3rd International Conference on Industrial Engineering and Industrial Management
XIII Congreso de Ingeniería de Organización
Barcelona-Terrassa, September 2nd-4th 2009

Resecuenciación en talleres de flujo considerando múltiples eventos*

Ketrina Katragjini, Eva Vallada, Rubén Ruiz

Grupo de Sistemas de Optimización Aplicada, Instituto Tecnológico de Informática, Universidad Politécnica de Valencia, Edificio 7A, Camino de Vera S/N, 46021 Valencia, España.{ketrina, evallada , rruiz}@iti.es

Palabras clave: taller de flujo, resecuenciación, estabilidad

1. Introducción

En un taller de flujo de permutación tenemos un conjunto N de trabajos, $N = \{1,\ldots, n\}$, a procesar en un conjunto M de máquinas, $M = \{1,\ldots, m\}$. Cada trabajo tiene que visitar todas las máquinas, siendo la secuencia de proceso la misma para todos los trabajos. La secuencia de los trabajos en las máquinas no cambia, es decir, si el trabajo 3 sigue al trabajo 5 en la primera máquina, también lo hará en el resto de máquinas.

En un problema de este tipo tenemos $n \cdot m$ operaciones, una por cada máquina y trabajo y un número de secuencias posibles igual a $n!$. Las secuencias de fabricación indican el orden y los tiempos de inicio y fin de todos los trabajos en todas las máquinas. La secuenciación o "*scheduling"* es el proceso de creación de la secuencia de fabricación para un conjunto dado de trabajos. La resecuenciación o "*rescheduling*" es el proceso de actualización continua de la secuencia, adaptándola a las circunstancias dinámicas del taller.

Las operaciones reales de fabricación se caracterizan por tener un gran número de eventos no previstos. Eventos que pueden llevar a la desviación de la secuencia real con respecto a la programada son: llegadas de nuevos trabajos urgentes, roturas inesperadas de máquinas, retrasos en la llegada de la materia prima, variaciones en los tiempos de proceso de los trabajos, cambios en las fechas de entrega etc. (Subramaniam, Raheja y Reddy, 2005). Dado que estos eventos inesperados pueden deteriorar las prestaciones del sistema, muchas veces es importante resecuenciar para reducir su impacto. Como consecuencia de todo ello, los imprevistos se definen frecuentemente como *factores de resecuenciación* (Dutta, 1990; Dhingra, Musser y Blankership, 1992).

1.1. Estrategias para resecuenciar

En entornos dinámicos, donde las condiciones del taller o del problema varían constantemente, se conocen dos estrategias de resecuenciación: secuenciación online y técnicas de tipo predictivo-reactivo (Vieira, Herrman y Lin, 2003).

En la primera, no se construyen secuencias propiamente dichas. En cada momento se usan reglas de asignación de trabajos a máquinas utilizando la información actual (reglas de despacho dinámicas). En la segunda estrategia, en una primera fase se crea una secuencia

* Este trabajo está parcialmente subvencionado por el Ministerio de Ciencia e Innovación, bajo los proyectos "OACS - Optimización Avanzada de la Cadena de Suministro" y "SMPA - Secuenciación Multiobjetivo Paralela Avanzada: Avances Teóricos y Prácticos" con referencias IAP-020100-2008-11 y DPI2008-03511/DPI, respectivamente

predictiva off-line tratando el problema estático o inicial, y luego se actualiza continuamente la secuencia adaptándola a las circunstancias dinámicas de la planta.

1.2. Políticas de resecuenciación

En la literatura se citan principalmente tres políticas de resecuenciación: periódica, por eventos e híbrida (Vieira, Herrman y Lin, 2003). En la primera, la resecuenciación se realiza periódicamente, no procesando de manera inmediata los eventos que ocurren entre dos puntos, en cambio, en la segunda, se resecuencia como consecuencia de eventos que afectan al sistema al tiempo que éstos ocurren. Combinando los dos métodos básicos se obtiene la forma híbrida donde se resecuencia periódicamente y también cuando ocurren eventos denominados como críticos, es decir, eventos que no pueden esperar o que requieren atención inmediata debido a su importancia.

