ISSN 0717-9103

HEURÍSTICA CONSTRUCTIVA VISIONARIA PARA EL PROBLEMA DE MÁQUINAS PARALELAS NO RELACIONADAS CON TIEMPOS DE SETUP DEPENDIENTES DE LA SECUENCIA

LOOK-AHEAD CONSTRUCTIVE HEURISTIC FOR THE UNRELATED PARALLEL MACHINE PROBLEM WITH SEQUENCE DEPENDENT SETUP TIMES

FELIPE TOMÁS MUÑOZ VALDÉS [1]
REINALDO JAVIER MORAGA SUAZO [2]

Departamento de Ingeniería Industrial, Universidad del Bío-Bío, Chile
Department of Industrial & Systems Engineering. Northern Illinois University, USA

RESUMEN

Este artículo presenta una heurística constructiva visionaria para resolver el problema de máquinas paralelas no relacionadas con tiempos de setup dependientes de la secuencia, donde el objetivo es minimizar el tiempo de completación del último trabajo, o makespan. Este es un problema NP-hard. A pesar de que una gran variedad de situaciones prácticas pueden ser modeladas usando este problema, existe una pequeña cantidad de artículos desarrollados que se puede encontrar en la literatura. La heurística constructiva propuesta en este artículo posee un mecanismo visionario basado en un criterio de ahorro, que permite mejorar la calidad de las soluciones que se obtienen al comparar este enfoque con otras reglas heurísticas encontradas en la literatura.

Palabras clave: heurísticas constructivas, makespan, scheduling, máquinas paralelas no relacionadas.

ABSTRACT

This paper presents a look-ahead constructive heuristic for solving the unrelated parallel machine problem with sequence dependent setup times, where the objective is to minimize the completion time of the last job, or makespan. This is an NP-hard problem. Although a variety of practical situations can be modeled using this problem, still a modest number of articles devoted to it can be found in literature. The constructive heuristic proposed in this paper has a look-ahead mechanism based on a saving criterion, which permits to improve the quality of solutions against another constructive heuristic rule already reported in literature.

Keywords: constructive heuristics, makespan, scheduling, unrelated parallel machines.

[1] Departamento de Ingeniería Industrial, Universidad del Bío-Bío. Avenida Collao 1202, Casilla 5C, Concepción, Chile. E-mail: fmunoz@ubiobio.cl.
[2] Department of Industrial & Systems Engineering. Northern Illinois University. De Kalb, Illinois 60115. Illinois. USA. E-mail: moraga@ceet.niu.edu.

1. INTRODUCCIÓN

En este artículo se presenta una heurística constructiva visionaria (look-ahead) para resolver el problema de máquinas paralelas no-relacionadas con tiempos de setup dependientes de la secuencia, donde el objetivo es minimizar el tiempo de completación del último trabajo (makespan). El principal objetivo en este trabajo es desarrollar una heurística constructiva para encontrar buenas soluciones para el problema en estudio. Esta solución debe ser obtenida rápidamente para poder aplicarse a problemas reales.

Desarrollar heurísticas constructivas es necesario porque el funcionamiento de muchas metaheurísticas requiere de soluciones iniciales, las cuales pueden ser entregadas por una heurística constructiva.

Este artículo se organiza de la siguiente manera: en la sección 2 se presenta una revisión bibliográfica, en la sección 3 se realiza una descripción del problema en estudio. En la sección 4 se presenta la heurística constructiva propuesta y un ejemplo numérico de su uso, en la sección 5 se presentan los resultados computacionales. Finalmente en la sección 6 se presentan las conclusiones.

2. REVISIÓN BIBLIOGRÁFICA

Casos prácticos de problemas de máquinas paralelas son muy comunes en la vida real. En la literatura se pueden encontrar aplicaciones en las áreas de producción de textiles (Guinet, 1991), transporte (Guinet, 1993), programación de actividades de oficina (Herrmann *et al.,* 1997), metalurgia de aluminio (Dhaenens-Flipo, 2001), producción de semiconductores (Kim *et al.,* 2002), producción de plásticos por moldeo (Lin *et al.,* 2002), producción de placas de circuito impresas (Yu *et al.,* 2002).

Revisiones del estado del arte sobre problemas de programación de la producción en máquinas paralelas pueden ser encontradas en Graves (1981), Cheng & Sin (1990), Lawler *et al*. (1993), y más recientemente en Mokotoff (2001). En general, las investigaciones que consideran el problema $R_m|S_{ijk}|C_{max}$ son escasas. Sin embargo la literatura relacionada con variantes del problema es bastante extensa. Algunas de estas investigaciones han considerado la restricción de tiempos de setup dependientes de la secuencia (Franca *et al.,* 1996; Lee & Pinedo, 1997; Anagnostopoulos & Rabadi, 2002; Helal & Hosni, 2003; Al-Salem, 2004; Rabadi *et al.,* 2006).

Anagnostopoulos & Rabadi (2002) proponen un algoritmo de Simulated Annealing para el problema de máquinas paralelas no-relacionadas con tiempos de setup dependientes de la secuencia, minimizando el makespan. La solución de partida que usa este algoritmo es obtenida en forma aleatoria. Helal & Hosni (2003), estudiaron el mismo problema y desarrollan un algoritmo basado en Tabú Search y proponen la utilización de la regla SPT (shortest procesing time) para construir una solución inicial, debido a la inexistencia a esa fecha de una heurística constructiva para el problema en consideración.

Al-Salem (2004), propone un enfoque de solución basado en particionamiento, el cual consiste en tres fases. La primera de ellas es una fase constructiva que asigna trabajos a las máquinas según una regla simple, la segunda fase es una fase de mejoramiento al resultado entregado en la primera fase. La última fase es donde se realiza el secuenciamiento de trabajos, basándose en técnicas para resolver el problema del agente viajero (TSP).

