Pedro Sánchez Martín
Ingeniero del ICAI (1993) y Doctor en Ingeniería Industrial por la UPCO (1998). Profesor del Departamento de Organización Industrial del ICAI. Miembro de IIT. Profesor del Master de Logística Integral del Instituto de Postgrado y Formación Continua de la UPCO.

Programación de tareas, un reto diario en la empresa

La programación de tareas es una labor cotidiana que no solo un ingeniero realiza con el fin de organizar cómo realizar muchas actividades relacionadas entre sí. En este artículo se explica la naturaleza de la programación de tareas con varios ejemplos, su aplicación en distintos ámbitos empresariales, así como una clasificación de métodos de resolución y un caso concreto de aplicación. Para el lector más interesado se ha creado un link que permite el uso de una herramienta informática de programación www.iit.upco.es\scheduling

Santiago López de Haro
Ingeniero Industrial del ICAI (2002). Experiencia profesional en los sectores eléctrico, ferroviario y de automoción. Actualmente es Investigador en Formación del grupo de Planificación y Operación del Instituto de Investigación Tecnológica.

Introducción

La escritura de un artículo técnico como éste puede ser un buen ejemplo de programación de tareas. Santiago y Pedro han llevado a cabo investigaciones y proyectos en este área de conocimiento y por ello se han decidido a escribir este artículo. Antes de empezar se han planteado varios interrogantes sobre el proceso de la escritura del artículo:

- ¿Hay fecha límite para su entrega a la revista?
- ¿Cuáles son las tareas que han de llevarse a cabo?
- ¿Quién ha de realizar cada una de dichas tareas?
- ¿Existe una secuencia lógica para la escritura del artículo?
- ¿De cuánto tiempo dispone cada autor?
- ¿Ambos autores pueden hacer todas las tareas y con igual eficiencia?
- ¿Los autores prefieren realizar las tareas del artículo lo más seguido posible, o por el contrario, quieren tener intervalos de tiempo intermedios no dedicados al artículo?
- ¿Qué modificaciones habrá que hacer al artículo para que éste sea aceptado por los editores de la revista?

Las respuestas a algunos de estos interrogantes provienen de condicionantes externos como puede ser la fecha límite de recepción que la revista establezca. Sin embargo, otras preguntas tienen respuesta tras el análisis del proceso de escritura de este artículo. A continuación se indican las tareas que han de llevarse a cabo:

1. Elaboración de un ejemplo introductorio.
2. Escribir un apartado sobre aplicaciones posibles.
3. Escribir un apartado explicando procedimientos de resolución.
4. Elaboración y explicación de un problema ejemplo.
5. Describir la resolución del problema ejemplo.
6. Elaboración de gráficos:
- Gráficos del ejemplo introductorio.
- Gráficos de los procedimientos de resolución.
- Gráficos sobre la resolución del problema ejemplo.

7. Bibliografía.

Ciertamente el reparto de las tareas entre los autores depende de su disponibilidad, de su eficiencia y de su capacitación para realizar las distintas tareas. Se reparten las tareas de forma que Pedro se encarga de completar las tres primeras tareas y la bibliografía, mientras que Santiago se dedica a las tareas restantes.

Parece lógico establecer que los gráficos se elaboren después de haberse completado la escritura del apartado donde van a estar insertados. Por este motivo, Santiago ha de esperar a que Pedro termine de escribir ciertos apartados del artículo con el fin de elaborar gráficos acordes a su contenido. La elaboración de algunos apartados del artículo no tiene dependencia alguna, tal y como ocurre por ejemplo con la escritura de la bibliografía. Esta flexibilidad en la secuencia temporal de las tareas permite reducir tiempos de espera de un autor mientras el otro termina una determinada tarea.

En el gráfico Gantt que se muestra en la Figura 1 se muestra un reparto posible de tareas entre los dos autores. La duración de la escritura combinada del artículo es de 19 horas. Durante las cuales Pedro trabaja 13 horas y Santiago 15 horas. Con la secuencia establecida para Pedro, Santiago ha de esperar a que Pedro complete la tarea 3 ("Procedimientos de resolución") antes de elaborar los gráficos de dicha sección, 6b, y por ello ha de esperar 4 horas.

