VIII Congreso de Ingeniería de Organización
Leganés, 9 y 10 de septiembre de 2004

Asignación de recursos a actividades variables

José Manuel García Sánchez, Ricardo Galán de Vega, Jesús Racero Moreno

Dpto. Organización Industrial y Gestión de Empresas. Escuela Superior de Ingenieros. Universidad de Sevilla. Camino de los Descubrimientos, s/n 41092 Sevilla. jmgs@esi.us.es, rdevega@ us.es , jrm@esi.us.es

Resumen

En este trabajo nos ocupamos del problema de la asignación de recursos a actividades variables, conocido como Variable Job Scheduling Problem (VSP). VSP se caracteriza como el problema de planificar, sobre un conjunto de máquinas en paralelo, un conjunto de trabajos no interrumpibles, caracterizados cada uno de ellos por un tiempo de procesamiento y un intervalo dentro del cual debe ser procesado el trabajo. El objetivo considerado en el problema es maximizar el numero de trabajos procesados. Para la resolución del problema se propone un enfoque heurístico basado en búsqueda tabú. La comparación con los resultados de otros métodos heurísticos propuestos para el problema sobre una batería de problemas generada aleatoriamente, pone de manifiesto que la búsqueda tabú encuentra soluciones de mejor calidad utilizando menor tiempo.

Palabras clave: Asignación, Trabajos variables, Búsqueda Tabú.

1. Introducción

En ciertos entornos industriales y de servicios, las tareas a desarrollar disponen de poca flexibilidad de horarios, teniendo que ser realizadas dentro de determinados márgenes o intervalos de tiempo. Este sería el caso del problema que nos ocupa en este trabajo.

En términos de la teoría de la programación de trabajos en máquinas (*scheduling*), la asignación de recursos a actividades variables se corresponde con el *Variable Job Scheduling Problem* (VSP). VSP se caracteriza como el problema de planificar, sobre un conjunto de máquinas en paralelo, un conjunto de trabajos, no interrumpibles, caracterizados cada uno de ellos por un tiempo de procesamiento, un intervalo dentro del cual debe ser procesado el trabajo y una clase de trabajo. Las máquinas se caracterizan por pertenecer a una clase de máquina, de forma que cada clase de máquina puede únicamente procesar trabajos de un subconjunto de clases de trabajos.

Dos objetivos se definen para el problema VSP:

- Minimizar el número de máquinas necesarias para procesar todos los trabajos

- Para un número fijo de máquinas, maximizar el numero ponderado de trabajos procesados, asumiendo un peso o valor para cada trabajo.

El primero de los objetivos ha sido considerado por Gertsbakh y Stern (1978) para el caso de una sola clase de máquina y trabajo, mostrando que VSP es NP-Completo en todos los casos. El segundo objetivo es considerado en Gabrel(1995), asumiendo el mismo valor para todos

los trabajos. Aquí VSP es modelado como un problema de cálculo del máximo conjunto independiente de nodos en un grafo. Sobre dicho problema se proponen 3 enfoques heurísticos para su resolución.

Sin embargo, la versión del problema más ampliamente estudiada ha sido aquélla que considera cada intervalo de tiempo como un punto, es decir, cada trabajo posee un instante fijo de inicio y finalización. Este problema se denomina *Fixed Job Scheduling Problem* (FSP). FSP ha sido considerado por diversos autores, los cuales consideran tanto una como varias clases de trabajos y máquinas. Considerando una única clase, el problema puede ser resuelto en tiempo polinomial mediante un cálculo de flujo a coste mínimo (Kroon *et al*, 1995).

FSP y VSP juegan un importante papel en aplicaciones prácticas de asignación de recursos, como pueden la asignación de puertas a vuelos en aeropuertos, la planificación de habitaciones de hotel, la asignación de conductores a líneas de autobuses, etc.

Para este trabajo nos centramos en el siguiente escenario del problema VSP: Consideramos el problema de encontrar una programación óptima para un conjunto de n trabajos, denotados por $J = \{1 \ldots n\}$ sobre un conjunto de m máquinas idénticas en paralelo. Cada trabajo posee un tiempo de proceso t_i y una ventana temporal $[a_i, b_i]$ para el instante de comienzo (Figura 1). El objetivo es encontrar el mayor número de trabajos que pueden ser procesados.

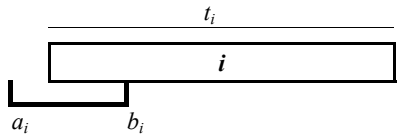

Figura 1. Datos asociados al trabajo i

Para la resolución del problema se propone un enfoque heurístico basado en búsqueda tabú. Dicho enfoque será analizado sobre una batería de problemas generada aleatoriamente, y los resultados serán comparados con las heurísticas propuestas en Gabrel (1995).