Recientemente, Rabadi *et al.*(2006) propone la aplicación de Meta-RaPS para resolver el problema en estudio en este trabajo. Adicionalmente propone una heurística constructiva glotona, llamada SAP-SL, para obtener soluciones de partida.

3. DESCRIPCIÓN DEL PROBLEMA

El problema de máquinas paralelas no relacionadas (R_m) consiste en *n* trabajos disponibles en el tiempo cero que deben ser programados sin interrupción en *m* máquinas en paralelo que pueden procesar trabajos con tiempos de procesamiento arbitrarios. Cada trabajo es asignado a una máquina y cada máquina puede procesar un trabajo a la vez. Cuando un trabajo va a ser procesado en una máquina, es necesario preparar ésta, por lo que se requiere un tiempo de setup. Esto significa que si el trabajo *j* es programado en la máquina *k*, el tiempo necesario para procesar ese trabajo es P_j^k, que depende del trabajo *j*, y la máquina *k*. Adicionalmente, el problema considera tiempos de setup dependientes de la secuencia (S_{ij}^k), donde el tiempo necesario de setup para procesar el trabajo *j* después del trabajo *i* en la máquina *k* puede ser diferente de aquel que se necesita para procesar el trabajo *i* después del trabajo *j* en la misma máquina. La idea es encontrar el mejor programa de producción que minimice el tiempo de completación del último trabajo (también llamado makespan o C_{max}). En términos de la notación de tres parámetros propuesta por Graham *et al.* (1979) el problema en estudio en este trabajo puede ser representado por la tripleta $R_m|S_{ij}^k|C_{max}$.

El problema de programar trabajos en máquinas paralelas idénticas minimizando el makespan es un conocido problema NP-hard, aún cuando se consideran 2 máquinas (Pinedo, 2002). Por lo tanto como el problema en consideración en este estudio es una generalización de aquel problema básico, éste debe ser también NP-hard. El modelo matemático del problema, propuesto por Muñoz *et al.* (2005), se presenta a continuación:

$$\text{Minimizar } C_{\max} \tag{1}$$

Sujeto a

$$\sum_{\substack{i=0 \\ i\neq j}}^{n}\sum_{k=1}^{m} X_{ij}^k = 1; \qquad \forall\, j = 1,\ldots,n \tag{2}$$

$$\sum_{j=1}^{n} X_{0j}^k = 1; \qquad \forall\, k = 1,\ldots,m \tag{3}$$

$$\sum_{i=1}^{n} X_{i0}^k = 1; \qquad \forall\, k = 1,\ldots,m \tag{4}$$

$$\sum_{\substack{i=0 \\ i\neq h}}^{n} X_{ih}^k = \sum_{\substack{j=0 \\ j\neq h}}^{n} X_{hj}^k; \qquad \forall\, h = 1,\ldots,n \quad k = 1,\ldots,m \tag{5}$$

$$WL^k = \sum_{i=0}^{n}\sum_{\substack{j=1 \\ j\neq i}}^{n} (P_j^k + S_{ij}^k) X_{ij}^k; \qquad \forall\, k = 1,\ldots,m \tag{6}$$

$$C_{\max} \geq WL^k; \qquad \forall\, k = 1,\ldots,m \tag{7}$$

$$WL^k \geq 0; \qquad \forall\, k = 1,\ldots,m \tag{8}$$

$$X_{ij}^k \in \{0,1\}; \qquad \forall\, i = 1,\ldots,n \quad j = 1,\ldots,n \quad k = 1,\ldots,m \tag{9}$$

En el modelo anterior, *n* es la cantidad de trabajos, *m* es la cantidad de máquinas, WL^k es la carga de trabajo en la máquina *k*, P_j^k es el tiempo de procesamiento del trabajo *j* en la máquina *k*, S_{ij}^k es el tiempo de setup dependiente de la secuencia para procesar el trabajo *j* después del trabajo *i* en la máquina *k*, S_{0j}^k es el tiempo de setup inicial si se procesa el trabajo *j* primero en la máquina *k*. Las variables de decisión son:

X_{ij}^k = 1 si el trabajo *j* es procesado directamente después del trabajo *i* en la máquina *k*, 0 en otro caso,
X_{0j}^k = 1 si el trabajo *j* es el primer trabajo procesado en la máquina *k*, 0 en otro caso,
X_{i0}^k = 1 si el trabajo *i* es el último trabajo procesado en la máquina *k*, 0 en otro caso.

La función objetivo (1) es minimizar el tiempo de completación del ultimo trabajo (makespan); el grupo de restricciones (2) asegura que cada trabajo es programado solamente una vez y es procesado por una máquina; los grupos (3) y (4) aseguran que cada máquina tiene sólo un trabajo inicial y un trabajo final respectivamente; el grupo (5) asegura que cada trabajo tiene un predecesor y un sucesor en la máquina. El grupo de restricciones (6) determina la carga de trabajo en cada máquina. El grupo (7) asegura que el makespan sea igual a la máxima carga de trabajo entre todas las máquinas. El grupo (8) asegura la no-negatividad de las cargas de trabajo. Mientras que el grupo (9) especifica que cada variable de decisión X_{ij}^k es binaria.

4. HEURÍSTICA CONSTRUCTIVA LOOK-AHEAD PARA $R_m|S_{ijk}|C_{max}$.

La heurística constructiva visionaria propuesta (llamada LACH, Look-Ahead Constructive Heuristic) se basa en criterios de ahorro. En la cual se asignan y secuencian iterativamente los trabajos a las máquinas, de manera de aumentar las cargas de trabajo de las máquinas lo menos posible. Lo cual tiene como resultado un programa de producción que logra un mejor balance de cargas de trabajo en las máquinas y un menor *makespan o* C_{max}.