Considerando que la tarea 5 ("Resolución del problema ejemplo") no requiere haber realizado antes la tarea 6b ("Gráficos de los procedimientos de resolución"), su orden de realización se puede intercambiar, tal y como se indica en la Figura 2. Se comprueba como el tiempo de espera de Santiago

Figura 1

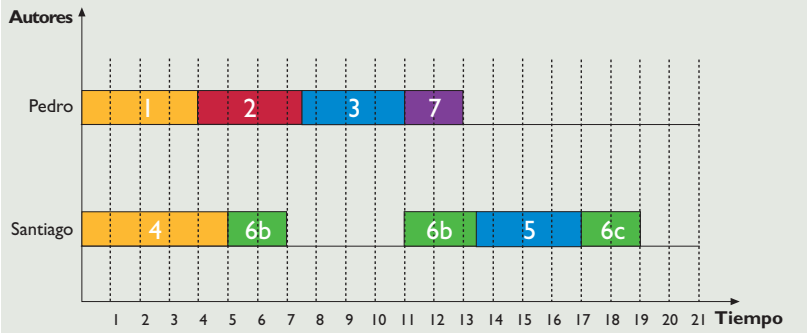
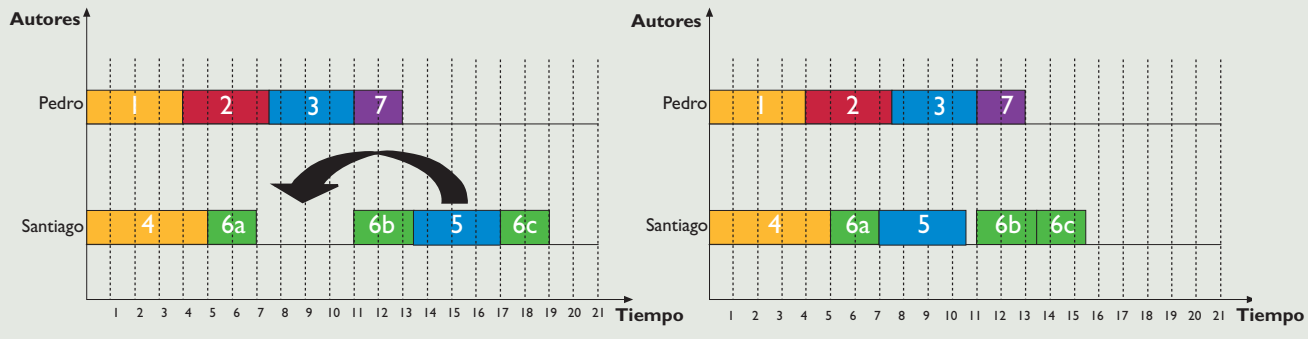


Figura 2



se reduce a media hora y por ello el tiempo de redacción del artículo resulta ser de 15 horas y media. El resultado se aproximó bastante a lo que sucedió en la realidad, si bien la revisión final del contenido del artículo alargó la duración del proceso de confección.

En un ámbito más cercano, todo el mundo se organiza las tareas que ha de hacer durante el día o la semana. Se trata de un trabajo sencillo, basta con apuntar las cosas a hacer según nos vamos acordando de ellas. No obstante, en el ámbito empresarial resulta más complicado conocer cuál es el orden con el cual llevarlas a cabo según distintas prioridades tales como fechas de entrega, inventario en curso y otras.

Aplicaciones

La programación de tareas apoya a las principales áreas funcionales de una empresa:

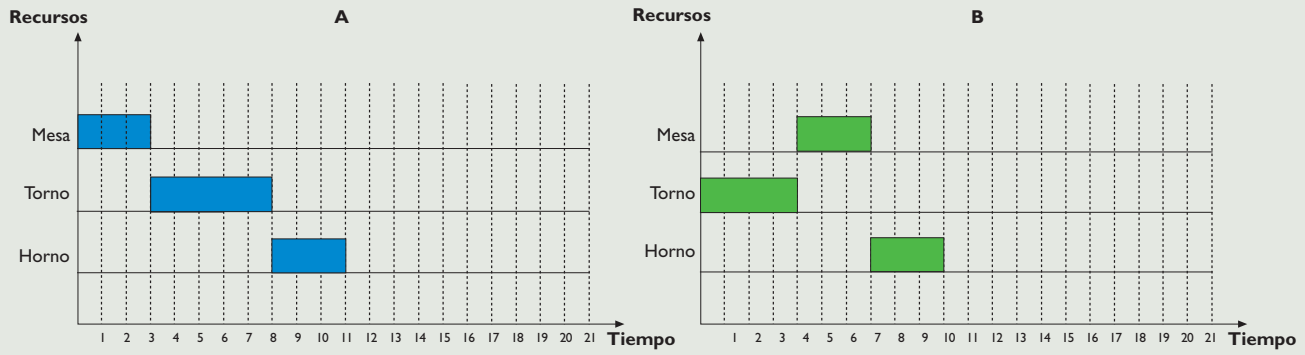
- El departamento de Producción necesita conocer la secuencia real de productos a fabricar en el día. Por ello, su personal ha de consultar la programación de tareas que resulta tras la planificación maestra de los diferentes productos a fabricar (MRP).
- El departamento de Contabilidad, a partir de la información de la programación y de las fechas de entrega a los clientes prevé el ingreso y coste de cada trabajo y además realiza un análisis del "cash-flow" de la empresa.
- El departamento de marketing puede establecer medidas de la eficiencia de la programación para determinar si los tiempos de cumplimentación de los pedidos están aportando una ventaja competitiva y si las entregas se están realizando a tiempo. El conocimiento de los tiempos de entrega de cada pedido permite al departamento de Marketing ofrecer tiempos de entrega más ajustados.
- El departamento de Compras puede realizar el seguimiento de los pedidos a lo largo del proceso productivo para solicitar a los proveedores que adelanten las fechas de entrega de las piezas o por el contrario, que las retrasen.
- Un sistema de información se puede encargar de almacenar en base de datos el tiempo de comienzo de cada trabajo así como el enrutamiento de los mismos por cada uno de los recursos disponibles. También se puede encargar de realizar el seguimiento del movimiento de cada producto a través del proceso y de modificarlo si fuese conveniente.

Las empresas diseñadas para satisfacer pedidos de lotes de producto o un volumen elevado de servicios poseen unos procesos productivos altamente repetitivos. En estos casos la programación de tareas se suele utilizar para determinar la secuencia productiva de lotes de un mismo producto o servicio que permita cumplir con las fechas de entrega del pedido. También en un contexto de fabricación, la programación de tareas se puede utilizar para equilibrar la carga de trabajo entre los operarios asignándoles convenientemente las operaciones constitutivas del proceso productivo.

Las empresas diseñadas para proveer al cliente de productos o servicios personalizados llevan a cabo procesos productivos específicos, tal y como ocurre por ejemplo en los servicios técnicos de reparación de equipos. La programación de las tareas en este tipo de empresas ha de tener en cuenta la limitación de capacidad del número de máquinas y operarios disponibles entre otros condicionantes.

La programación de tareas resulta también necesaria en actividades empresariales tales como el transporte de viajeros y mercancías. En estos entornos se necesitan herramientas capaces de programar temporalmente los viajes conforme a las necesidades de la planificación de servicios con las particularidades propias de medios de transporte tan diferentes como el transporte por carretera o ferrocarril.

Figura 3. Secuencias técnicas de los trabajos A y B



En los últimos años, se ha avanzado notablemente en los procedimientos de resolución del problema de programación de tareas. Como fruto de ello existen en el mercado herramientas informáticas para la programación de corto plazo, las cuales se han integrado en plataformas de planificación de la cadena de suministros (Enterprise Resource Planning, ERP). Estas plataformas utilizan alguno de los procedimientos de resolución comentados más adelante en este artículo.

Los autores del presente artículo han desarrollado parte de su labor profesional investigando y colaborando con varias empresas en este campo. Recientemente, se ha puesto a disposición pública una herramienta informática de programación de tareas limitada a un dimensionamiento máximo de 15 x 15 tareas y un tiempo límite de respuesta de 5 minutos. Dicha herramienta informática está disponible en el link www.iit.upco.es/scheduling.

El problema de la mesa, el torno y el horno

A menudo se producen conflictos entre diferentes trabajos que emplean los mismos recursos debido a que tan solo pueden procesar un trabajo simultáneamente. La Figura 3 muestra un ejemplo de dos trabajos (A y B) compuestos por tres operaciones cada uno que han de ejecutarse empleando los mismos recursos: la mesa, el torno y el horno. La Figura 4 muestra la programación resultante de la superposición de estas secuencias. Como se puede observar, si los recursos no pueden procesar más de un trabajo a la vez, se producen conflictos en el torno y el horno.