1.3. Medidas de eficacia

Las medidas de eficacia para problemas de resecuenciación de máquinas se pueden dividir en dos grupos: medidas de eficacia de la secuencia y medidas de estabilidad. Las medidas que se basan en el tiempo son las clásicas del problema estático de scheduling inicial, como por ejemplo el Makespan o $C_{\text{máx}}$, objetivo que pretende minimizar el tiempo máximo de finalización, o lo que es lo mismo, el tiempo de finalización del último trabajo de la secuencia en la última máquina. Estas medidas se conocen muy bien y se han estudiado en la literatura con profusión. Comparativamente, la literatura científica es mucho más escasa en lo que se refiere al estudio del impacto que tienen los eventos o cambios en el taller sobre la secuencia original o sobre el valor del objetivo de optimización original.

Este impacto se conoce como "estabilidad" pero no existe una definición estándar de la medida (Rangsaritratsamee et al. 2004). Church y Uzsoy (1992) usaron como indicador de estabilidad el número de veces que se resecuencia, Wu, Storer y Chang (1993) definen la estabilidad en términos de desviación de los tiempos de inicio de los trabajos entre la secuencia original y la resecuenciada. Otra medida relacionada con la estabilidad es el "nerviosismo" de las secuencias. El termino "nerviosismo" se empezó a utilizar originalmente en el contexto de los sistemas de planificación de requerimientos de materiales (MRP) donde se definía como "Cambios continuos en la planificación" o "Inestabilidad" (Vollmann, William y Whybark, 1997). En la literatura sobre la resecuenciación de máquinas, el "nerviosismo" se usa para indicar cambios continuos en la secuencia debidos a acciones frecuentes de reprogramación y, en muchas ocasiones, se trata como opuesto a la estabilidad.

En entornos estáticos la estabilidad no es un problema ya que no hace falta actualizar las secuencias, en cambio, en entornos dinámicos la estabilidad es importante ya que continuos cambios pueden dar origen a costes adicionales de preparación de máquinas, reconfiguración de recursos secundarios, reprogramación de los turnos de trabajo etc. (Pfeiffer, Kadar y Monstori, 2007).

1.4. Motivaciones y objetivos de este trabajo

La mayoría de la literatura sobre la secuenciación de trabajos no considera las características de los entornos reales de fabricación. Los artículos sobre resecuenciación tampoco consideran de manera simultánea un número representativo de eventos diferentes que se dan en los sistemas de producción. La mayoría de los trabajos se basan en la simulación, como consecuencia, los resultados son interpretables solo en el contexto del sistema simulado. Adicionalmente, existe una laguna importante entre la teoría y la práctica en problemas de secuenciación (Aytug et al. 2005).

En este trabajo se presenta un banco de pruebas y una metodología estándar para evaluar distintos algoritmos de resecuenciación, se estudia el caso más realista de perturbación de la secuencia original mediante varios tipos de eventos que ocurren de manera simultánea y se proponen algoritmos que permiten obtener un buen compromiso entre calidad y estabilidad de la solución.

El trabajo está organizado de la siguiente manera: en la Sección 2 se detallarán los eventos considerados, así como la función objetivo del problema y los métodos de resecuenciación implementados. La Sección 3 muestra los resultados obtenidos tras los experimentos computacionales. Por último, en la Sección 4 encontramos un resumen y conclusiones del trabajo.

2. Metodología

La política de resecuenciación utilizada en este trabajo es de tipo reactivo. Para proporcionar la solución inicial en la fase predictiva se ha utilizado el algoritmo IG voraz iterativo descrito en Ruiz y Stützle (2007). Mediante simulación se han pregenerado tres tipos de eventos para un subconjunto de las conocidas instancias de Taillard (1993). Los eventos que se consideran simultáneamente son: roturas de máquinas, llegadas de nuevos trabajos y retrasos en los inicios de los trabajos. El número de eventos generados para cada instancia es proporcional al *Makespan* de la secuencia original, que al fin y al cabo es la duración optimizada del problema inicial antes de que empiecen a producirse eventos.