Como definimos anteriormente, sea *m* la cantidad de máquinas y *n* la cantidad de trabajos. Denotaremos por P_j^k al tiempo de procesamiento del trabajo *j* en la máquina *k*, y a S_{ij}^k como el tiempo de setup que necesita la máquina *k* para procesar el trabajo *j* si anteriormente se realiza el trabajo *i*.

Para abordar el problema $R_m|S_{ijk}|C_{max}$, se utilizará el concepto de *Matriz de Tiempos de Procesamiento Ajustado* [AP^k] para cada máquina *k* propuesto por Rabadi *et al.* (2006) (Figura N°1). En general las *Matrices de Tiempos de Procesamiento Ajustado* [AP^k] para cada máquina se construyen utilizando la siguiente ecuación:

$$ap_{ij}^k = S_{ij}^k + P_j^k;\ \forall k = 1,...,m;\ \forall i = 1,...,n;\ \forall j = 1,...,n. \qquad (10)$$

Donde:
ap_{ij}^k: tiempo de procesamiento ajustado, es decir, al tiempo necesario para realizar el trabajo *j* después del trabajo *i* en la máquina *k*. Agrupa el tiempo de procesamiento y el tiempo de setup de la máquina.

Con lo anterior se pueden construir *m* matrices de orden $(n+1)\times n$, que contienen los tiempos de procesamiento ajustado, de tal forma que el elemento *(i,j)-ésimo* de la matriz *k* sea el tiempo de procesamiento ajustado si se realiza el trabajo *j* después del trabajo *i* en la máquina

k. En esas matrices, los elementos de la primera fila, llamada fila cero (ap_{0j}^{k}), indican el tiempo necesario para procesar el trabajo *j* como el primero en el programa de producción de la máquina *k*.

En la Figura N°2 se muestra un diagrama de la heurística constructiva LACH.

Dada las características de la heurística, su aplicación estará limitada para problemas de tamaño pequeño, es decir, la heurística puede ser aplicada a problemas de gran tamaño. Específicamente a los problemas donde se cumpla la inecuación $n \geq 3m$, que indica que la cantidad de trabajos del problema debe ser mayor o igual a tres veces la cantidad de máquinas.

A continuación se describe cada parte del diagrama detalladamente.

Parte 1: Asignar inicialmente una secuencia de dos trabajos a cada máquina.

Esta es la parte inicial de la heurística, y su objetivo es asignar una secuencia de dos trabajos a cada una de las *m* máquinas. Esa secuencia inicial no indica que el primer trabajo sea el trabajo inicial en el programa de producción. Para ello es importante hacer una diferencia entre el programa de producción final y el programa de producción parcial. Por lo tanto al finalizar esta parte, cada máquina tendrá un programa parcial compuesto por dos trabajos. En las siguientes etapas de la heurística, ese programa parcial de cada máquina será completado, de tal manera que los demás trabajos serán secuenciados antes del *primer* trabajo y después del *segundo* (último) trabajo del programa parcial.

La variable *L* es un contador de los trabajos que han sido programados. Por lo tanto, cuando se finalice esta etapa, *L* tendrá un valor igual a 2*m* (dos trabajos para cada máquina).

Un criterio glotón para seleccionar los dos trabajos del programa parcial asignado a cada máquina, podría considerar solamente que el tiempo de procesamiento ajustado asociado sea mínimo, por lo que bastaría buscar en las matrices de tiempos de procesamiento ajustado aquel ap_{ij}^{k} menor para una determinada máquina *k*. Sin embargo, utilizar ese criterio podría llevar a programas parciales muy malos. Eso se debe principalmente a que no se considera que trabajo puede ser asignado antes del trabajo *i* o después del trabajo *j*. Para considerar esto LACH utiliza un estimador, que indica que tan prometedora es la asignación de dos trabajos en una máquina en términos de aumentar en mínima cantidad su carga de trabajo. En la Figura N°3 se muestra gráficamente una evaluación de dos trabajos (*x* e *y*), donde *F(x)* y *F(y)* son estimadores del tiempo de procesamiento ajustado de los trabajos que podrían asignarse antes del trabajo *x,* y después del trabajo *y* respectivamente.

El estimador, llamado *cota inferior* se calcula mediante la siguiente expresión:

$$LB(x,y) = \min_{i \in J}\{ap_{i\,x}\} + ap_{x\,y} + \min_{j \in J}\{ap_{y\,j}\} \quad (11)$$

De donde se puede observar que los estimadores *F(x)* y *F(y)* son:

$$F(x) = \min_{i \in J}\{ap_{i\,x}\} \quad (12)$$

$$F(y) = \min_{j \in J}\{ap_{y\,j}\} \quad (13)$$

LACH utiliza los estimadores *F(x)* y *F(y)* para evitar la evaluación de una gran cantidad de combinaciones que se pueden generar. Si *n* indica la cantidad de trabajos y *m* la cantidad de máquinas, la cantidad de combinaciones a evaluar sin el uso de los estimadores (búsqueda exhaustiva) sería del orden $O(n^4m)$, para *n* suficientemente grande. Luego, mediante el uso de los estimadores *F(x)* y *F(y)*, la búsqueda se reduce a un orden $O(n^2m)$, para *n* suficientemente grande.

Supongamos un problema donde n=100 y m=10. Mediante búsqueda exhaustiva de combinaciones se deberían probar 941.094.000 combinaciones, mediante el uso del estimador esa búsqueda se reduce a 99.000, lo cual es significativamente menor.

Los estimadores *F(x)* y *F(y)* se obtienen buscando el menor elemento de la columna *x* y el menor elemento de la fila *y* en la matriz $[AP^k]$, sin considerar la fila cero. Esto es similar a buscar que trabajo programar antes de *x,* y después del trabajo *y* respectivamente, aumentando en menor cantidad la carga de trabajo de la máquina *k*.

