HEURÍSTICOS APLICADOS AL PROBLEMA JOB SHOP

Marcela Gutiérrez Mejía

Universidad EAFIT, mgutie12@eafit.edu.co, Medellín.

Abstract: Se presentan los aspectos teóricos más importantes del problema Job Shop y de algunos métodos como algoritmos constructivos, búsqueda local y algoritmos genéticos utilizados para encontrar buenas soluciones. Se observó que en los problemas pequeños se pueden obtener buenas soluciones fácilmente, pero en los problemas grandes se debe elegir el algoritmo y los parámetros con más cuidado ya que éstos tienen gran importancia en la solución que se obtiene. Finalmente se concluye que en este caso el método con el cual se obtienen mejores resultados en el algoritmo genético.

Keywords: Job Shop, Algoritmo constructivo, Búsqueda local, Recocido simulado, Algorimto genético.

1. INTRODUCCIÓN

Actualmente se busca encontrar formas de resolver ciertos problemas que pueden ser generalizados en la industria, en este caso se trata el problema de Job Shop que podría ser aplicado en casos particulares para programar el orden de las actividades que se realizan intentando minimizar el tiempo utilizado teniendo en cuenta ciertas restricciones presentes. El problema del job shop se trata teniendo en cuenta las características principales que son: un conjunto de *n* trabajos, un conjunto de *m* máquinas o recursos y un conjunto de n_j operaciones en cada trabajo donde n_j es el número de operaciones en el trabajo j.

En este problema, cada una de las operaciones pertenece a un sólo trabajo, se realiza en una única máquina y tiene un tiempo de procesamiento determinado. También se debe considerar que cada trabajo consiste en una secuencia predeterminada de operaciones que deben ser realizadas en un orden establecido (J.Blazewicz).Cada una de estas operaciones debe realizarse en una máquina durante un período de tiempo no interrumpido. Las máquinas tienen una restricción de capacidad, cada una puede hacer sólo un trabajo a la vez y además se supone que no se procesan dos operaciones seguidas de un mismo trabajo en la misma máquina como se muestra en (1), pues si esto sucede, se consideran éstas como una sola operación.

$$\mu_{ij} \neq \mu_{i+1j} \quad j = 1,2,..n \quad i = 1,2,...,n_j - 1 \tag{1}$$

El objetivo al resolver este problema es encontrar una programación realizable que tome el menor tiempo posible. Para lograr esto se deben tener en cuenta las restricciones presentadas en (2) donde S_{ij} es el tiempo de inicio de la operación i del trabajo j y p_{ij} es el tiempo que se demora la operación i del trabajo j en ser realizada (Brucker P, Jurisch B)

$$S_{ij} + p_{ij} \leq S_{i+1,j}$$
$$S_{ij} + p_{ij} \leq S_{uv} \xrightarrow{o} S_{uv} + p_{uv} \leq S_{ij} \quad (2)$$

Para resolver este problema se utilizaron diferentes métodos heurísticos, a continuación se presenta una descripción general de éstos y en el próximo capítulo se muestran los resultados y las comparaciones entre éstos.

1.1. Método constructivo

La idea del método constructivo es encontrar una solución factible de problema. Se implementó un método constructivo sin factor aleatorio y otro incluyéndolo. En el primero el orden en el que se realizaban las actividades dependía del tiempo que tomaba cada una, siendo más importantes aquellas que se demoraban más. En el segundo cada actividad tiene una probabilidad de ser elegida, teniendo más probabilidad de ser elegidas aquellas que tomaban más tiempo en ser realizadas. Después de determinar el orden en el que serán realizadas las actividades, se programan las actividades y se calcula el tiempo que toma realizar todas las actividades, en éste se debe determinar el número de iteraciones que se van a realizar antes de correr el programa. En el primer caso se obtiene una solución, en el segundo se conserva la solución con menor tiempo de ejecución.