La resolución de la programación de tareas consiste en decidir, para cada conflicto entre tareas y recursos, cuál de las tareas

Figura 4. Programación simultánea de ambos trabajos

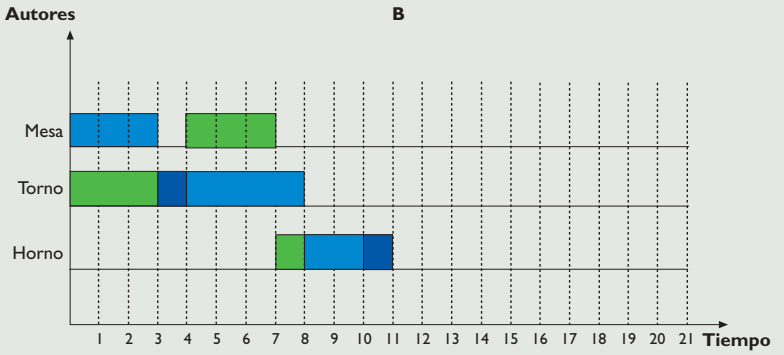
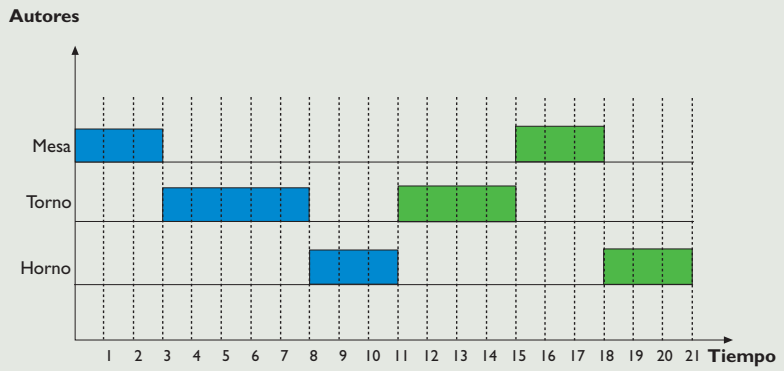
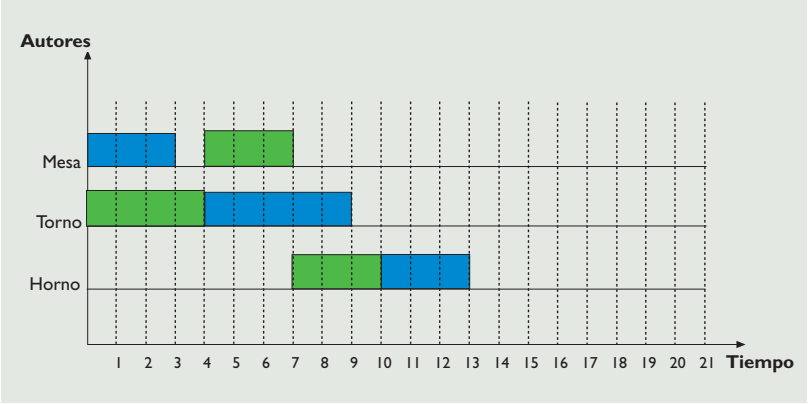


Figura 5. Solución óptima en retraso máximo



debe ejecutarse antes. Normalmente, el objetivo se establece definiendo el orden de la asignación de tareas por recurso que minimice el tiempo de finalización del conjunto de trabajos, también conocido como *makespan*. Existen otros objetivos posibles tales como el máximo retraso en la entrega, el retraso medio, o el tiempo medio de finalización por trabajo. Así, se supone que el

Figura 6. Solución óptima en makespan



trabajo A tiene que entregarse en el tiempo $t=11$ y el trabajo B en $t=21$. La solución dada por la Figura 5, elaborada esperando a que se ejecuten –todas las tareas del trabajo A antes de iniciar las del trabajo B, sería la mejor opción posible desde el punto de vista de la minimización del máximo retraso, y la solución mostrada en la Figura 6 sería la solución de mínimo *makespan*.

Procedimientos de resolución

La naturaleza de los problemas de programación de tareas es combinatorial y, por ello, el tiempo necesario para encontrar la solución óptima crece exponencialmente con el número de tareas consideradas. De hecho, el problema mt10, un caso particular propuesto por Muth & Thompson en 1959 en el que el objetivo es la minimización del makespan para 10 trabajos con 10 tareas en máquinas diferentes, no pudo ser resuelto óptimamente hasta 20 años después.