La mayoría de la literatura en resecuenciación se centra en eventos de llegadas de nuevos trabajos y fallos de máquinas, pero las plantas reales se enfrentan diariamente con otro tipo de eventos inesperados como son los retrasos en la entrega de la materia prima. Estos eventos provocan retrasos en los inicios de los trabajos y en consecuencia, afectan negativamente al taller. A continuación se explica brevemente la manera en la que se han generado cada uno de los tres tipos de eventos considerados simultáneamente en este trabajo:

- Rotura de máquina: son fallos temporales, es decir, indican indisponibilidad de la máquina durante un periodo de duración aleatoria. Cada evento de este tipo se caracteriza por el valor del tiempo t en el que ocurre, la maquina m en la que se produce la rotura y la duración de la indisponibilidad.
- Llegada de nuevo trabajo: se caracteriza por el instante t de llegada y los tiempos de proceso del nuevo trabajo en cada máquina. Estos tiempos se han generado aleatoriamente a partir de una distribución uniforme entre 1 y 99.
- Retraso en tiempo de inicio de trabajo: se caracteriza por la duración del retraso que también se ha generado de forma aleatoria con distribución uniforme entre 1 y 99.

En la Tabla 1 se presenta el número medio de eventos generados de cada tipo por tamaño de instancia. Por ejemplo, para una instancia de 20 trabajos y 5 máquinas se van a generar en promedio, dos eventos de rotura de máquina. Sin embargo, si la instancia es de 100 trabajos y 20 máquinas, se generarán en promedio siete eventos de rotura de máquina.

	20×5	20×10	20×20	50×5	50×10	50×20	100×5	100×10	100×20
Fallos de máquinas	2	2	3	3,3	3,6	4	6	6	7
Llegadas de nuevos trabajos	0,7	0,7	1,4	1,3	1,4	1,8	2,5	2,2	3,2
Retrasos de trabajos	0,5	1,3	3,3	3	4,6	10,1	4	10,1	17,8

Tabla1: Numero medio de eventos por tamaño de instancia

Como se ha comentado anteriormente, no conocemos otro trabajo donde se estudien de manera simultánea estos tres eventos tan importantes. En Yamamoto et al. (1985), Abumaizar y Svetska (1997), Jain y Elmaraghy (1997), Liao y Chen (2004), Arnaut y Rabadi (2008), se consideran solo eventos de roturas de maquinas, Kim y Kim (1994), Hall y Potts (2004), Moratori, Petrovic y Vazquez (2008), consideran solo llegadas de nuevos trabajos urgentes, Adhitya, Srinvasan y Karimia (2007) consideran retrasos en los inicios de los trabajos debidos a retrasos en la disponibilidad de la materia prima.

Se define como punto de resecuenciación o reprogramación el instante de tiempo en el cual se produce un evento inesperado que afecta al rendimiento del sistema.

2.1. Función objetivo

En la subsección 1.3. se han comentado varias formas de medir la eficacia de los distintos métodos de resecuenciación. En este trabajo proponemos una función objetivo con la siguiente estructura:

$$Z = \alpha \cdot Mn + (1 - \alpha) \cdot In \tag{1}$$

Mn representa la medida de eficacia estándar de secuenciación (*Makespan normalizado*), *In* la de estabilidad y α es un factor de peso que viene a modelizar las preferencias del decisor.

Las dos medidas de eficacia y estabilidad se han normalizado entre 0 y 1 para evitar los problemas de escala que se puedan generar por los distintos órdenes de magnitud de ambos objetivos (*Makespan* y estabilidad).

$$Mn = \frac{C_{máx} - \min(C_{máx})}{máx(C_{máx}) - \min(C_{máx})} \tag{2}$$

$C_{máx}$ indica el *Makespan* de la secuencia después de la resecuenciación, $\min(C_{máx})$ y $máx(C_{máx})$ son respectivamente cotas inferiores y superiores calculadas para cada evento en cada punto de resecuenciación.

