Heurística para minimizar el *makespan* y la tardanza máxima en el problema de *scheduling Job-Shop* multi-recurso con rutas lineales

Andrés Alberto García-León[1]

Resumen

El presente artículo tiene por objetivo la formulación y validación de una heurística, basada en la búsqueda local para minimizar el *makespan* (mayor tiempo de finalización de pedidos) y la tardanza máxima (máximo retraso), para el problema de *scheduling Job-shop* multi-recurso con rutas lineales. Este problema modela situaciones de los procesos industriales en los cuales existe un conjunto de máquinas para seleccionar un subconjunto de ellas y desarrollar cada una de las operaciones de pedidos, que se organizan según una configuración de *Job-shop*. La validación se realizó considerando casos de la literatura y los resultados muestran que la heurística encuentra nuevas y mejores soluciones para varias instancias, en tiempos racionales de ejecución.

Palabras claves

Scheduling, Job-shop multi-recurso, rutas lineales, *makespan*, tardanza máxima

1. Introducción

El enfoque moderno de la administración de operaciones motiva a las empresas a obtener productos de máxima calidad, con precios cada vez más bajos. Para el logro de este objetivo, las empresas deben desarrollar estructuras organizativas capaces de maximizar simultáneamente la productividad y el servicio al cliente. Los criterios regulares, que han sido empleados para optimizar el servicio al cliente, están basados en los tiempos de finalización de los pedidos y revisten de importancia al considerar las fechas que se han establecido con los clientes para dar cumplimiento.

El criterio regular que ha sido más estudiado es el makespan "C_{max}" (Genova & Guliashki, 2015), que consiste en minimizar el tiempo máximo de finalización de todos los pedidos. Sin embargo, minimizar otros criterios que capturan factores críticos que afectan la rentabilidad y, por consiguiente, su competitividad (García-León et al., 2015). Para responder a este requisito, minimizar tardanza máxima "T_{max}" conduce al mejoramiento del servicio al cliente.

El objetivo de la presente publicación consiste en desarrollar una heurística capaz de solucionar cada criterio en problemas de *scheduling*. Para el caso, se analiza el problema de *Job-shop* multi-recurso con rutas lineales; un problema que reviste de complejidad computacional y puede modelar diversos escenarios industriales.

El documento es organizado como sigue: en la sección dos se definen los aspectos relevantes al problema como lo son su definición, la complejidad y los criterios regulares. En la sección tres se ilustran los pasos para la formulación de la heurística. En la sección cuatro se ilustran los resultados y en la sección cinco se describen los usos potenciales.

2. Formulación del problema

Para comprender el problema se requiere, inicialmente, definir el enfoque clásico de *scheduling Job-shop, Job-shop Scheduling Problem, JSP. JSP* es uno de los desafíos de optimización combinatoria más complejos y en este se debe procesar un conjunto J de n pedidos $J = \{J_1, J_2, \ldots J_n\}$ en un conjunto M de m máquinas, que están siempre disponibles para procesar los pedidos.

[1] Programa de Ingeniería Industrial, Facultad de Ingeniería, Universidad de Ibagué, Ibagué, Colombia. Grupo de investigación GINNOVA. Correo electrónico andres.garcia@unibague.edu.co

Cada máquina solo puede realizar una operación a la vez. Cada pedido consiste en una secuencia de operaciones fija denominada ruta. No se permite la prioridad de operaciones, lo que significa que una operación no puede ser interrumpida una vez inicie. Los tiempos de procesamiento de las operaciones son enteros, conocidos e incluyen los tiempos de alistamiento. C_i representa el tiempo de finalización de ***Ji***, mientras que d_i representa su fecha de entrega. El objetivo del problema es determinar un programa, es decir, la secuencia (orden) de las operaciones en las máquinas y los tiempos de finalización para optimizar un objetivo.

Aunque JSP permite representar configuraciones del escenario real, la tendencia moderna en la producción conlleva a la flexibilidad; por ejemplo, en las operaciones que son realizadas por varias máquinas de forma simultánea de un grupo de posibilidades. Este supuesto no puede ser verificado por el JSP. Por lo tanto, la integración de esta flexibilidad en el JSP conduce a un modelo más general, que es considerado por el *Job-shop* multi-recurso con rutas lineales, JSMRL.

