración de cada solución propuesta según el número de tareas

El calculo de cada solución crece de forma lineal con el número de tareas. Si la esperanza matemática del número de iteraciones crece de forma polinómica, la duración total del algoritmo crecería a su vez de forma polinómica.

6.5. Ejemplo de ejecución en un caso real

Se va a considerar el comportamiento real del algoritmo con un ejemplo real. Se va a considerar el caso en que se quiera planificar las tareas de mantenimiento del trayecto Villalba de Guadarrama - Cercedilla.

Se realiza la ejecución con los parámetros, que se han fijado para las distintas pruebas realizadas previamente en este capítulo, Como se mencionó antes, los valores usados han sido:

- Velocidad de enfriamiento: 0,9
- Temperatura inicial: 1
- Longitud de la cadena: 50
- Número de intentos por cadena: 100
- Número de cadenas sin mejora: 20

Se ejecuta la planificación con 3 horas laborables diarias y el calendario laboral del año 2006. El factor de utilización de los recursos obtenido con estos datos es del 91,9%. El sistema encuentra una solución factible en poco más de diecisiete segundos.

6.6. Evaluación de la solución

El algoritmo que se ha discutido en los apartados anteriores tiene la desventaja de que no es determinista. Dos ejecuciones distintas sobre el mismo conjunto de tareas pueden ofrecer resultados distintos. No solo las soluciones pueden ser distintas, sino que una ejecución puede encontrar una solución factible y la otra sea incapaz de hallar ninguna. La probabilidad de que ocurra esto viene dada por cómo se han definido los parámetros de la búsqueda iterativa, pero para el problema que nos ocupa siempre será pequeña.

El tiempo que tarda el algoritmo para encontrar una solución factible viene dado por dos factores que dependen, en gran medida, del número de tareas de que conste el problema y de como estén distribuidos: el número estimado de iteraciones totales y el tiempo medio por iteración. Estos dos factores determinan cómo va a escalar el problema cuando se aumente la carga.

En el capítulo anterior se vio que el crecimiento del tiempo de cálculo de cada iteración crece de forma, aproximadamente, lineal con el número de tareas. La pendiente de la recta, además, es de pendiente suave.

No es fácil predecir el número de iteraciones para un problema concreto. Los parámetros concretos que se determinen para la ejecución influyen en cierta medida en el número de soluciones a probar. También influye en gran medida cómo se ha definido la energía interna. La energía interna debe ser de tal modo que guíe al algoritmo hacia la solución. Si la energía no cumple bien su papel, la búsqueda tendrá un mayor componente aleatorio y puede degenerar, en el peor de los casos, en búsqueda al azar. Con los parámetros y la definición de energía interna que se han usado en las pruebas el crecimiento ha estado por debajo de una exponencial.

El tiempo total de la ejecución del algoritmo será entonces la multiplicación de los dos factores mencionados. La composición de las dos tendencias hace que el crecimiento de la duración de la ejecución con respecto al número de tareas consideradas sea sub-exponencial. Es decir, escala mejor que el algoritmo de backtracking.

Si se quiere encadenar planificaciones sucesivas se encuentra el mismo problema que para el algoritmo de backtracking. Si las tareas no son armónicas, no hay instantes críticos y, por tanto, no aparecen puntos donde hacer el tránsito de una planificación a otra de forma natural. Las mismas técnicas que se aplican para realizar el cambio de planificación en ese caso se pueden aplicar en éste.

Este algoritmo resulta más apropiado cuando el número de tareas es mayor y están dispersas entre más clases de equivalencia distintas. Al ser, en general, más rápido que el de backtracking puede estar más indicado cuando el tiempo para hallar la solución sea determinante. Este algoritmo no está recomendado para los casos en los que sea imprescindible hallar una solución cuando exista, ya que, existen algunos pocos casos donde se puede no hallar una planificación factible aunque exista.

Capítulo 7. Conclusiones y trabajos futuros

7.1. Conclusiones del trabajo

En el presente trabajo se han evaluado algoritmos tradicionales para alcanzar planificaciones factibles que satisfagan los plazos de ejecución de conjuntos de tareas, junto con aproximaciones inspiradas en las técnicas de inteligencia artificial.