La cota inferior se ha calculado de la siguiente manera: En cada punto de resecuenciación tenemos una lista de trabajos ya secuenciados y una lista de trabajos para secuenciar. Se coge cada uno de los trabajos de la segunda lista y se secuencian al final de la primera, insertando cada uno de ellos en todas las posibles posiciones. Nos quedamos con la permutación parcial que da el menor valor del *Makespan*. Luego se suman a este valor de *Makespan* los tiempos de proceso de todos los trabajos restantes en la última máquina.

El cálculo de la cota superior es también fácil: sin cambiar el orden de los trabajos, se calcula el *Makespan* teniendo en cuenta que la primera operación de cada trabajo no puede empezar antes de la finalización de la última operación del trabajo anterior.

Veamos ahora la expresión elegida para medir la estabilidad.

$$In = \frac{Q - \min(Q)}{\max(Q) - \min(Q)} \tag{3}$$

En (3) $\min(Q)$ es una cota inferior para el número de operaciones cuyos tiempos de inicio sufren retrasos o adelantos y $\max(Q)$ es la cota superior. Es evidente que $\min(Q)$ es igual a 0, es decir ninguna operación se mueve, y $\max(Q)$ es igual a $m \cdot n$, es decir, todas las operaciones se desplazan en el peor caso posible. De manera más precisa, Q se calcula de la siguiente manera:

$$Q = \sum_{j=1}^{n} \sum_{i=1}^{m} U_{ji} \tag{4}$$

$$U_{ji} = \begin{cases} 1, & \text{si } \left| s_{ji}^{*} - s_{ji} \right| > 10 \\ 0, & \text{si } \left| s_{ji}^{*} - s_{ji} \right| \leq 10 \end{cases} \tag{5}$$

En (5) s_{ji}^{*} indica el nuevo tiempo de inicio del trabajo j en la máquina i después de la resecuenciación y s_{ji} el tiempo de inicio antes de la resecuenciación, es decir, el tiempo de inicio del trabajo en la secuencia original. En (5) el valor 10 se refiere al umbral por debajo del cual no es relevante considerar la operación como desplazada, es decir, si la diferencia entre su nuevo tiempo de inicio y el tiempo de inicio original es inferior a 10 unidades de tiempo, no se considera que afecte a la estabilidad de la secuencia.

2.2. Métodos de resecuenciación

En este trabajo se han implementado cuatro métodos de resecuenciación: reparación, búsqueda local de inserción, búsqueda local de inserción hasta óptimo local y el algoritmo voraz iterativo IG descrito en Ruiz y Stützle (2007). Para poder comparar los distintos métodos implementados se lanzan todos ellos en cada punto de resecuenciación, de esta forma todos los algoritmos se enfrentan con idénticas condiciones en el taller. La mejor solución de todas se selecciona como solución del actual evento para lidiar con el siguiente evento. Con este método, todos los algoritmos resuelven el mismo problema en cada evento y ninguno se encuentra con condiciones del taller más o menos favorables.

La principal ventaja obtenida por esta metodología es evitar hacer numerosas simulaciones por cada método y luego comparar los resultados dados por todos los métodos individualmente.

Cada vez que ocurre un evento, es decir, en cada instante de tiempo en el que es necesario resecuenciar, se parte la secuencia original en tres partes:

- Subsecuencia que no se ve afectada por el evento que se está procesando. Estará formada por todos aquellos trabajos que ya han finalizado todas sus tareas en el taller.
- Subsecuencia que se tiene que actualizar sin poder cambiar el orden de los trabajos. Estará formada por todos aquellos trabajos donde alguna de sus tareas ha comenzado ya, es decir, trabajos que están en curso (recordemos que estamos en el taller de flujo de permutación).
- Subsecuencia cuyos trabajos todavía no han empezado y que se puede cambiar y optimizar para acomodar el evento. Los algoritmos propuestos trabajan sobre ésta tercera secuencia parcial.