2. Un enfoque basado en búsqueda tabú

En esta sección se describe el algoritmo de búsqueda tabú propuesto para resolver el problema VSP. El método de búsqueda tabú propuesto por Glover (1997) es un procedimiento heurístico que explora el espacio de soluciones a través de movimientos repetidos desde una solución a la mejor de sus soluciones vecinas, con ciertas restricciones en la elección de la solución de cada iteración del proceso. Búsqueda tabú ha sido utilizado con éxito para resolver complejos problemas de optimización combinatoria, particularmente en el campo de la programación de trabajos. Los elementos del algoritmo se describen a continuación:

Solución Inicial: Para obtener una solución inicial consideramos el problema FSP. Si asignamos un instante de inicio aleatorio para cada pedido, la solución óptima de este problema se obtiene resolviendo un problema de flujo a coste mínimo sobre un grafo construido mediante la estructura descrita en Kroon *et al*(1995).

Movimientos y Estructura de la Vecindad: Dada una solución S, denotamos por N(S) al conjunto de todas las soluciones admisibles que pueden ser obtenidas desde S usando uno de los siguientes movimientos:

- Intercambio: Un trabajo $i \in S$ se sustituye por otro trabajo j que no está en la solución actual y se solapa con el pedido i en algún instante de tiempo. Para la asignación del trabajo j se comprueban todos los instantes de comienzo, desde el instante más temprano en la ventana temporal, hasta encontrar un instante admisible con respecto al resto de los pedidos.

  El mejor movimiento de intercambio es aquel que posee la menor diferencia entre el tiempo de proceso del trabajo i, $i \in S$, y el trabajo j, $j \notin S$.

- Extracción: Movimiento de extraer un pedido de la solución. Este movimiento es debido a la definición de lista tabú realizada en la sección 3.C. Obviamente, este movimiento lleva a soluciones de peor calidad pero es útil como herramienta de diversificación.

- Inserción: En ocasiones, después de aceptar un movimiento de intercambio o extracción, otros pedidos podrían ser introducidos en el conjunto solución. Este movimiento siempre produce una mejora en la función objetivo con respecto a la solución actual y, por ello, es el primer tipo de movimiento considerado en cada iteración del algoritmo.

  Cuando no existe ningún movimiento de inserción admisible, se acude al movimiento de intercambio para generar soluciones vecinas. Si todos los trabajos que no pertenecen a la solución son tabú, entonces se realiza un movimiento de extracción.

En la Figura 2 se ilustran dos iteraciones del algoritmo, asumiendo que inicialmente no existe pedidos tabú.

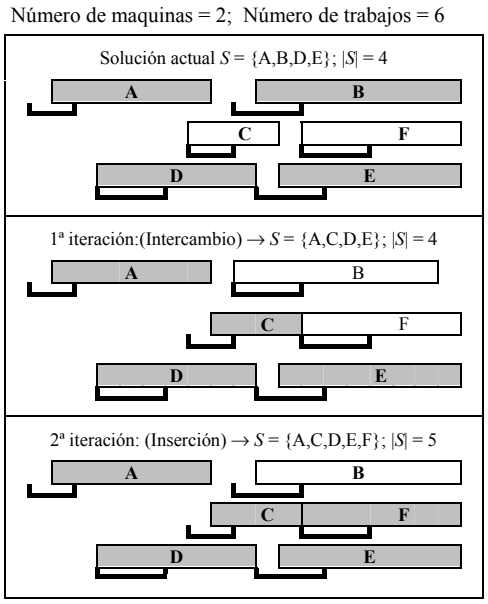

Figura 2. Ejemplo de movimiento de inserción e intercambio

Lista Tabú: Cuando se reemplaza o elimina un trabajo de la solución, introducir este trabajo de nuevo en la solución se considera tabú durante un número determinado de iteraciones. En la búsqueda tabú, la lista tabú puede ser fija o variable. Glover *et al*(1993) sugiere el uso de listas tabú con tamaño variable. En nuestro método implementamos una estrategia de tamaño variable, donde el tamaño se modifica, con incrementos y decrementos, con objeto de ajustar el número de iteraciones entre la aceptación de dos movimientos de extraer un trabajo de la

solución. Un movimiento de extracción es únicamente aceptado cuando todos los trabajos que no están en la solución son tabú. Por tanto, necesitamos definir una frecuencia de incremento $frec_{inc}$, una cantidad a incrementar I y una cantidad a decrementar D, para la lista tabú. No necesitamos definir una frecuencia de decremento porque el tamaño de la lista tabú se reduce automáticamente cada vez que se acepta un movimiento de extracción. Inicialmente, el tamaño de la lista se incrementa en cada iteración hasta que todos los trabajos que están fuera de la solución sean tabú.