Un procedimiento tan sobradamente utilizado en otros contextos de la producción como es la programación matemática [1] no resulta apropiado en este ámbito ya que su formulación lineal entera mixta da lugar a tiempos de ejecución elevados en problemas de tamaño real, lo cual la hace poco aconsejable.

Otros procedimientos disponibles son los heurísticos [2]. Éstos se basan en reglas lógicas extraídas de la experiencia del usuario, para generar soluciones válidas en tiempos razonables. Aunque generalmente las soluciones así generadas son bastantes buenas, ningún heurístico puede garantizar la obtención del óptimo para el problema considerado. Además la mayoría de estos algoritmos son difícilmente adaptables a problemas ligeramente diferentes a aquéllos para los que fueron concebidos.

Los algoritmos heurísticos que son fácilmente generalizables a varios tipos de problemas, como los algoritmos genéticos, el recocido simulado o la búsqueda tabú, son denominados algoritmos metaheurísticos. La mayoría de ellos emulan procesos que ocurren en la naturaleza. Así, los algoritmos genéticos [3] imitan el procedimiento de la selección natural sobre generaciones sucesivas de individuos representativos de posibles soluciones al problema considerado. Los nuevos individuos se crean cruzando parejas que son seleccionadas dando preferencia a aquéllas soluciones con mejor valor de la función objetivo (individuos más adaptados al entorno).

Por otro lado el recocido simulado [4] emula la tendencia de los materiales metálicos a alcanzar estructuras de mínima energía a medida que disminuye la temperatura. Este algoritmo parte de una solución inicial generada de forma aleatoria o mediante algún heurístico. La solución va siendo modificada sucesivamente mediante pequeños desplazamientos seleccionados de entre el conjunto de soluciones parecidas a una dada, denominado "vecindario". Un desplazamiento que empeore la solución anterior es aceptado con una probabilidad que desciende con la temperatura. De este modo, a alta temperatura prácticamente cualquier desplazamiento puede ser aceptado. Sin embargo, a baja temperatura sólo los desplazamientos que mejoran la función objetivo lo son, lo que provoca que la solución se estabilice en las regiones más cercanas al óptimo.

La búsqueda tabú [5], al igual que el reconocido simulado, se basa en una solución que es modificada sucesivamente mediante desplazamientos que, aunque no siempre mejoran la función objetivo, tienden a desplazar a la solución a las regiones más cercanas al óptimo. Está basado en una memoria de corto plazo llamada lista tabú que contiene los últimos desplazamientos realizados. En cada iteración se evalúan todos los posibles desplazamientos en el vecindario de la solución actual y se selecciona al mejor de entre aquellos que no se encuentran en la lista tabú. De este modo se evita que el algoritmo se quede atascado iterando en bucles cerrados. Descrito de otra forma, en cada iteración se evalúa el vecindario de la solución actual y se clasifican los desplazamientos en desplazamientos "favorables", desplazamientos "no favorables no

tabú", desplazamientos "favorables tabú" y desplazamientos "no favorables tabú", siendo éste el orden de prioridad en la selección de los mismos.

Un procedimiento más reciente para la resolución de los problemas de programación de tareas es la programación de restricciones [6]. Este algoritmo es similar al procedimiento de "ramificación y poda" (Branch & Bound) en cuanto a que genera un árbol de búsqueda que recoge todo el espacio de soluciones posibles y que, a medida que evoluciona la exploración, las ramas que han demostrado ser menos prometedoras van siendo eliminadas (podadas) del árbol. Pero se diferencia de éste que emplea las restricciones del problema para crear condiciones adicionales de poda de ramas. La ventaja fundamental de este algoritmo es que garantiza que la solución obtenida es el óptimo global del problema. En [7] se ha hecho un estudio comparativo experimental de la eficiencia de distintos metaheurísticos y del método de programación de restricciones.

Resolución del problema ejemplo

El problema ejemplo de la mesa, el torno y el horno se va a resolver mediante un algoritmo metaheurístico sencillo llamado "búsqueda avariciosa" que, al igual que la "búsqueda tabú" o el "reconocido simulado", puede ser englobado en el conjunto o el "recocido simulado" en los denominados procedimientos de "búsqueda local". La búsqueda local está basada en la suposición de que es posible encontrar una secuencia de soluciones entre la solución inicial y la óptima tal que cada una de ellas es ligeramente diferente a la inmediatamente anterior.