En la Figura 1 podemos observar las tres subsecuencias parciales que se pueden extraer a partir del evento de rotura de la máquina 4 que sucede en el instante de tiempo 300. Los trabajos 3, 1 y 5 no se ven afectados, ya que están finalizados para cuando ocurre el evento. Los trabajos 8 y 2 formarían la subsecuencia que se tiene que actualizar, ya que están en curso en el instante 300. Por tanto, será necesario recalcular los tiempos de inicio y fin solo en las máquinas 4 y 5. La permutación parcial que va del trabajo 4 al 9 se puede reordenar y optimizar en función del nuevo evento ya que ninguno de estos trabajos ha empezado a procesarse en el taller (subsecuencia de trabajos no empezados). Esta permutación parcial será sobre la que trabajarán los métodos de resecuenciación propuestos en este trabajo.

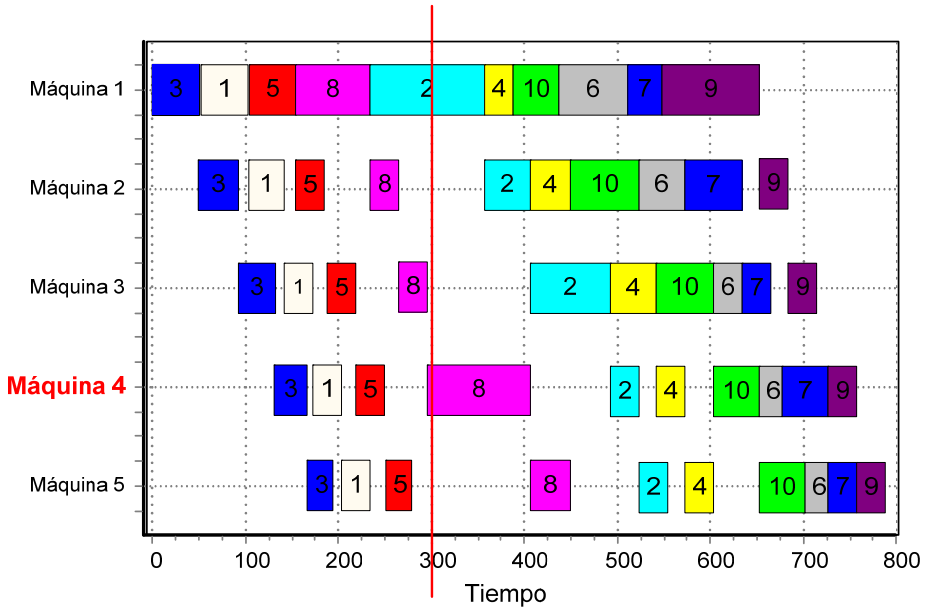

Figura 1: Diagrama de Gantt de una secuencia sujeta a un evento de rotura de máquina.

A continuación pasamos a describir brevemente cada uno de los métodos implementados.

2.2.1. Reparación

La reparación es la acción comúnmente utilizada en plantas reales de fabricación para actualizar la secuencia programada frente a eventos inesperados. Cuando se rompe una máquina, se inserta el intervalo de rotura en el tiempo de proceso del trabajo en la máquina averiada y se actualizan los tiempo de inicio y fin de las operaciones de todos los trabajos afectados.

Cuando llegan nuevos trabajos al taller, se insertan directamente al final de la secuencia ya programada, imitando el proceder habitual de las industrias. Por último, para el tercer evento considerado en este trabajo, si algún trabajo se retrasa, se posponen los inicios de las operaciones de todos los trabajos afectados.

2.2.2. Búsqueda local de inserción (BL)

Este mecanismo aplica un procedimiento de búsqueda local basado en el vecindario de inserción, ya que es el que mejor resultados da para este tipo de problemas de secuenciación (Ruiz, Maroto y Alcaraz, 2006; Ruiz y Stützle, 2007) .Todos los trabajos de la permutación parcial se insertan en todas las posibles posiciones y al final nos quedamos con la permutación que da el menor valor de la función objetivo comentada anteriormente.

2.2.3. Búsqueda local de inserción hasta mínimo local (BLMinLoc)

Se repite el procedimiento anterior hasta llegar a un óptimo local con respecto al vecindario de inserción, es decir, si mejoramos el valor de la función objetivo al insertar uno o más trabajos en una posición distinta, se vuelve a repetir el proceso de insertar todos los trabajos en todas las posiciones. El proceso solo para cuando después de haber insertado todos los trabajos en todas las posibles posiciones, no se mejora el valor de la función objetivo.