En resumen, esta parte de la heurística consiste en asignar un par de trabajos mediante el estimador *LB* a cada máquina del problema.

Parte 2: Ordenar máquinas en orden descendente de cargas de trabajo.

Esta parte consiste simplemente en ordenar las máquinas en forma descendente según la carga de trabajo asociada a su programa parcial, de tal forma que la primera de la lista sea la que tiene la mayor carga de trabajo, y la última máquina de la lista sea la que tiene la menor carga de trabajo.

La idea fundamental de esta parte es ordenar los trabajos según una prioridad, de tal forma que la máquina que tenga la mayor carga de trabajo, es aquella que tiene mayores posibilidades de aumentar el makespan si se le asigna un trabajo adicional. Luego, la máquina que tiene la siguiente mayor carga de trabajo tiene la siguiente mayor prioridad y así sucesivamente. De tal forma que la máquina que tenga la menor carga de trabajo aumentará en menor cantidad el makespan si se le asigna un trabajo adicional. Como se explicó en la Parte 1, ese trabajo puede asignarse antes del primer o después del último trabajo en la secuencia parcial.

El orden establecido en esta parte se utiliza para realizar el proceso de reserva de trabajos (Parte 3) y el proceso de asignación de trabajo (Parte 4).

Parte 3: Para todas las máquinas excepto la última, reservar el trabajo que incrementa en menor cantidad su carga de trabajo.

Para explicar más fácilmente este paso, consideraremos que la notación de una secuencia parcial de trabajos en cada máquina estará dada por [*X-...-Y*], donde *X* e *Y* representan el primero y último trabajo respectivamente en la secuencia parcial de cada máquina. La heurística asigna o reserva trabajos al comienzo o al final de la secuencia parcial [*X-...-Y*] hasta completar el programa de producción (secuencia de trabajos completa) en cada máquina.

Se denotará por *Z* a un trabajo que es asignado o reservado antes de *X* (es decir, *Z-X-...-Y*); y de denotará por *W* a un trabajo que es asignado o reservado después de *Y* (es decir, *X-...-Y-W*).

En esta parte de la heurística, se reservan trabajos utilizando el orden establecido en la Parte 2. De tal forma que la máquina que tiene la mayor carga de trabajo busca dentro de los trabajos factibles, aquellos trabajos *Z* o *W* que aumentasen en menor cantidad su carga de trabajo. Aquel trabajo *Z* o *W* se saca temporalmente de la lista de trabajos factibles. Esto se repite de manera sucesiva con la siguiente máquina en la lista hasta llegar a la última máquina, para la cual no se realiza el proceso de reserva. En ese momento se sigue con el siguiente paso (proceso de asignación).

El proceso de reserva consiste básicamente en utilizar la información disponible para reservar un trabajo, utilizando un criterio basado en *"el peor de los casos"*. La filosofía detrás de esto es: *"si se fuese a asignar un trabajo en una máquina en particular, ese trabajo debiese afectar lo menos posible su carga de trabajo, y con ello afectar lo menos posible el makespan"*. Esta filosofía unida al orden entregado por la lista creada en el Paso 2, permiten el uso del criterio basado en el peor de los casos.

Las dos formas de reservar trabajos se denotarán por P1 y P2. Donde P1 consiste en asignar un trabajo (*W*) después del último trabajo (*Y*) en el programa parcial, y P2 consiste en asignar un trabajo (*Z*) antes del primer trabajo (*X*) en el programa parcial. Luego el trabajo reservado se obtiene del menor valor (tiempo) asociado a P1 o P2. Esto es lo mismo que buscar el menor elemento en la columna *X* de la matriz [AP^k] para obtener *Z*, y buscar el menor elemento en la fila Y de la matriz [AP^k] para obtener W (sin considerar la fila cero de cada matriz).

En definitiva para cada máquina en el proceso de reserva es necesario evaluar la siguiente regla:

$$P_1 = ap_{0\,X} + ap_{Y\,W} \qquad W \text{ pertenece a los trabajos factibles no reservados}$$

$$P_2 = ap_{0\,Z} + ap_{Z\,X} \qquad Z \text{ pertenece a los trabajos factibles no reservados}$$

Si $P_1 < P_2$, se reserva el trabajo *W*
Si $P_1 > P_2$, se reserva el trabajo *Z*
En caso de empate, elegir arbitrariamente entre *W* o *Z* y reservarlo.

Parte 4: Para la última máquina, asignar el trabajo que incrementa en menor cantidad su carga de trabajo.

En esta parte, se considera solamente la máquina que tiene el programa de producción con la menor carga de trabajo, considerando como trabajos factibles los que no fueron reservados en la Parte 3. En esa máquina se realiza el proceso de asignación, que es exactamente igual que el proceso de reserva, con la salvedad que en esta parte el trabajo no es reservado, sino que asignado. Por lo tanto en esta parte de la heurística se altera el programa parcial de la máquina que tiene la menor carga de trabajo. Al terminar esta parte, el contador de trabajos asignados (*L*) debe incrementarse en 1, y los trabajos reservados en la Parte 3 son liberados, es decir, se devuelven al conjunto de trabajos factibles.

Parte 5: Eliminar máquina con la mayor carga de trabajo (el primero en la lista).

Esta parte se realiza siempre que la cantidad de trabajos factibles sea menor que la cantidad de máquinas por programar, y consiste en eliminar una máquina. La máquina eliminada es aquella que tiene la mayor carga de trabajo asociada al programa, de tal forma que al continuar el proceso se tendrán la misma cantidad de trabajos que máquinas.

Parte 6: Calcular y Reportar C_{max}

Esta parte se realiza cuando todos los trabajos han sido asignados, y consiste en calcular el tiempo de completación del último trabajo (makespan) y reportar el programa de producción de cada máquina.