JSMRL es una extensión del JSP en la que cada operación requiere, por lo menos, una máquina para ser procesada. La máquina o las máquinas, dependiendo de las características de la operación, son seleccionadas de un subconjunto de M. Para obtener una solución factible del JSMRL, las operaciones deben ser asignadas y secuenciadas en máquinas. La asignación consiste en seleccionar para cada operación, la(s) máquina(s) que realizará(n) la operación procesamiento. Mientras que la secuenciación trata de obtener el orden de las operaciones en cada una de las máquinas seleccionadas.

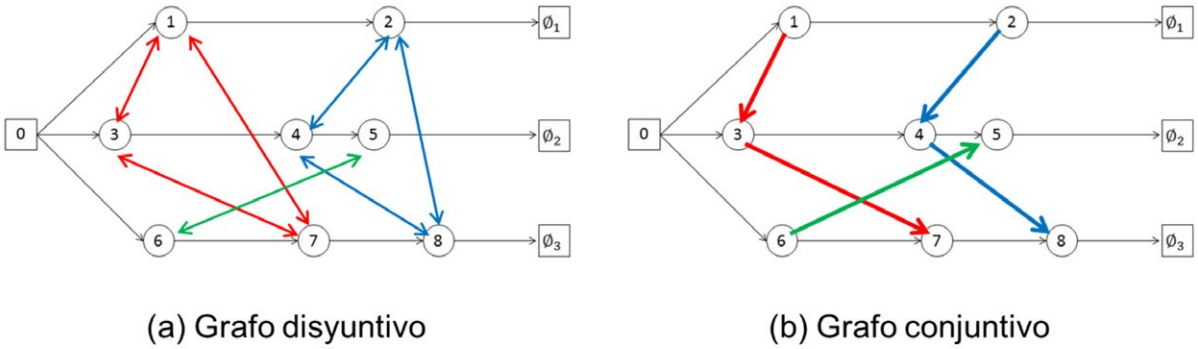
Los grafos disyuntivo y conjuntivo, formulados por (Roy & Sussmann, 1964) son los más ampliamente utilizados para resolver los problemas de programación *Job-shop*. Una ventaja importante de este grafo es su capacidad para modelar, de manera eficiente, diferentes restricciones y características de los problemas de programación de talleres.

El gráfico se representa como **G = (V, A, E)**, donde V es el conjunto de nodos asociados a las operaciones de pedidos; un nodo ficticio 0 que representa el inicio de cada trabajo; y n nodos φi, cada uno de los

cuales representa el tiempo de finalización de Ji. El conjunto de arcos conjuntivos A contiene arcos que conectan dos operaciones consecutivas de cada pedido, el nodo 0 y cada primera operación de cada pedido, y la última operación de Ji con el nodo φi.

Finalmente, E es el conjunto de arcos disyuntivos entre pares de operaciones asignados a cualquier máquina. La figura 1 ilustra los dos grafos con que se representa el problema de JSP y, en este caso, un problema con tres pedidos (arcos horizontales) con nodos terminales φi. Para el caso del pedido 1, este tiene dos operaciones, las cuales son representadas por los nodos 1 y 2. Los nodos 3, 4 y 5 hacen parte del pedido 2 y los otros del pedido 3. En el grafo disyuntivo (a) se observa conflicto en la secuencia de las operaciones; por ejemplo, los arcos con doble flecha, los de color rojo, en un algoritmo se debe determinar la dirección o la secuencia de operaciones para optimizar algún criterio. Al solucionar el conflicto se obtiene un grafo conjuntivo (b), el cual debe asegurar la no existencia de ciclo en la solución.

Figura 1. Ejemplo ilustrativo de grafos para el JSP



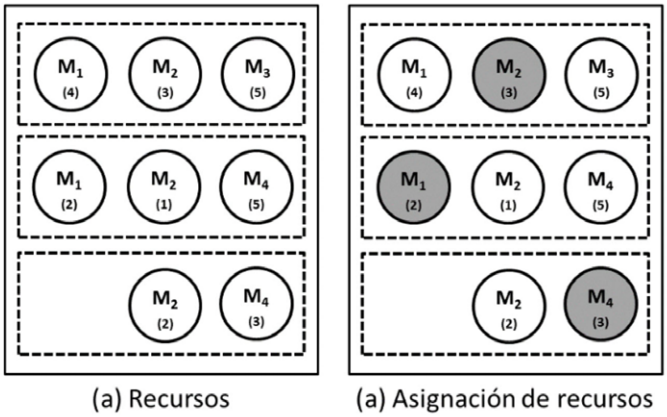
Fuente: este estudio

Una selección Sk, que corresponde a un programa del JSP para una tarea determinada, se obtiene fijando una dirección a cada arco disyuntivo. La selección es factible si el grafo inducido no conduce a un ciclo. Vale la pena mencionar que el grafo contiene muchos arcos redundantes que deben eliminarse para garantizar que cada nodo tenga como máximo un predecesor y un sucesor en la máquina que se realice. Por lo tanto, todos los nodos correspondientes a las operaciones del JSP tendrán dos arcos de entrada y dos arcos de salida, excepto la primera (o la última), operación en la secuencia de máquinas que solo tendrá un arco de entrada (o de salida).