La técnica del recocido ha demostrado que puede resolver el problema de las planificaciones en un calendario real más rápido que las técnicas deterministas. También ha demostrado que escala mejor que las técnicas directas como la de backtracking.

Pero la utilidad de este trabajo no se limita a proponer y evaluar los diferentes algoritmos presentados. Una vez se ha comprobado que el problema de la planificación se adapta bien a las técnicas de la IA, queda abierta la puerta para usar otras técnicas heurísticas para atacar dicho problema.

Probablemente, la aportación más universal de este trabajo sea reducir la complejidad de este tipo de problemas partiendo el espacio en clases de equivalencia. Todas aquellas técnicas que intenten realizar planificaciones como si se tratase de hallar un nodo hoja factible pueden aprovechar esa aproximación al problema para reducir la complejidad. No solo se puede aplicar para las planificaciones sobre un calendario real, si no que también es valido para planificaciones en tiempo homogéneo.

7.2. Trabajos futuros

A lo largo de la realización de este trabajo se han ido realizado asunciones y simplificaciones allí donde parecía necesario. Se pueden explorar modos de eliminar dichas limitaciones auto-impuestas en aras de una mayor generalidad.

7.2.1. Mejoras sobre la definición del calendario

El calendario con el que se trabaja consta de días laborables, idénticos entre si y días festivos. No costaría demasiado introducir el concepto de media-fiesta (jornadas reducidas) o excepciones (días en el que la fuerza productiva es menor o mayor de la habitual).

7.2.2. Mejoras sobre la gestión de los desplazamientos entre tareas

Una de las mayores diferencias entre el mantenimiento de infraestructuras ferroviarias y otros tipos de mantenimiento, por ejemplo, en la planta de una fábrica, consiste en la ubicación geográfica de los sistemas mantenibles. Desde este punto de vista, la gestión del desplazamiento de las cuadrillas de trabajo realizada en este proyecto puede ser simplista en exceso.

En concreto, se podría realizar una evaluación dinámica de las horas laborables de un día según la ubicación de las tareas que contiene. Supondría una modificación del método que calcula si una tarea cabe en un determinado día o no. La mayor desventaja de esta posibilidad, aparte de la mayor complejidad y necesidades de computo, consiste en que rompe el concepto de clase de equivalencia. Dos tareas con idénticos período y duración, pero con distinta ubicación geográfica no pertenecerían a la misma clase de equivalencia.

Aún así, se puede definir otra clase de equivalencia con los elementos de idéntico período, duración del mantenimiento y ubicación geográfica. Debido a que gran parte de los sistemas se acumulan en los enclavamientos, habría cierta agrupación en clases de equivalencia, pero habría que comprobar si lo bastante como para resultar útil.

7.2.3. Mejoras sobre el proceso de armonización

Tal y como se implementa la armonización en este trabajo el proceso de armonización tiene un punto negativo. A la hora de buscar el período principal, se prueba con un número fijo de posibilidades. Sería más correcto fijar un umbral de aceptación y probar períodos de manera secuencial hasta que se alcance.

La armonización de los períodos del problema inicial tiene dos propósitos básicos: simplificar el árbol de búsqueda cuando se utiliza junto con sistemas de planificación mediante vuelta atrás y garantizar la existencia de instantes críticos cuando se quiere poder extender el período planificado.

En el primero de los supuestos, la armonización solo es uno de los posibles caminos para llegar a ese fin. Se pueden explorar otras vías de reducir la complejidad del problema reduciendo el número de clases de equivalencia. Un método idóneo para realizar esto es mediante técnicas de clustering como, por ejemplo, k-medias. El problema final es más cercano al de partida y se obtiene con más facilidad, pero se pierde la propiedad de que los períodos sean armónicos.