Un concepto asociado a la búsqueda local es la descripción del conjunto de soluciones parecidas a una dada, denominado "vecindario". Para este problema se ha optado por definir que dos soluciones son "vecinas" si una puede ser obtenida a partir de la otra intercambiando el orden entre dos tareas que pertenecen al mismo recurso y que son consecutivas en el camino crítico[1].

La "búsqueda avariciosa", parte de una solución inicial generada aleatoriamente o mediante un algoritmo heurístico y evalúa todas las soluciones que se encuentran en su vecindario, escogiendo la más prometedora de entre las que mejoren el valor de la función objetivo. La repetición de este proceso termina en un óptimo del problema, ya sea local o global.

Figura 7

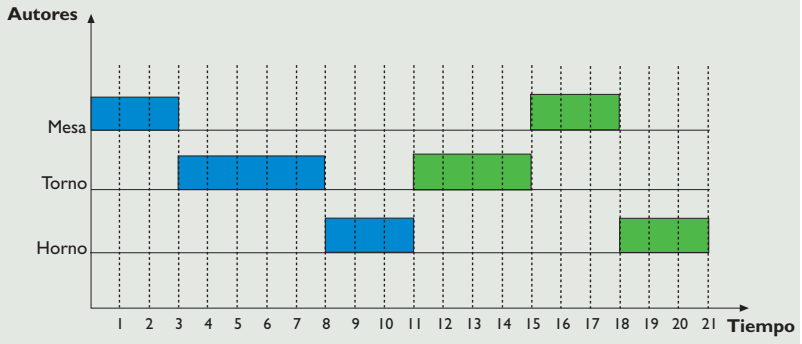
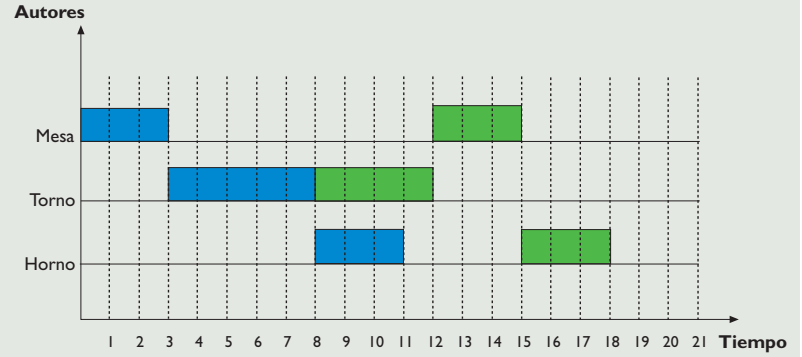


Figura 8



Las Figuras 8, 9 y 10 muestran su aplicación al problema de la minimización del *makespan* de los trabajos A y B en la mesa, el torno y el horno. Parte de la solución descrita por la Figura 7, que ha sido generada realizando todas las tareas del trabajo A antes que las tareas del trabajo B.

En un primer paso, el algoritmo adelantaría le ejecución del trabajo B hasta que las tareas que se realizan en el torno sean adyacentes (ver figura 8) En adelante, siempre que se pueda adelantar la ejecución de las tareas en un determinado recurso sin alterar el orden de las mismas se hará automáticamente.

La primera iteración del algoritmo comienza por evaluar las soluciones vecinas. En este caso, atendiendo a la definición de ve-

(1) Camino crítico: Conjunto de tareas cuya realización consecutiva establece el tiempo total de finalización de los trabajos programados.