JSMRL adiciona dos tipos de flexibilidad a JSP: multi-recurso y recurso flexible. Multi-recurso considera que para realizar una operación se requiere de una o varias máquinas, mientras que recurso flexible implica la selección de estas de un conjunto dado. Así mismo, el tiempo de procesamiento de la operación estará determinado por el mayor tiempo de operación de la máquina seleccionada.

Para clarificar, la Figura 2 ilustra con un ejemplo de una operación que requiere tres recursos o máquinas. En este caso, la primera máquina debe ser seleccionada entre $\boldsymbol{M_1}$, $\boldsymbol{M_2}$ o $\boldsymbol{M_3}$ (rectángulo superior en a) y el tiempo de procesamiento está determinado por el número inferior entre paréntesis. Para la segunda máquina la selección se realizará entre M_1, M_2 o M_4 y para la tercera entre M_2 o M_4. Vale la pena mencionar que solo se presenta una máquina de cada tipo. El rectángulo de la derecha (b) ilustra la asignación que se ha realizado, en el cual se seleccionaron las máquinas M_2, M_1 y M_4, respectivamente, y el tiempo de procesamiento de la operación es de tres unidades.

Figura 2. Asignación de máquinas en una operación



Fuente: este estudio

Obtener una solución factible para el JSMRL implica solucionar el problema de asignación en las operaciones y la secuencia en las máquinas, sin generar ciclos. Respecto al grafo, de cada operación deben ingresar y salir de una operación cuántos recursos estén asignados, el tiempo de inicio de la operación será el de mayor tiempo de finalización de todos sus predecesores inmediatos. Una vez se cuente con la solución de un grafo, se puede extraer información que contribuirá a la construcción de heurísticas.

3. Construcción de la heurística

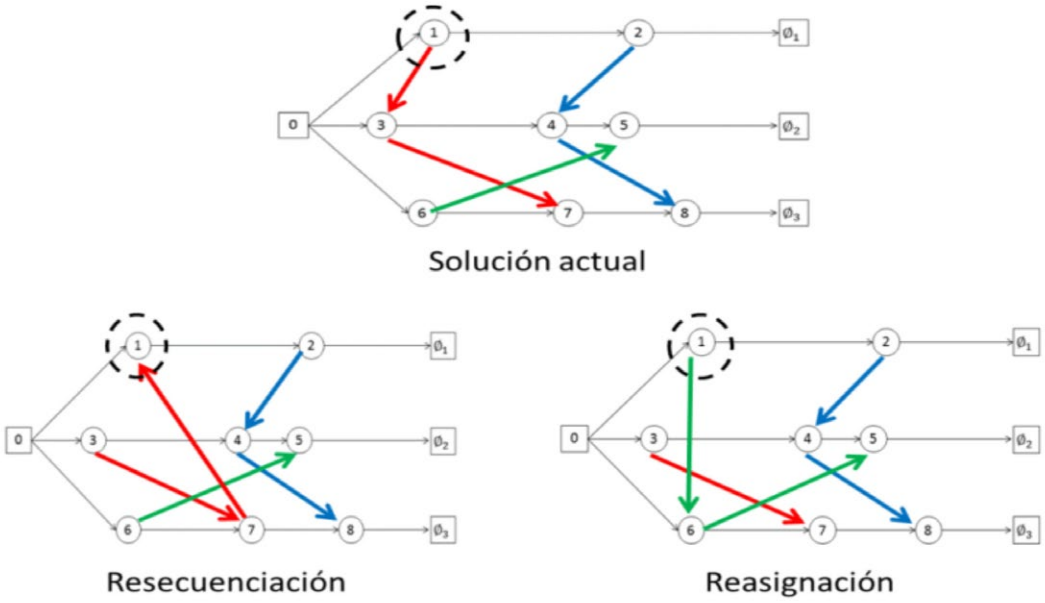
Para el caso del problema, el objeto del proceso de búsqueda local consiste en determinar, a partir de una solución representada en el grafo, la factibilidad del movimiento de operaciones y los tiempos de finalización de los pedidos, sin hacer transformaciones. De hacerse alguna transformación no tendría sentido la formulación de este tipo de enfoques.