Como se mencionó anteriormente, el diseño de LACH no permite resolver problemas de pequeño tamaño, específicamente los problemas donde no se cumpla la inecuación $n \geq 3m$. Esto se debe a que LACH comienza con la asignación de dos trabajos a cada máquina (Parte 1). Donde, si m es la cantidad de máquinas, se asigna una cantidad de $2m$ trabajos del total de trabajos del problema. Adicionalmente LACH requiere de un proceso completo de reserva y asignación de trabajos (Parte 3 y 4), por lo que se necesita como mínimo una cantidad de trabajos igual a la cantidad de máquinas (m). Por lo tanto para que LACH funcione se requiere de al menos $3m$ trabajos.

4.1 Algoritmo LACH.

Notación
m: cantidad de máquinas del problema.
n: cantidad de trabajos del problema.
MC^k: vector que contiene los mínimos elementos de cada columna en la matriz $[AP^k]$ para la máquina k sin considerar la fila cero. mc_i^k es el *i-ésimo* elemento en el vector MC^k.
MF^k: vector que contiene los mínimos elementos de cada fila en la matriz $[AP^k]$ para la máquina k sin considerar la fila cero. mf_j^k es el *j-ésimo* elemento en el vector MF^k.
ap^k_{ij}: es el *(i,j)-ésimo* elemento de la matriz $[AP^k]$ de tiempos de procesamiento ajustado para la máquina k.
WL^k: carga de trabajo en la máquina k.

Se considera que en caso de cualquier empate, ese se rompe arbitrariamente.
El conjunto *VM* contiene todas las máquinas disponibles a cargar con trabajos y s es la cantidad de máquinas disponibles a cargar. El conjunto *J* contiene los trabajos disponibles para programar y *L* es un contador de la cantidad de trabajos asignados en cualquier momento. Además *M*, *T* y *b* son variables de la heurística constructiva. El algoritmo de esta heurística constructiva visionaria se presenta en la Tabla Nº1.

4.2 Aplicación de LACH a un ejemplo numérico.

Se considerará un ejemplo de dos máquinas (m=2) y siete trabajos (n=7), donde los tiempos de procesamiento y setup de los trabajos se muestran en las Tablas N°2 y N°3 respectivamente. Como este problema cumple con la condición $n \geq 3m$, se puede aplicar LACH.

En la Tabla N°3, la fila cero indica el tiempo de setup de la máquina para realizar un trabajo, si anteriormente no se ha realizado ninguno. Por ejemplo, si el trabajo 3 es el primero en programarse en la máquina 1, el tiempo de setup de la máquina es 5 unidades de tiempo (de la Tabla N°3), y el tiempo de procesamiento de ese trabajo es 20 unidades de tiempo (de la Tabla N°2).

Con los datos presentados anteriormente, se pueden obtener los tiempos de procesamiento ajustado que se muestran en la Tabla N°4.

La forma como se resuelve un problema usando LACH, se muestra en los siguientes pasos:

Paso 0. VM={1, 2}, M= VM, y J={1, 2, 3, 4 ,5 ,6, 7}, L= 0, s= 2, WL^1= 0 y WL^2= 0.

Paso 1. Para la máquina 1: MC^1={21, 28, 23, 18, 32, 34, 17}, y MF^1={19, 24, 17, 23, 22, 19, 21}. Para la máquina 2: MC^2={38, 14, 38, 40, 35, 13, 40}, y MF^2={14, 14, 13, 21, 18, 23, 16}.

Paso 2. Para k^*= 2, i^*= 6 y j^*= 2, $\alpha^2_{6,2} = ap^2_{62} + mc^2_6 + mf^2_2 = 23+13+14 = 50$ es el mínimo.
Entonces L= 2, WL^2= 23, X^2= 6, Y^2= 2, J={1, 3, 4, 5, 7} y M={1}. Asignar la secuencia (6, 2) a la máquina 2. Eliminar fila 6, columna 2 y elemento (2, 6) de [AP^2]. Eliminar filas y columnas 6 y 2 de [AP^1].

Paso 3. Como $M \neq \emptyset$, ir al Paso 1.

Paso 1. Como M={1} y J={1, 3, 4, 5, 7};
MC^1={21, --, 23, 18, 32, --, 17} y MF^1={19, --, 17, 23, 22, --, 21}.

Paso 2. Para k^*=1, i^*=7 y j^*=3, $\alpha^1_{7,3} = ap^1_{73} + mc^1_7 + mf^1_3 = 23+17+17 = 57$ es el mínimo.
Entonces L= 2+2=4, WL^1= 23, X^1= 7, Y^1= 3, J={1, 4, 5} y M=$\emptyset$. Asignar la secuencia (7, 3) a la máquina 1. Eliminar fila 7, columna 3 y elemento (3, 7) de [AP^1]. Eliminar filas y columnas 7 y 3 de [AP^2]. Hasta aquí, las matrices [AP^k] estarían en la forma que se muestra en la Tabla Nº5.

Paso 3. Como M=$\emptyset$, ir al Paso 4.

Paso 4. T=J={1, 4, 5}, M={1, 2} y b= 2.

Paso 5. Se encuentra que $\lambda^1 = WL^1 + ap^1_{0,X^1} = WL^1 + ap^1_{07} = 23+27 = 50$ es el máximo.
Entonces, k^*= 1.
Dado que k^*= 1, X^1= 7, y Y^1= 3. Para W^1= 4 y Z^1= 1, se cumple que:
$P^1_1 = ap^1_{0,X^1} + ap^1_{Y^1,W^1} = ap^1_{07} + ap^1_{34} = 27+18 = 45$ y
$P^1_2 = ap^1_{0Z^1} + ap^1_{Z^1,X^1} = ap^1_{01} + ap^1_{17} = 24+21 = 45$ son mínimos respectivamente.
Let $M = M - \{1\} = \{2\}$.
Como $P_1{}^1 = P_2{}^1$, se elige arbitrariamente $P_2{}^1$, entonces: $T = T - \{1\} = \{4, 5\}$.