1.2. Búsqueda Local

El objetivo al realizar algoritmos de búsqueda local es no quedarse en óptimos locales. En este caso, se realizó un algoritmo búsqueda local y partiendo de éste, un recocido simulado. En el primer caso los parámetros de entrada del algoritmo son: El tamaño de vecindario y el número de vecindarios (criterio de parada). Inicialmente se encuentra una solución inicial con el algoritmo constructivo aleatorizado, se calculan los vecinos de esta solución (soluciones obtenidas realizando un cambio en la original) y si alguno de estos vecinos tiene un mejor valor de la función objetivo se toma éste como solución inicial y se buscan sus vecinos. Se realiza este procedimiento hasta que no se encuentren mejores vecinos o si se han realizado un número de iteraciones determinado. En el recocido simulado los parámetros de entrada diferentes a los de la búsqueda local son: La temperatura inicial del sistema, el número de iteraciones que se realizarán sin cambiar el valor de la temperatura inicial (Lt) y la tasa de enfriamiento de la temperatura (un número entre 0 y 1). Éste tiene la misma estructura de la búsqueda local pero cuando un vecino no tiene mejor valor de función objetivo tiene una probabilidad determinada por (3) de ser escogida como solución inicial para buscar sus vecinos.

$$P = \exp\left[\frac{Z'(s) - Z(s)}{C \times T}\right] \quad (3)$$

Donde Z'(s) es el valor de la función objetivo de la nueva solución, Z(s) es el valor de la función objetivo de la mejor solución encontrada hasta el momento, T es la temperatura actual la cual depende de los parámetros de entrada y C es una constante (1.38054×10^{-3}). Siempre se guarda la mejor solución obtenida durante el algoritmo, es decir, la que tenga menor valor de la función objetivo.

1.3. Algoritmo genético

Los métodos poblacionales buscan aplicar las leyes de la evolución biológica a problemas de optimización, es decir, las soluciones con mejor valor de función objetivo pasan a las siguientes generaciones, cada solución contiene la información para las siguientes generaciones, se realiza cruce entre dos soluciones (padres) obteniendo una nueva (hija) con características de ambos padres y puede darse mutación, es decir pequeños cambios en alguna solución obtenida. Los parámetros de este algoritmo son: El tamaño de la población inicial (N), el número de generaciones (G). Se generan N soluciones (padres) con el algoritmo constructivo aleatorizado y partiendo de éstas se generan otras N (hijas), se toman las N soluciones con mejor de la función objetivo y se repite el procedimiento el número de generaciones determinado (G). Se guarda la mejor solución obtenida durante todo el procedimiento.

2. IMPLEMENTACION DE LOS ALGORITMOS Y RESULTADOS

La matriz que se ingresa tiene la estructura presentada en la Figura 1.

Operación	1	2	…	n
Tiempo				
Trabajo				
Maquina				

Fig 1. Estructura de la matriz de los datos de entrada.

Al correr el algoritmo constructivo aleatorizado variando el número de iteraciones se obtuvieron los resultados presentados en la Tabla 1.

Tabla 1. Resultados obtenidos con el algoritmo constructivo aleatorizado

Iteraciones	Mejor Solución	Peor Solución	media	Varianza	Tiempo cómputo
50	1370	1711	1526	5196	1.53 s
100	1343	1711	1517	7030	3.04 s
250	1328	1842	1526	8205	7.58 s
500	1300	1821	1519	7872	15.06 s

Como se ve en la Tabla 1, la solución obtenida al aumentar las iteraciones es mejor, pero también aumenta la variabilidad de las soluciones y el tiempo de cómputo.

Posteriormente se implementó el algoritmo de búsqueda local y de recocido simulado. En la Tabla 2, se presentan las mejores soluciones obtenidas con el recocido simulado con diferentes valores de los parámetros de entrada. Cuando se varía un parámetro los otros dos se mantienen constantes.

De la Tabla 2. se puede concluir que no hay comportamientos claros que permitan concluir sobre la influencia de los parámetros Lt y tasa de enfriamiento en las soluciones. Pero la temperatura inicial tiene gran impacto en las soluciones obtenidas.