2.2.4. Algoritmo voraz iterativo (IG) de Ruiz y Stützle (2007)

El tercer método de resecuenciación propuesto está basado en un algoritmo muy eficaz para el problema del taller de flujo de permutación con el objetivo de minimizar el makespan. Dicho algoritmo es el algoritmo voraz iterativo (IG) propuesto por Ruiz y Stützle (2007). El funcionamiento del algoritmo, en su versión de resecuenciación, es muy sencillo: En una primera fase se aplica la búsqueda local de inserción a la permutación parcial para obtener una buena secuencia de partida. En una segunda fase, llamada fase de destrucción, se eliminan aleatoriamente uno o más trabajos de la permutación. En la última fase, llamada de reconstrucción, cada elemento eliminado se inserta en todas las posiciones de la permutación parcial, buscando minimizar el objetivo ponderado y normalizado.

3. Experimento computacional y resultados

Para comparar los resultados de las diferentes técnicas de resecuenciación se ha usado un subconjunto de las instancias de Taillard (1993) con los eventos generados en la fase inicial tal y como se han descrito en la sección 2. En concreto, y para no elevar de manera exagerada los tiempos de computación, se han elegido todas las instancias de hasta 100 trabajos y 20 máquinas. Cada vez que se produce un evento, se lanzan los cuatro métodos anteriores y la mejor solución se guarda como solución actual (solución de resecuenciación), que se verá afectada por el siguiente evento. Todos estos métodos han sido desarrollados en Delphi 2007 y ejecutados en ordenadores Core 2 Duo, 2,4 GHz con 2 GB de RAM. Como criterio de parada para el IG se ha elegido el tiempo total de CPU calculado siguiendo la expresión $n_p \cdot (m/2) \cdot t$ milisegundos, donde n_p representa el número de trabajos de cada permutación parcial, m el número de máquinas y t se ha fijado a 150. De esta manera permitimos más tiempo de CPU a aquellas instancias con un mayor número de trabajos y/o máquinas. También se permite más tiempo para reoptimizar secuencias que tienen más trabajos por empezar.

En todas las instancias estudiadas, el tiempo de cálculo en el proceso de reparación es inferior a un segundo. La búsqueda local de inserción tarda alrededor de un segundo en la instancias con 100 trabajos y 20 maquinas y menos de un segundo en el resto. La búsqueda local hasta mínimo local tarda alrededor de tres segundos en las instancias más grandes y menos de un segundo en las restantes. Por tanto, los métodos propuestos, a excepción del IG, son muy rápidos y eficientes, lo que los hace adecuados para su uso en entornos reales de fabricación.

En cada uno de los puntos de resecuenciación se ha calculado el incremento porcentual sobre la mejor solución encontrada por cada método mediante la expresión (6)

$$IP = \frac{Z_{sol} - \min(Z)}{\min(Z)} \cdot 100 \qquad (6)$$

Z_{sol} representa el valor de la función objetivo obtenida por el método de resecuenciación y min(Z) indica el menor valor encontrado por todos los métodos en ese evento.

En las tablas (2), (3), (4) se presenta el promedio de los valores del *IP* (IPM) obtenidos en todos los puntos de resecuenciación por tamaño de instancia.

Cada uno de los tres experimentos mostrados utiliza un valor distinto del parámetro α que recordemos que es el factor de ponderación de los dos objetivos de estudio: makespan normalizado y estabilidad (ver expresión (1)). La tabla 2 corresponde a un experimento donde se le da mucha importancia a la minimización del makespan normalizado (valor de α muy elevado) y por tanto muy poca a la estabilidad. La tabla 3 corresponde a justo el experimento opuesto, es decir, mucha importancia al objetivo de estabilidad. En la tabla 2 observamos el resultado para un experimento donde se da la misma importancia a ambos objetivos (α =0,5). De esta manera podemos ver cómo se comportan los distintos métodos en función del peso que se le da a cada objetivo.