Paso 6. b= 2-1=1. Como b= 1, ir al Paso 7.

Paso 7. L= 4+1=5.
Dado que la máquina remanente es h= 2, X^2= 6 y Y^2= 2. Para W^2=5 y Z^2=5, se cumple que:
$P_1^2 = ap_{06}^2 + ap_{25}^2 = 23+39 = 62$ y
$P_2^2 = ap_{05}^2 + ap_{56}^2 = 37+19 = 56$ son mínimos respectivamente.
Como $P_2^2 < P_1^2$, asignar el trabajo 5 antes del trabajo 6 en la máquina 2.
$WL^2 = WL^2 + ap_{56}^2 = 23+19 = 42$. Eliminar fila 5, columna 6 y elemento (2,5) de $[AP^2]$, y fila y columna 5 de $[AP^1]$. X^2= 5 y J={1,4}.

Paso 8. Como se cumple que $L \le n - m$ (5≤ 5), ir al Paso 4.

Paso 4. $T = J$ = {1, 4}, $M = VM$ = {1, 2} y b= s= 2.

Paso 5. Se encuentra que $\lambda^2 = WL^2 + ap_{05}^2 = 42+37 = 79$ es el máximo. Entonces, k^*=2.
Dado que k^*= 2, X^2= 5, y Y^2= 2. Para W^2= 4 y Z^2= 4, se cumple que:
$P_1^2 = ap_{05}^2 + ap_{24}^2 = 37+45 = 82$ y
$P_2^2 = ap_{04}^2 + ap_{45}^2 = 40+35 = 75$ son mínimos respectivamente.
Hacer $M = M - \{2\} = \{1\}$.
Como $P_2^2 < P_1^2$, entonces $T = T - \{4\} = \{5\}$.

Paso 6. b= 2-1=1. Como b= 1, ir al Paso 7.

Paso 7. L= 5+1=6.
Dado que la máquina remanente es h= 1, X^1= 7, y Y^1= 3. Para W^1= 1 y Z^1= 1. se cumple que:

$P_1^1 = ap_{07}^1 + ap_{31}^1 = 27+21 = 48$ y
$P_2^1 = ap_{01}^1 + ap_{17}^1 = 24+21 = 45$ son mínimos respectivamente.
Como $P_2^1 < P_1^1$, asignar el trabajo 1 antes del trabajo 7 en la máquina 1.
$WL^1 = WL^1 + ap_{17}^1 = 23+21 = 44$. Eliminar fila 1, columna 7 y elemento (3,1) de $[AP^1]$, y fila y columna 1 de $[AP^2]$. X^1= 1 y $J = J - \{1\} = \{4\}$.

Paso 8. Como $L > (n - m)$=5 y $L < n$ (6 > 5 y 6 < 7), ir al Paso 9.

Paso 9. Como VM={1, 2} clacular β^1 y β^2.
$\beta^{k*} = \max_{k \in VM}\{ap_{0X^k}^k + WL^k\} = \max_{k \in VM}\{24+44;\ 37+42\} = \max_{k \in VM}\{68;79\} = 79$, entonces k^*=2.
Hacer $VM = VM - \{2\} = \{1\}$ y s=2–1=1. Ir al Paso 4.

Paso 4. $T = J$ = {4}, $M = VM$ = {1} y b= s= 1. Como b= 1, ir al Paso 7.

Paso 7. L= 6+1=7.
Dado que la máquina remanente es h= 1, X^1= 1, y Y^1= 3. Para W^1= 4 y Z^1= 4, se cumple que:
$P_1^1 = ap_{01}^1 + ap_{34}^1 = 24+18 = 42$ y
$P_2^1 = ap_{04}^1 + ap_{41}^1 = 25+23 = 48$ son mínimos respectivamente.
Como $P_1^1 < P_2^1$, asignar el trabajo 4 después del trabajo 3 en la máquina 1.
$WL^1 = WL^1 + ap_{34}^1 = 44+18 = 62$. Eliminar fila 3, columna 4 y elemento (4,1) de $[AP^1]$, y fila y columna 4 de $[AP^2]$. Y^1= 4 y $J = J - \{4\} = \{\emptyset\}$.

Paso 8. Como $L = n$ (7 = 7), ir al Paso 10.

Paso 10. $WL^1 = WL^1 + ap_{01}^1 = 62 + 24 = 86$; y $WL^2 = WL^2 + ap_{05}^2 = 42 + 37 = 79$.

Por lo tanto $C_{\max} = \max_{k=1,\ldots,m} \{WL^k\} = max\{86;79\} = 86$, y los trabajos son asignados a cada máquina según las siguientes secuencias, para la máquina 1: 0-1-7-3-4 y para la máquina 2: 0-5-6-2.

5. RESULTADOS

La heurística constructiva LACH fue comparada con una heurística constructiva glotona (greedy) llamada SAP-SL (Rabadi *et al.,* 2006) utilizando la librería de problemas propuesta por Rabadi (2005). Los problemas propuestos consideran combinaciones de 20, 40, 60, 80, 100, 120 trabajos, y *m* = 2, 4, 6, 8, 10, 12 máquinas. Estos problemas de prueba consideran tres tipos de escenarios y fueron generados según se muestra en la Tabla Nº6. La librería contiene 15 problemas para cada combinación de cantidad de trabajos, máquinas y para cada escenario. En total 1620 problemas.