Por último se implementó el algoritmo genetico, los mejores resultados se obtuvieron cuando se realizó el cruce por mutación, éstos se presentan en la Tabla 3 y 4. En estas dos Tablas se realizan pruebas con los mismos parámetros pero variando la probabilidad de mutación.

Tabla 2. Resultados obtenidos con el recocido simulado

T.inicial	20	40	60	**80**	100
Mejor Solución	1380	1367	1399	1317	1355
Lt. Max	10	20	30	**40**	50
Mejor Solución	1380	1367	1429	1366	1413
Tasa de enfriamiento	0.2	0.4	0.6	0.8	**0.95**
Mejor olución	1393	1376	1440	1398	1376

Tabla 3. Resultados obtenidos con el algoritmo genético con cruce por mutación (Prob 0.05)

Parámetros	[10 5]	[20 8]	[30 5]	[40 10]
Valor función objetivo	1328	1333	1322	1307
Tiempo Computacional	1.84 s	5.57 s	5.6 s	13.7 s

Tabla 4. Resultados obtenidos con el algoritmo genético con cruce por mutación (Prob 0.1)

Parámetros	[10 5]	[20 8]	[30 5]	[40 10]
Valor función objetivo	1294	1342	1298	1294
Tiempo Computacional	1.9 s	5.57 s	5.5 s	13.82 s

De las Tabla 3 y 4 se puede ver que en este caso, al aumentar la probabilidad de mutación se obtienen mejores soluciones, es decir, su valor de dunción objetivo es menor.

Por último, en la Tabla 5. Se presentan resultados obtenidos con los diferentes métodos para poder compararlos. Los parámetros de cada método son los mencionados en el capítulo 1. En esta Tabla se puede ver claramente que para el problema en particular que se está tratando, las mejores soluciones implican mayor tiempo de cómputo. Por otro lado, las soluciones obtenidas con el algoritmo genético con cruce por mutación son menos variables que las obtenidas por los demás métodos.

Tabla 4. Resultados

Algoritmo	Mejor Solu-ción	Peor Solu-ción	Parámetros	Tiempo computacio -nal
Constructivo	1519	1519		0.03 s
Constructivo Aleatorizado	1300	1842	500	15.06 s
Búsqueda Local	1336	1617	[5 8]	0.09 s
Recocido simulado	1317	1658	[20 50 80 0.9 40]	0.003 s
Cruce por un punto	1307	1490	[20 8]	5.39 s
Cruce por dos puntos	1300	1437	[30 5]	5.54 s
Mutación	1296	1390	[10 5 0.1]	13.82 s

3. CONCLUSIONES

- Cuando el problema es grande, como en este caso, se debe probar a partir de cuantas iteraciones se puede obtener una buena solución. En este caso se debe tener en cuenta la aleatoriedad de las soluciones, pues con las mismas iteraciones pueden obtenerse soluciones, en algunos casos, muy diferentes.

- En este problema en particular, las buenas soluciones se obtuvieron con un tiempo de cómputo alto.

- En el recocido simulado es de gran importancia la temperatura inicial elegida, pues ésta tiene gran influencia en la calidad de las soluciones.

- Se observa que en este caso, al aumentar la probabilidad de mutación en el algoritmo genético se obtienen soluciones con valor de la función objetivo menores.

- Los mejores resultados se obtuvieron con el algoritmo genético con cruce por mutación. Pero se debe tener presente que debido al alto componente aleatorio no es posible decidir cual es mejor.

REFERENCIAS

Brucker P, Jurisch B. “A branch and bound algorithm for the job-shop scheduling”

Dae Sung Leea, Vassilios S. Vassiliadisb, Jong Moon Parkc “A novel thresholdaccepting meta-heuristic for the job-shop scheduling problem”.

J.Blazewicz, K.Ecker, E.Pesh. *Handbook on scheduling*.