	Reparación	BL	BLMinLoc	IG
Ta20×5	60,12	3,58	1,48	0,00
Ta20×10	30,33	6,93	3,15	0,00
Ta20×20	28,49	4,51	2,02	0,00
Ta50×5	118,25	19,14	7,39	0,00
Ta50×10	93,08	22,17	6,64	0,00
Ta50×20	37,67	8,29	3,92	0,00
Ta100×5	181,94	10,71	4,32	0,00
Ta100×10	119,67	11,68	4,57	0,00
Ta100×20	84,95	14,34	4,19	0,00
Promedio	83,83	11,26	4,19	0,00

Tabla 2: Incremento porcentual medio sobre la mejor solución encontrada (IPM) α=0.9.

	Reparación	BL	BLMinLoc	IG
Ta20×5	42,59	4,03	2,16	0,00
Ta20×10	31,80	10,20	6,35	0,00
Ta20×20	32,67	8,92	5,83	0,00
Ta50×5	113,88	16,44	7,56	0,00
Ta50×10	72,81	17,00	5,32	0,00
Ta50×20	55,72	9,69	4,13	0,00
Ta100×5	221,36	13,09	5,62	0,00
Ta100×10	175,95	16,48	7,81	0,00
Ta100×20	129,43	13,64	3,32	0,00
Promedio	97,36	12,17	5,34	0,00

Tabla 3: Incremento porcentual medio sobre la mejor solución encontrada (IPM) α=0.5.

	Reparación	BL	BLMinLoc	IG
Ta20×5	46,70	4,57	2,48	0,00
Ta20×10	38,27	11,56	7,63	0,00
Ta20×20	38,45	8,96	5,85	0,00
Ta50×5	157,28	14,92	4,53	0,00
Ta50×10	108,04	27,33	8,76	0,00
Ta50×20	50,54	11,53	5,31	0,00
Ta100×5	323,74	17,69	8,84	0,00
Ta100×10	223,46	22,20	7,23	0,00
Ta100×20	118,50	16,74	4,36	0,00
Promedio	122,78	15,06	6,11	0,00

Tabla 4: Incremento porcentual medio sobre la mejor solución encontrada (IPM) α=0.1.

Como se puede ver en todas la tablas, el algoritmo basado en el IG ha proporcionado siempre la mejor solución en todos los eventos y para todos los valores de α, es decir, los resultados son los mejores con independencia del peso que tenga cada uno de los objetivos. IG proporciona los mejores resultados tanto si se le da más peso al makespan normalizado, como a la estabilidad como el mismo peso a los dos. Esta característica es muy importante para este tipo de algoritmos, ya que diseñar un buen algoritmo que sólo se comporte de manera eficaz para uno de los objetivos estudiados sería de poca utilidad en la práctica. La reparación, típicamente usada en plantas reales puede dar soluciones hasta casi un 324% peores que los resultados del IG (Tabla 4). También cabe destacar que con una búsqueda local muy sencilla se pueden obtener resultados bastante buenos y mucho mejores que con la reparación estándar.

4. Conclusiones

En este trabajo hemos estudiado el problema de la resecuenciación de trabajos para el caso del problema del taller de flujo de permutación. Se ha trabajado con tres posibles eventos de resecuenciación con el objetivo de minimizar dos objetivos ponderados mediante un parámetro α. Por un lado, el makespan normalizado y por otro la estabilidad normalizada de la secuencia una vez resecuenciada. Hemos generado un banco de pruebas con tres tipos de eventos para un subconjunto de las instancias de Taillard (1993). Hemos comparado la reparación (método más utilizado en la práctica) con tres métodos de resecuenciación: dos de ellos basados en búsqueda local en el vecindario de inserción y el tercero basado en el algoritmo IG de Ruiz y Stützle (2007). Los resultados obtenidos han podido demostrar que la reparación es el peor método para todos los valores de α. El método basado en el algoritmo IG obtiene los mejores resultados en todos los casos, para todos los valores de α fijados en los experimentos. Como trabajo futuro se pretenden considerar más tipos de eventos de manera simultánea, otras políticas de resecuenciación y procedimientos que obtengan resultados de tipo frontera de Pareto para objetivos relacionados con la estabilidad y eficacia.