Los resultados preliminares de LACH fueron presentados por Muñoz *et al.* (2005). En la Tabla Nº7 se muestra un resumen de los resultados obtenidos agrupados por escenario según Rabadi (2005). El análisis excluye los problemas para combinaciones de *m* = 8, 10, 12; y *n*=20 debido a que no cumplen la condición $n \geq 3m$ exigida por LACH (se excluyeron un total de 135 problemas debido a esta razón).

Las desviaciones (δ) fueron calculadas según la expresión propuesta por Rabadi *et al.* (2006), de tal manera que desviaciones negativas indican que SAP-SL reporta mejores resultados. En caso contrario LACH reporta mejores resultados.

$$\delta = \left(\frac{Existente - Propuesto}{Existente} \right) \times 100\% \qquad (14)$$

Donde:
Existente: resultado generado por la aplicación de SAP-SL.
Propuesto: resultado generado por la aplicación de LACH.

En el 61.89% de los problemas, LACH encontró mejores resultados que SAP-SL. Para los tres escenarios (Escenario 1, Escenario 2 y Escenario 3), las soluciones encontradas por LACH son mejores que las encontradas por SAP-SL en 0.58%, 0.60%, 0.52% respectivamente en promedio. En general, LACH entrega soluciones que son 0.57% mejores que SAP-SL en promedio. Se puede observar que las desviaciones (δ) son pequeñas, eso se debe exclusivamente a la forma en que se calcularon.

En la Figura Nº4 se muestra un histograma de las desviaciones obtenidas para todos los problemas del estudio, donde se puede observar que la mayor cantidad de observaciones tiene desviaciones ubicadas en la parte positiva del eje horizontal, lo que indica que LACH reporta mejores resultados para la mayoría de los problemas. También se observa que en los problemas donde LACH no logra encontrar mejores soluciones que SAP-SL las desviaciones son bajas (cercanas a cero).

6. CONCLUSIONES

Este articulo introduce una nueva heurística constructiva para minimizar el makespan en problemas de máquinas paralelas no relacionadas con tiempos de setup dependientes de la secuencia. Los resultados indican que el 61.89% de las instancias en estudio, la técnica propuesta arroja mejores resultados con respecto a otra heurística constructiva encontrada en la literatura. Esta nueva heurística puede ser usada como solución inicial para otros procedimientos de resolución como las meta-heurísticas o búsqueda local; esto constituye una línea de futuras investigaciones para los autores.

7. REFERENCIAS

1. Al-Salem, A. (2004). *Scheduling to minimize makespan on unrelated parallel machines with sequence dependent setup times*. Engineering Journal of the University of Qatar, Vol. 17, pp 177-188.

2. Anagnostopoulos, G. & Rabadi, G. (2002). *A simulated annealing algorithm for the unrelated parallel machine scheduling problem*. In: Proceedings of the World Automation Congress 2002, Orlando, FL, June 09-13.

3. Cheng, T. & Sin, C. (1990). *A state-of-the Art review of parallel-machine scheduling research*. European Journal of Operation Research, 47, 271-292.

4. Dhaenens-Flipo, C. (2001). *A bicriterion approach to deal with a constrained single-objective problem*. International Journal of Production Economics, Vol. 74, 93-101.

5. Franca, P., Gendreau, M., Laporte, G. & Müller, F. (1996). *A tabu search heuristic for the multiprocessor scheduling problem with sequence dependent setup times*. International Journal of Production Economics, Vol. 43, 79-89.

6. Graham, R.L., Lawler, E.L., Lenstra, J.K. & Rinnooy Kan, A.H.G. (1979). *Optimization and approximation in deterministic sequencing and scheduling: a survey*. Annals of discrete mathematics.

7. Graves, S.C. (1981). *A review of Production Scheduling*, Operation Research, 29, 646-675.

8. Guinet, A. (1993). *Scheduling sequence-dependent jobs on identical parallel machines to minimize completion time criteria*. International Journal of Production Research, Vol. 31, Nº7, 1579-1594.

9. Guinet, A. (1991). *Textile production systems: a succession of non-identical parallel processor shops.* Journal of the Operational Research Society, Vol. 42, Nº 8, 655-671.

10. Helal, M. & Hosni, Y. (2003). *A Tabu Search Approach for the Non-identical Parallel-Machines Scheduling Problem With Sequence-Dependent Setup Times*. In: Proceedings of the 2003 Industrial Engineering Research Conference, Portland, OR, May 18-21.

11. Herrmann, J., Proth, J. & Sauer, N. (1997). *Heuristics for unrelated machine scheduling with precedence constraints.* European Journal of Operational Research, 102, 528-537.

12. Kim, D., Kim, K., Jang, W. & Chen, F. (2002). *Unrelated parallel machine scheduling with setup times using simulated annealing*. Robotics and Computer Integrated Manufacturing, 18, 223-231.

13. Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G. & Shmoys, D.B. (1993). *Sequencing and Scheduling: Algorithms and complexity,* Handbooks in Operations Research and Management Science 4, Logistics of Production and Inventory, North Holland, Amsterdam, 445-524.

14. Lee, Y. & Pinedo, M. (1997). *Sheduling jobs on parallel machines with sequence-dependent setup times*. European Journal of Operational Research, 100, 464-474.

15. Lin C., Won, C. & Yeung, Y. (2002). *Heuristic approaches for a scheduling problem in the plastic molding department of an audio company*. Journal of heuristic, 8, 515-540.

16. Mokotoff, E. (2001). *Parallel machine scheduling problems: A survey*. Asia-Pacific Journal of Operational Research, 18, 193-242.

17. Moraga, R. (2004). *Meta-RaPS Approach for the Unrelated Parallel Machine Problem with Sequence-Dependant Setup Times,* In: Proceedings of the 2004 Industrial Engineering Research Conference, Houston, TX, May 15-19.