Otro aspecto a considerar en el algoritmo para mejorar su funcionamiento y permitir una mejor exploración del espacio de soluciones, es la de evitar que siempre sean los mismos trabajos los que salgan de la solución cuando sea necesario aplicar un movimiento de extracción. Esto suele ocurrir generalmente, puesto que el algoritmo escoge siempre el que posee un tiempo de proceso mayor. Por ello, el algoritmo mantiene también una segunda lista tabú que prohíbe a un pedido, durante un número determinado de movimientos de extracción *Tb_Ext*, abandonar la solución.

Criterio de Aspiración: Usamos la forma más simple de criterio de aspiración, la cual acepta un movimiento tabú si produce mejor solución que la mejor solución encontrada hasta ese momento.

Criterio de Parada: El algoritmo finaliza después de un número fijo de iteraciones *Num_Iter*.

3. Resultados Computacionales

La primera etapa de los experimentos computacionales es la construcción de un conjunto de problemas. Posteriormente, se comparan los resultados de la búsqueda tabú con los resultados previos obtenidos usando dos métodos heurísticos propuestos por Gabrel (1995).

3.1. Generación aleatoria de problemas

Las instancias que se crearon para probar los diferentes métodos heurísticos fueron generadas aleatoriamente a partir de un ratio de utilización a priori p diseñado en Kroon *et al* (1995) para el problema FSP.

El ratio ρ del sistema es un indicador de la carga de trabajo por unidad de capacidad disponible, y se define como sigue:

$$\rho = \frac{\text{carga total de trabajo esperada (en unidades de tiempo)}}{\text{capacidad total (en unidades de tiempo)}} = \frac{n \times \frac{1}{2} D}{T \times m}$$

donde:

- D indica la duración máxima de un trabajo
- T representa la longitud del horizonte temporal
- n denota el número de trabajos
- m es el número de máquinas

Tres valores del ratio de utilización serán utilizados:

- Ratio de utilización baja (ρ=0.8)
- Ratio de utilización media(ρ =1)
- Ratio de utilización alta (ρ=1.2)

Consideramos un horizonte temporal de 1000 unidades de tiempo e instancias con n=50, n=75 y n=100 trabajos. El número de máquinas fue de m=4, m=6 y m=8.

El parámetro D se obtuvo desde la formula a partir de cada valor especificado por el ratio de utilización. El tiempo de proceso t_i de cada trabajo i se generó aleatoriamente de la distribución uniforme (0,D), y el primer instante de comienzo a_i se generó aleatoriamente de igual forma en el intervalo (0, T-t_i). Para cada combinación (TI, ρ, m, n) se generaron 10 instancias, lo que produce un total de 540 instancias. Todos los valores que aparecen en los resultados van referidos al promedio sobre las 10 instancias generadas para cada combinación.

3.2. Resumen de Resultados

La Tabla 1 muestra los valores asignados a los parámetros de la búsqueda tabú. Dichos valores fueron determinados de un estudio previo de sensibilidad del algoritmo.

Tabla 1. Configuración de parámetros de la búsqueda tabú

parámetro	valor
$frec_{inc}$	n/4
I	0.2
D	0.6
Tb_Ext	n/2
Num_Iter	1000

Para probar el comportamiento de la búsqueda tabú, comparamos los resultados con los obtenidos de la aplicación de tres procedimientos heurísticos propuestos en Gabrel(1995) para resolver el problema VSP. En ese trabajo se modela VSP como un problema de Máximo Conjunto Independiente sobre un grafo que recoge los solapamientos entre trabajos. En las tablas se recoge siempre el mejor resultado de las tres heurísticas, denotado por *HG*.

La Tabla 2 muestra el promedio sobre el número de trabajos realizados (*Num_T*), el promedio de tiempo de CPU (*T_CPU*), dado en segundos, requerido para resolver los problemas, y el número de instancias, sobre las 10 instancias de cada tipo de problema, donde cada heurística fue la mejor (*Num_I*).

De estos resultados se puede concluir que la búsqueda tabu es mejor en la amplia mayoría de los casos que los resultados de las aproximaciones heurísticas *HG*. Sólo para algunas instancias con pocas máquinas, *HG* presenta un mejor promedio de trabajos realizados.

Respecto a los tiempos de CPU, la búsqueda tabu presenta menores tiempos para prácticamente todas las instancias.

4. Conclusiones

En este trabajo, hemos presentado un procedimiento de resolución basado en búsqueda tabu para el problema conocido como *Variable Job Scheduling Problem* (VSP). Para probar la calidad del algoritmo de búsqueda tabú, se han comparado sus resultados con los mejores

resultados alcanzados con tres procedimientos heurísticos definidos para el problema. Los resultados computacionales ponen de manifiesto que nuestro método encuentra soluciones de mejor calidad utilizando además menores recursos de tiempo.