En el caso de que sea deseable que aparezcan instantes críticos en la planificación, las técnicas de clustering no son las más apropiadas por que el resultado no es armónico, como ya se ha comentado anteriormente. Al igual que es posible sustituir el algoritmo de programación de las tareas por backtracking con un algoritmo que usa recocido, se podría intentar adaptar el proceso de recocido para obtener un problema armónico lo más similar posible al original y que sea resoluble.

7.2.4. Mejoras sobre el proceso de recocido

El algoritmo de recocido aplicado o uno muy similar ha sido paralelizado de forma satisfactoria mediante grid-computing [5]. Se podría utilizar para reducir los tiempos de cómputo necesarios al tiempo que se mejora la precisión de la solución.

Finalmente, habría que considerar otras medidas de la energía interna y comprobar si los resultados mejoran, empeoran o permanecen igual.

Apéndice A. Presupuesto

El presente trabajo de investigación lleva asociado un trabajo. Es el producto resultante de aplicar un esfuerzo profesional a un problema. Si este trabajo se hubiera producido como resultado de la prestación de servicios profesionales, habría que considerar la parte económica del mismo.

En el transcurso de una transacción comercial dentro del entorno de las tecnologías de la información, así como en general en el resto de sectores económicos, es vital saber cuantificar la inversión monetaria que es preciso realizar para obtener el producto final. En este caso, los gastos acarreados son, básicamente, gastos debidos a la mano de obra y gastos debidos al material utilizado.

A.1. Mano de obra

El proyecto implica el desempeño de distintos roles a lo largo de su realización. Los roles concretos que se pueden considerar que han intervenido pueden variar según como se dividan las tareas. Cada rol lleva asociado un coste por hora distinto. Se va a considerar, en este caso, que los roles participantes son los tres que se muestran a continuación.

- **Jefe de proyecto.** Realiza la planificación del proyecto, prioriza las tareas, gestiona los recursos y determina el alcance.
- **Analista.** Diseña la arquitectura del sistema, ratifica que la implementación corresponde con el diseño y determina los parámetros de eficiencia.
- **Programador.** Prepara el entorno de desarrollo y el de documentación. Realiza la implementación de los sistemas considerados y los métodos de medición.

A.1.1. Tareas

El proyecto se va a considerar dividido en cuatro fases, en cada una de las cuales participan uno o más roles de forma coordinada para llegar al resultado deseado.

Cada fase del desarrollo consiste en un conjunto de tareas. Cada tarea está asignada a uno de los roles que es el responsable de realizarla.

Tabla A.1. Fase de Análisis

Tarea	**Trabajo (horas)**	**Rol**	**Coste (€)**
Investigación bibliográfica	52	Analista	3.120
Determinar el alcance	12	Analista	720
Crear algoritmos	80	Analista	4.800
Gestión del análisis	12	Jefe de proyecto	1.080

Tabla A.2. Fase de Desarrollo

Tarea	**Trabajo (horas)**	**Rol**	**Coste (€)**
Establecer entorno desarrollo	12	Programador	480
Codificación	112	Programador	4.480
Pruebas unitarias	40	Programador	1.600
Gestión del desarrollo	16	Jefe de proyecto	1.440

Tabla A.3. Fase de Pruebas

Tarea	Trabajo (horas)	Rol	Coste (€)
Determinar parámetros de ejecución	16	Analista	960
Soporte a la ejecución	32	Programador	1.280
Ejecución y medición	216	Programador	8.640
Representación gráfica	28	Programador	1.120
Análisis de los resultados	16	Analista	960
Gestión de las pruebas	28	Jefe de proyecto	2.520

Tabla A.4. Fase de Documentación

Tarea	Trabajo (horas)	Rol	Coste (€)
Establecer entorno documentación	40	Programador	1.600
Realizar documento	184	Analista	11.040
Gestión de la documentación	24	Jefe de proyecto	2.160

Tabla A.5. Resumen por fases

Fase	Trabajo (horas)	Coste (€)
Análisis	156	9.720

Fase	Trabajo (horas)	Coste (€)
Desarrollo	212	9.280
Pruebas	304	14.200
Documentación	248	14.800

El coste total de la mano de obra a lo largo del proyecto es de 48.000€ empleados en 920 horas de trabajo.