En el JSMRL, dos tipos de movimientos pueden realizarse considerando operaciones críticas (resecuenciación y reasignación). La resecuenciación consiste en cambiar la posición de una operación en la secuencia de la máquina en que fue asignada, y la reasignación en la sustitución de un recurso por otro, siempre y cuando sea factible.

Para explicar la resecuenciación y la reasignación en un JSP flexible en la asignación de máquinas, la Figura 3 ilustra un grafo conjuntivo, en el cual se considera que la operación 1 (encerrada en un círculo discontinuo) es crítica y está secuenciada en la máquina representada por los arcos rojos. La operación 1 es la primera operación en la máquina. Si se considera la resecuenciación, en el grafo inferior izquierdo, pasa a ubicar en la tercera posición, como se ilustra en la Figura. El grafo inferior derecho ilustra la reasignación, en la cual, se asume que esta operación puede realizarse en la máquina que representa el color verde y, en este caso, pasaría a ser la primera operación. Es importante mencionar que este movimiento no debe generar ciclo.

La heurística tiene por objetivo llegar a una solución de calidad (cercana a la óptima), en tiempos relativamente cortos de computación. Esta solución la alcanza a considerar una solución inicial y, a partir de esta identifica, la(s) ruta(s) crítica(s) de los pedidos que afectan un criterio; por ejemplo, en el *makespan* la ruta que conlleve al pedido de mayor duración. En el caso de la tardanza máxima, aquel pedido que tenga mayor retraso. En las rutas críticas que se hallen se determinan los arcos críticos, los cuales son los insumos para aplicar ecuaciones al determinar su factibilidad. El resultado de este análisis permite determinar las inserciones factibles con sus valores estimados. Luego, la heurística transformará el grafo a aquel que contenga el menor valor criterio que el actual.

Figura 3. Resecuenciación y reasignación de una operación

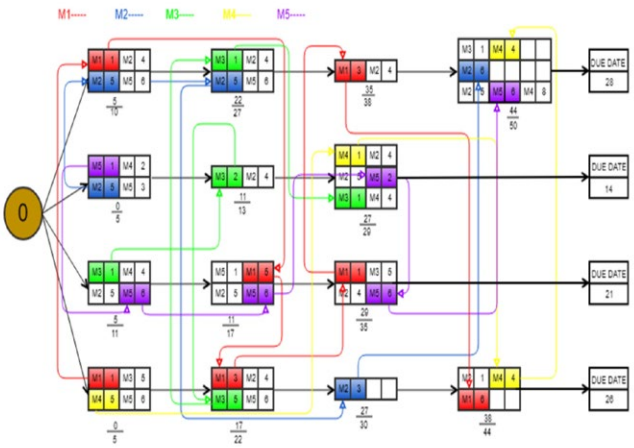


Fuente: este estudio

La heurística se desarrolla en dos etapas iterativas denominadas mejoramiento y diversificación. En la fase de perfección se busca la estimación que optimice el criterio para ir transformando el grafo. Cuando no es posible disminuir el criterio, empieza la diversificación en la cual se realizan movimientos aleatorios en un intervalo [a,b], para continuar con una nuevo ciclo de mejoramiento y, en caso de que se reduzca el criterio, se retorna a la etapa inicial. El proceso de ejecución está dado en tiempo y, en caso de que se alcance un valor de criterio de cero, la heurística finaliza el proceso. Este se da en el caso de que la tardanza máxima sea cero.

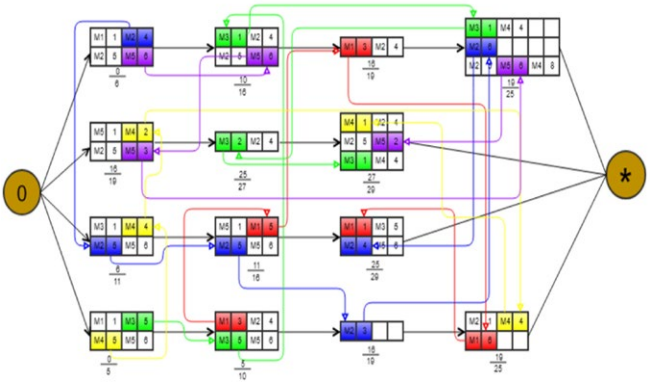
Cuando la heurística finaliza, el resultado computacional son dos archivos de texto, los cuales corresponden a la solución inicial y a la mejor solución alcanzada. En las figuras 4 y 5 se ilustra la asignación de máquinas a las operaciones con un *makespan* de 50 y luego se optimiza en la figura 5 con un *makespan* de 29.