Tabla 2. Resumen de resultados

			TI=1						*TI=2*					
			NUM_T		*T_CPU*		*NUM_I*		*NUM_T*		*T_CPU*		*NUM_I*	
ρ	*m*	*n*	*HG*	*TS*	*HG*	*TS*	*H*	*TS*	*HG*	*TS*	*HG*	*TS*	*HG*	*TS*
0.8	4	50	39.9	**40.3**	**6.9**	7.3	6	**8**	**38.9**	**38.9**	16.8	**11.8**	6	**7**
		75	**59.3**	58.9	38.8	**13.8**	**7**	3	**60.5**	60.2	63.4	**34.9**	**6**	4
		100	**80**	79.5	72.5	**29.6**	**6**	4	**82**	80.2	123.4	**47.1**	**8**	1
	6	50	41.8	**42.4**	27.2	**8.1**	1	**9**	41.5	**41.9**	38.5	**19.3**	5	**8**
		75	63	**64**	87.1	**18.9**	3	**8**	61.7	**62.2**	195.6	**32.3**	4	**9**
		100	83.6	**84.4**	157.5	**35.1**	3	**8**	**83**	82.7	345.8	**64.8**	**6**	**6**
	8	50	40.4	**41.4**	72.9	**11.7**	2	**10**	40.4	**41.4**	134.4	**18.7**	3	**10**
		75	63.2	**64.6**	157.2	**20.1**	2	**9**	63.6	**64.2**	366	**37.9**	5	**9**
		100	86.2	**86.9**	283.2	**33.3**	3	**6**	85.3	**85.6**	378.4	**57.9**	**6**	5
1	4	50	35.4	**36.1**	19.3	**11**	4	**10**	**36.5**	36.1	**18.3**	19.3	**9**	6
		75	54.2	**54.5**	44.7	**28.5**	7	**9**	54.7	**55.1**	93.7	**38.2**	7	**8**
		100	**72.6**	72.4	75.9	**38.1**	**6**	4	**74.1**	72.7	172.4	**73.5**	**9**	2
	6	50	35.8	**37.1**	43.6	**13.8**	0	**9**	38.6	**39.4**	99.7	**24.8**	4	**10**
		75	55.6	**57.3**	96.2	**30.7**	2	**10**	56.2	**56.6**	201	**45.9**	6	**9**
		100	**75.7**	75.6	168.2	**48.3**	**6**	**6**	**77.5**	77.4	362.4	**89.9**	**7**	5
	8	50	37.2	**37.6**	75.5	**16.5**	4	**8**	37.6	**38.3**	86.9	**24.8**	3	**9**
		75	57.5	**58.7**	162.8	**32.6**	3	**9**	58.2	**59.2**	306.9	**62.6**	2	**10**
		100	78.1	**79.2**	316.3	**69.7**	2	**8**	78.4	**78.8**	632.8	**118.4**	5	**7**
1.2	4	50	**34.3**	**34.3**	21.1	**13.1**	**8**	**8**	**33.3**	33.2	18.8	23.7	**8**	7
		75	50.5	**51.1**	44.3	**26.2**	6	**9**	49.9	**50**	75.9	**62**	5	**6**
		100	66.8	**67.3**	81.8	**56**	6	**10**	67.7	**67.8**	163.9	**100.9**	6	**8**
	6	50	34	**34.7**	48.1	**16.9**	4	**8**	34.4	**34.9**	47.4	**29.8**	5	**9**
		75	52.8	**53.4**	103.9	**40.2**	4	**8**	52	**52.7**	213.6	**73.4**	3	**10**
		100	70.2	**71.1**	165.9	**73.7**	2	**9**	**69.1**	68.6	482.9	**139.6**	**8**	6
	8	50	34.3	**35.2**	81.6	**20.6**	2	**10**	34.1	**34.9**	74.7	**37.2**	4	**9**
		75	50.8	**51.6**	135.9	**52.5**	2	**9**	53.2	**54**	384.6	**91.9**	4	**9**
		100	71.2	**72.8**	185	**90.5**	2	**10**	68.7	**68.8**	675.3	**169.9**	**7**	5

Referencias

Gabrel V. (1995). Scheduling jobs within time windows on identical parallel machines: New model and algorithms. *European Journal of Operational Research*, 83, pp. 320-329.

Gertsbakh I. and Stern H. (1978). Minimal Resources for Fixed and Variable Job Schedules. *Operations Research*, 26 (1), pp. 68-85.

Glover F. and Laguna M (1997). Tabu Search, Kluwer Academic Publishers, Dordrecht.

Glover F., Taillard E. and Werra D.(1993). A user's guide to tabu search. *Annals of Operations Research*, 41, pp. 3-28.

Kroon L.G., Salomon M. and Van Wassenhove L. N. (1995). Exact and approximation algorithms for the operational fixed interval scheduling problem. *European Journal of Operational Research*, 82, pp. 190-205.