A.2. Materiales

El material tangible que se ha utilizado a lo largo de este proyecto es un ordenador de sobremesa valorado en 1.600€. Se considera que la vida útil del ordenador abarca cinco años. Suponiendo que en un año hay 298 días laborables y que se consideran jornadas de 4 horas, se puede deducir que el total de horas de la vida útil del ordenador es de 5.960 horas. Por tanto, el gasto imputable al proyecto en maquinaria es:

$$\text{Coste} = \frac{\text{Valor} \cdot \text{duración}}{\text{vida}} = \frac{1.600€ \cdot 920\text{horas}}{5.960\text{horas}} \simeq 247€$$

Ecuación A.1. Coste imputable por maquinaria

Aparte de la maquinaria, hay que considerar el coste asociado al software. En la tabla que aparece a continuación se detalla el coste de dicho software.

Tabla A.6. Coste debido al software

Ítem	Coste imputable
GNU/Linux	0
IDE Eclipse	0
Sun Java SDK 1.5	0
Docbook	0

Ítem	Coste imputable
Labplot	0
JEuclid	0
Jaxe	0

El gasto en material, por tanto, suma 247€.

A.3. Resumen

El total del gasto del proyecto es el que se ve en la tabla siguiente:

Tabla A.7. Coste total del proyecto

Categoría	Coste
Mano de obra	48.000€
Material	247€
Total	*48.247€*

Acrónimos

Arcos	Arquitectura de computadores y sistemas.
RAIL	Reliability centered management Aproach for the Infraestructure and Logistics of railway operation.
RCM	Reliability Centered Maintenance (Mantenimiento Centrado en la Fiabilidad).
Adif	Administrador de Infraestructuras Ferroviarias.
IA	Inteligencia artificial.
BT	Bloqueo telefónico.
BEM	Bloqueo eléctrico manual.
CCR	Control de circulación por radio.
BA	Bloqueo automático.
BAU	Bloqueo automático de vía única.
BAD	Bloqueo automático de vía doble.
BAB	Bloqueo automático banalizado.
BLA	Bloqueo de liberación automática.
BCA	Bloqueo de control automático.
BSL	Bloqueo de señalización lateral.

Bibliografía

[1] Adif. 2006. *Declaración sobre la red 2006*. http://www.infraestructuras-ferroviarias.com/empresa/pdf/declarared/Libro_RED_2006.pdf.

[2] Alejandro Carrasco Yelmo. *Proyecto de fin de carrera: Herramienta para el análisis RCM aplicado al ferrocarril*. Universidad Carlos III de Madrid. 2006.

[3] John Moubray. 1997. *Reliability-Centered Maintenance*. ISBN: 0831130784. Industrial Press.

[4] S Kirkpatrick, C.D Gelatt, y M.P Vecchi. 13 Mayo 1983. *Optimization by simulated annealing*. Science. Vol.220. 4598. pp.672-679.

[5] Jesús Sánchez, Pablo Ruiz, y Pilar Moreno. 2000. *Ejecución cooperativa de algoritmos de recocido utilizando grid computing*. XI Jornadas de Paralelismo.

[6] Hermann Kopetz. 1997. *Real-time systems. Design principles for distributed embedded applications*. Ed. Kluwer Academic publishers.

[7] S.C Cheng. 1987. *Scheduling algorithms for hard real-time systems - A brief Survey*. J.A Stankovic. Ed. IEEE Press.

[8] M.R Garey y D.S Johnson. 1975. *Complexity results for multiprocessor scheduling under resource constraints*. SIAM journal of computing. Vol.4 (4). pp.397-411.

[9] Alan Burns y Andy Wellings. 2003. *Sistemas de tiempo real y lenguajes de programación*. ISBN: 84-7829-058-3. Pearson educación.

[10] Francisco Javier González Fernández. 2005. *Teoría y práctica del mantenimiento industrial avanzado*. ISBN: 978-84-96169-49-4. FC Editorial.