Figura 4. Ejemplo de solución inicial (Makespan=50)



Fuente: este estudio

Figura 5. Ejemplo de solución inicial (Makespan=29)



Fuente: este estudio

4. Resultados

Para evaluar la eficiencia de la heurística se han considerado las 68 instancias (mjs01 a mjs68, ver tabla 1), para el problema multi-recurso de Dauzère-Pérès (1998). La heurística fue diseñada en lenguaje Java y el tiempo de ejecución fue de 20 minutos.

Los experimentos computacionales, respecto al *makespan*, se ilustran en la tabla 1. En la columna H, un asterisco (*) representa que la solución de referencia (columna R) ha sido alcanzada y la respuesta en negrilla indica que se ha mejorado la solución de referencia. A pesar de que la heurística no es formulada para la minimización del *makespan*, se ha alcanzado el valor mejor conocido en 6 instancias y en 24 el valor ha mejorado. Esto significa que en 30 instancias (44.12%) el enfoque que ha sido dado para la tardanza máxima es validable en la minimización del *makespan*.

Para el cálculo de la tardanza máxima, se hace indispensable el conocimiento de las fechas de entrega de los pedidos. El procedimiento para calcular las fechas de entrega son propuestas basadas en dos enfoques: E_1 y E_2 (Singer, 1998). El procedimiento consiste en hallar los tiempos promedios de operación por pedido y multiplicarlos por un factor f. En este caso, se ha considerado un factor f de 1.3; es decir, se estimará que la fecha de entrega de cada pedido es incrementada en un 30 % por encima del tiempo promedio del pedido. Los resultados se ilustran en la tabla 2.

A partir de los resultados alcanzados, es evidente que la heurística alcanza una solución óptima en la mayoría de las instancias y que, en algunas de ellas, el algoritmo de solución inicial logra optimizar la solución. Si bien en la literatura considerar un factor (f) de 1.3 para optimizar el servicio al cliente en el problema de JSP resulta riesgoso, en el problema no lo es. Sin embargo, se considera que la evaluación de factores por debajo de 1.3 podría impactar en las soluciones que alcanzan solución óptima. No obstante, en las instancias desde mjs10 a mjs29, en donde no se alcanza un óptimo (al validar el hecho de que puede dar cero), puede resultar interesante la realización de estrategias de diversificación como la optimización de la tardanza total o la minimización de los tiempos de finalización de los pedidos.

Tabla 1. Comparación de mejores soluciones en C_{max}

Inst	*R*	*H*	*Inst*	*R*	*H*	*Inst*	*R*	*H*	*Inst*	*R*	*H*
mjs01	361	*	mjs18	1544	**1532**	mjs35	265	*	mjs52	317	364
mjs02	384	**382**	mjs19	1572	**1503**	mjs36	225	228	mjs53	353	378
mjs03	378	380	mjs20	1033	**1003**	mjs37	207	214	mjs54	311	337
mjs04	394	**391**	mjs21	916	**870**	mjs38	241	*	mjs55	493	590
mjs05	643	**637**	mjs22	924	943	mjs39	210	216	mjs56	508	572
mjs06	585	**561**	mjs23	957	**948**	mjs40	241	*	mjs57	500	567
mjs07	644	**627**	mjs24	918	**848**	mjs41	218	223	mjs58	530	587
mjs08	575	**568**	mjs25	1513	**1505**	mjs42	250	*	mjs59	490	547
mjs09	568	**564**	mjs26	1481	**1322**	mjs43	219	223	mjs60	268	**262**
mjs10	928	**925**	mjs27	1566	**1513**	mjs44	258	**256**	mjs61	303	306
mjs11	1057	**1004**	mjs28	1395	**1424**	mjs45	296	301	mjs62	284	285
mjs12	859	**806**	mjs29	1336	**1329**	mjs46	300	320	mjs63	289	294
mjs13	827	875	mjs30	218	221	mjs47	333	348	mjs64	240	245
mjs14	946	966	mjs31	218	224	mjs48	327	335	mjs65	381	406
mjs15	1469	**1418**	mjs32	219	227	mjs49	356	*	mjs66	423	444
mjs16	1312	1314	mjs33	224	229	mjs50	327	362	mjs67	408	448
mjs17	1572	**1562**	mjs34	213	217	mjs51	373	407	mjs68	400	429