Bibliografía

Abumaizar, R. J., and Svestka, J. A., 1997. Rescheduling job shops under disruptions. International Journal of Production Research, Vol. 35, pp. 2065-2082.

Adhitya, A.; Srinivasan, R.;Karimi, I. (2007). A model-based rescheduling framework for managing abnormal supply chain events. Computers and Chemical Engineering Vol. 31, pp. 496-518.

Arnaout, J.P.; Rabadi, G. (2008). Rescheduling of unrelated parallel machines under machine breakdowns. International Journal of Applied Management Science. Vol. 1, pp. 75-89

Aytug, H.; Lawley, M.A.; McKay, K.; Mohan, S.; Uzsoy, R. (2005). Executing production schedules in the face of uncertainties: a review and some future directions. European Journal of Operational Research, Vol. 161, pp. 86-110.

Church, L.K. ; Uzsoy, R. (1992). Analysis of periodic and event-driven rescheduling policies in dynamic shops. International Journal of Computer Integrated Manufacturing Vol. 3, pp. 153-163.

Hall, N.G.; Potts, N. C. (2004). Rescheduling for new orders. Operations Research, Vol. 52, pp. 440-453.

Jain, A. K.; Elmaraghy, H. A., (1997). Production scheduling/rescheduling in flexible manufacturing. International Journal of Production Research, Vol. 35, pp. 281-309.

Kim, M. H.; Kim, Y.D., (1994). Simulation-based real-time scheduling in a flexible manufacturing system. Journal of Manufacturing Systems, Vol. 13, pp. 85-93.

Liao, C.J.; Chen, W. J. (2004). Scheduling under machine breakdowns in a continuous process industry. Computers Operations Research, Vol. 31, pp. 415-428

Moratori, P.; Petrovic, S.; Vázquez, A. (2008).Match-Up Strategies for Job Shop Rescheduling. Lecture Notes In Artificial Intelligence; Vol. 5027, pp. 119-128

Pfeiffer, A; Kadar, B; Monostori, L. (2007). Stability-oriented evaluation of rescheduling strategies, by using simulation. Computers in Industry. Vol. 58, pp. 630-643.

Rangsaritratsamee, R.; Ferrell, W.G.; Kurz, M.B (2004). Dynamic rescheduling that simultaneously considers efficiency and stability. Computers and Industrial Engineering. Vol. 46, pp. 1-15.

Ruiz, R.; Maroto, C.; Alcaraz, J. (2006). Two new robust genetic algorithms for the flowshop. The International Journal of Management Science. Vol. 34, pp. 461-476

Ruiz, R.; Stützle, T. (2007). A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. European Journal of Operational Research, Vol. 177, pp. 2033-2049.

Subramaniam, V.; Raheja, A. (2003). mAOR: A heuristic-based reactive repair mechanism for job shop schedules. The International Journal of Advanced Manufacturing Technology. Vol. 22, pp. 669-680.

Taillard, E. (1993). Benchmarks for basic scheduling problems. European Journal of Operational Research, Vol. 64, pp. 278-285.

Vieira, G.; Herrman, J.; Lin, E. (2000). Predicting the performance of rescheduling strategies for parallel machine systems. Journal of manufacturing systems, Vol. 19, pp. 256-266.

Vieira, G.; Herrman, J.; Lin, E. (2003). Rescheduling manufacturing systems: a framework of strategies, policies and methods. Journal of scheduling. Vol. 6, pp. 39-62.

Vollmann, T. E.; William, L. B.; Whybark, D.C.(1997) Manufacturing Planning and Control Systems, fourth edition, Irwin/McGraw-Hill, New York.

Wu, S.D.; Storer, R.H.; Chang, P.C. (1993) One-machine rescheduling heuristics with efficiency and stability as criteria. Computers and Operations Research. Vol. 20**,** pp. 1-14**.**

Yamamoto M.; Nof S.Y. (1995). Scheduling rescheduling in the manufacturing operating system environment. International Journal of Production Research. Vol.23, pp. 705-722.