Figura 9

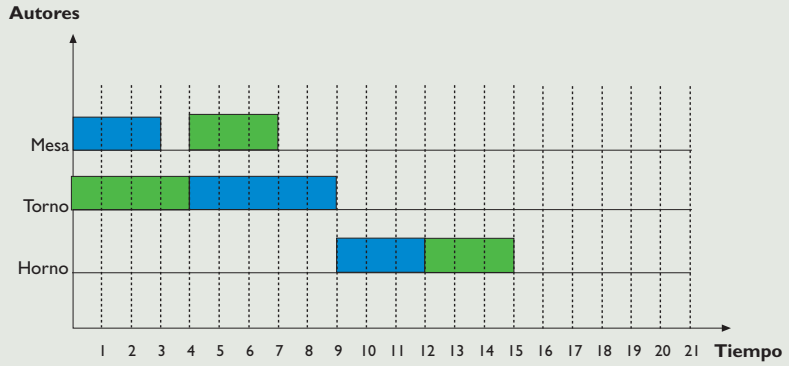
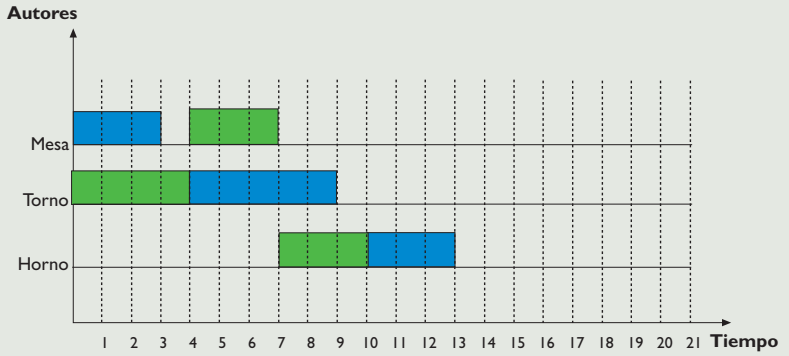


Figura 10



cindario basada en el camino crítico, tan solo se puede alterar el orden entre las dos tareas que se ejecutan en el torno. La Figura 9 se ha obtenido realizando esta operación y actualizando el tiempo de ejecución de todas las tareas afectadas. Esto implica adelantar la ejecución del trabajo B en la mesa y en el horno, a la vez que se retrasa una unidad de tiempo la ejecución del trabajo A en el horno.

Como se puede observar, en una única iteración de este algoritmo se ha conseguido reducir el makespan de 21 unidades de tiempo hasta 15. Sin embargo, esta solución todavía se puede mejorar. En este caso, existen dos transformaciones posibles. La primera es el intercambio del orden entre las tareas que se ejecutan en el torno, lo cual provocaría volver a la situación anterior (Figura 8) y, por lo tanto, no se escoge. La segunda es el intercambio del orden de las tareas que se ejecutan en el horno. La Figura 10 representa la solución generada al realizar este último desplazamiento obteniéndose un makespan óptimo de 13 unidades de tiempo.

El reto de los programas de tamaño real

Naturalmente, el problema considerado anteriormente es poco realista, a diferencia de los que existen en las programaciones semanales de centros de trabajo donde pueden existir cientos de tareas y decenas de recursos. En estos problemas lo más probable es que el procedimiento de búsqueda avariciosa converja prematuramente a soluciones mejores que cualquiera de sus vecinas pero peores que el óptimo del problema. Para disminuir la probabilidad de que esto ocurra se emplean procedimientos que permitan algunos desplazamientos desfavorables, como el recocido simulado o la búsqueda tabú.

Además, en los problemas de tamaño real se pueden encontrar otras características como la existencia de recursos de capacidad múltiple, tiempos de preparación de un recurso dependientes de la secuencia de las tareas en el mismo y muchas otras características propias de procesos empresariales en los que la programación de tareas puede mejorar notablemente la eficiencia de sus procesos. Por ello, una labor como la programación de tareas además de ser vital en muchas empresas se puede considerar como un reto diario al cual se debe dar la mejor respuesta posible. ▮

Notas

[1] Wagner, H.M.: An integrer programming model for machine scheduling. Nav. Res. Logist Q., 1999. 6: p. 131-140.

[2] Adams, J., Balas E. y Zawack D.: The Shifting Bottleneck Procedure for Job Shop Scheduling. Manager Science, 1988.

[3] Yamanda T. y Nakano, R.: A Genetic Algorithm Applicable to Large-Scale Job-Shop Problems. Parallel Problem Solving from Nature, 1992. 2: p. 281-290.

[4] Matsuo, H., Chang Juck Suh y Sullivan, R.S.: A controlled search simulated annealing metho for the single machine weighted tardiness problem. Annals of Operations Research, 1989. 21(1-4): p. 85-108

[5] Nowicki, E. y Smutnicki, C.: A fast taboo search algorithm for the job shop problem. Management Science, 1996.

[6] ILOG, Solver 6.0, Scheduler 6.0, 1998.

[7] López de Haro, S., Sánchez Martín, P. y Conde Collado, J.: Secuenciación de tareas mediante metaheurísticos. VIII Congreso de Ingeniería de Organización, Leganés, 2004, p. 1021-1031.