Fuente: este estudio

Tabla 2. Resultados para la tardanza máxima

Inst	E1		E2		Inst	E1		E2	
	Sol i	M	Sol i	M		Sol i	M	Sol i	M
mjs01	189	0	194	0	***mjs35***	3	0	196	0
mjs02	262	0	328	0	***mjs36***	42	0	203	0
mjs03	203	0	199	0	***mjs37***	44	0	280	0
mjs04	276	0	318	12	***mjs38***	0	0	182	0
mjs05	386	5	381	23	***mjs39***	30	0	167	0
mjs06	261	0	323	0	***mjs40***	0	0	170	0
mjs07	603	34	499	7	***mjs41***	78	0	232	0
mjs08	476	1	456	0	***mjs42***	0	0	130	0
mjs09	344	0	279	0	***mjs43***	0	0	221	0
mjs10	1475	543	1387	493	***mjs44***	107	0	144	0
mjs11	1557	662	1181	550	***mjs45***	0	0	259	0
mjs12	1282	420	1306	387	***mjs46***	0	0	297	0
mjs13	1118	447	1212	424	***mjs47***	182	0	329	0
mjs14	1304	569	1502	524	***mjs48***	0	0	137	0
mjs15	1982	788	1718	726	***mjs49***	0	0	255	0
mjs16	1916	767	1883	771	***mjs50***	572	0	702	0
mjs17	2208	1003	2507	972	***mjs51***	732	0	927	10
mjs18	2240	1000	2206	840	***mjs52***	546	0	527	0
mjs19	2028	968	1988	807	***mjs53***	616	0	664	1
mjs20	1566	632	1506	499	***mjs54***	551	0	766	0
mjs21	1408	527	1243	425	***mjs55***	828	0	882	124
mjs22	1677	531	1435	502	***mjs56***	1076	0	1027	179
mjs23	1470	573	1604	504	***mjs57***	747	0	936	84
mjs24	1218	451	1504	432	***mjs58***	840	0	995	188
mjs25	2251	910	2116	867	***mjs59***	945	0	1041	155
mjs26	2061	873	2139	787	***mjs60***	497	0	446	0
mjs27	2237	947	2085	913	***mjs61***	467	0	498	0
mjs28	2307	913	1830	799	***mjs62***	371	0	343	0
mjs29	2227	691	1977	739	***mjs63***	288	0	476	0
mjs30	0	0	142	0	***mjs64***	263	0	272	0
mjs31	66	0	305	0	***mjs65***	359	0	542	0
mjs32	154	0	198	0	***mjs66***	594	0	568	0
mjs33	75	0	224	0	***mjs67***	416	0	645	0
mjs34	0	0	104	0	***mjs68***	454	0	624	0

Fuente: este estudio

5. Potencial uso

Esta publicación hace parte de uno de los resultados de la investigación intitulada *Formulación y validación de heurísticas para optimizar el servicio al cliente en configuraciones flexibles en el departamento del Tolima*. Adicionalmente, se ha aplicado por la licencia de un software para solucionar problemas que se adapten a este tipo de configuraciones, pertenecientes al sector industrial. También se puede utilizar en el desarrollo de actividades pedagógicas para otros criterios que no han sido descritos.

6. Referencias

Dauzère-Pérès, S. (1998). Multi-resource shop scheduling with resource Flexibility. *European Journal of Operational Research* (107), 289-305.

García-León, A., Dauzère-Pérès, S., & Mati, Y. (2015). Minimizing regular criteria in the Flexible Job-shop scheduling problem. *Multidisciplinary International scheduling Conference: Theory & Applications*, 443-456.

Genova, K., & Guliashki, V. (2015). A survey of solving approaches for multiple objective flexible job shop scheduling problems. *Cybernetics and Information Technologies*, 15(2), 3-22.

Roy, B. Sussmann, B. (1964). Les problemes d'ordonnancement avec contraintes disjonctives. *Revue Française d'informatique et the recherche opérationnelle. Série verte*, 3, 51-65.

Singer, M. P. (1998). A computational study of branch and bound techniques for minimizing the total weighted tardiness in job-shops. *IIE Transactions*, 30(2), 109-118.