18. Muñoz, F., Moraga, R., & Baesler, F. (2005). *Look-Ahead Constructive Heuristic for the Unrelated Parallel Machine Problem with Sequence Dependent Setup Times.* In: Proceedings of the 2005 Industrial Engineering Research Conference, Atlanta, GA, May 14-18.

19. Pinedo, M. (2002). *Scheduling: Theory, Algorithms, and Systems*, Second Edition, Prentice-Hall, New Jersey, USA.

20. Rabadi, G., Moraga, R. & Al-Salem, A. (2006). *Heuristics for the Unrelated Parallel Machine Scheduling Problem with Setup Times*. Journal of Intelligent Manufacturing, 17, 85-97.

21. Rabadi, G. (2005). Librería de problemas de prueba par el problema de máquinas paralelas no-relacionadas con tiempos setup dependientes de la secuencia. Disponible en http://www.SchedulingResearch.com

22. Yu, L., Shih, H., Pfund, M., Carlyle, W. & Fowler, J. (2002*). Scheduling of unrelated parallel machines an application to PWB manufacturing*. IIE Transactions, 34, 921-931.

Tabla Nº1: Algoritmo de LACH

Paso 0.	VM= {1,…, m} y J= {1,…, n}. Inicialmente, M= VM, L= 0, s= m y WL^k= 0, para k= 1,..., m.
Paso 1.	Para cada $k \in M$, calcular vectores MC^k y MF^k sin considerar la fila cero. De tal forma que: $mc_j^k = \min_{i \in J}\{ap_{ij}^k\}$ para $j \in J$ y $mf_i^k = \min_{j \in J}\{ap_{ij}^k\}$ para $i \in J$.
Paso 2.	Hacer: $\alpha_{i^* j^*}^{k^*} = \min_{k \in M;\ i \in J;\ j \in J;\ i \neq j} \{ap_{ij}^k + mc_i^k + mf_j^k\}$, L= L+2, WL^{k^*}= $ap^{k^*}_{i^* j^*}$, X^{k^*}= i^*, Y^{k^*}= j^*, J= J – {i^*} – {j^*}, M= M – {k^*}. Asignar la secuencia (i^*, j^*) a la máquina k^*. Eliminar fila i^*, columna j^* y elemento (j^*, i^*) de $[AP^{k^*}]$. Eliminar filas y columnas i^* y j^* de $[AP^k]$ para todo $k \neq k^*$.
Paso 3.	Si M= $\emptyset$, ir al Paso 4. En otro caso, ir a Paso 1.
Paso 4.	Hacer T= J, M= VM y b= s. Si b= 1, ir al Paso 7. De lo contrario continuar.
Paso 5.	Encontrar $k^* \in M$ tal que: $\lambda^{k^*} = \max_{k \in M}(WL^k + ap_{0X^k}^k)$ Para máquina k^* encontrar los trabajos W^{k^*} y Z^{k^*} tal que $P_1^{k^*}$ y $P_2^{k^*}$ (costos de oportunidad) sean mínimos: $P_1^{k^*} = ap_{0X^{k^*}}^{k^*} + ap_{Y^{k^*}W^{k^*}}^{k^*}$ $W^{k^*} \in T$ y $P_2^{k^*} = ap_{0Z^{k^*}}^{k^*} + ap_{Z^{k^*}X^{k^*}}^{k^*}$ $Z^{k^*} \in T$ Hacer M = M–{k^*} Si $P_1^{k^*} < P_2^{k^*}$, hacer T= T – {W^{k^*}} En otro caso, hacer T= T – {Z^{k^*}}
Paso 6.	Hacer b= b–1 Si b= 1, ir al Paso 7. En otro caso, ir al Paso 5.
Paso 7.	Hacer L= L+1. Para la máquina remanente (h) en M, encontrar los trabajos W^h y Z^h tal que P_1^h y P_2^h sean mínimos: $P_1^h = ap_{0X^h}^h + ap_{Y^h W^h}^h$ $W^h \in T$ $P_2^h = ap_{0Z^h}^h + ap_{Z^h X^h}^h$ $Z^h \in T$ Si $P_1^h < P_2^h$ asignar el trabajo W^h después del trabajo Y^h en la máquina h. Hacer $WL^h = WL^h + ap_{Y^h W^h}^h$. Eliminar fila Y^h, columna W^h y elemento (W^h, X^h) de $[AP^h]$. Adicionalmente, eliminar fila y columna W^h de $[AP^h]$ para todo $k \neq h$. Hacer Y^h=W^h y J= J – {W^h}. En otro caso, asignar el trabajo Z^h antes de X^h en la máquina h. Hacer $WL^h = WL^h + ap_{Z^h X^h}^h$. Eliminar fila Z^h, columna X^h y elemento (Y^h, Z^h) de $[AP^h]$. Adicionalmente, eliminar fila y columna Z^h de $[AP^h]$ para todo $k \neq h$. Hacer X^h= Z^h y J= J – {Z^h}.
Paso 8.	Si $L \leq (n - m)$, ir al Paso 4. Si $L > (n - m)$ y $L < n$, ir al Paso 9. Si $L = n$, ir al Paso 10.
Paso 9.	Hacer $\beta^{k^*} = \max_{k \in VM}\{WL^{k^*} + ap_{0X^{k^*}}^{k^*}\}$. También VM= VM – {k^*} y s= s–1. Ir al Paso 4.
Paso 10.	Hacer $WL^k = WL^k + ap_{0X^k}^k$ para k = 1,...,m. Retornar $C_{\max} = \max_{k=1,\ldots,m}\{WL^k\}$

Tabla N°2: Tiempos de procesamiento.

Trabajo	1	2	3	4	5	6	7
Máquina 1	18	24	20	15	26	29	14
Máquina 2	35	6	35	34	31	7	37

Tabla N°3: Tiempos de setup dependientes de la secuencia.

Máquina 1

S_{ij}^1	1	2	3	4	5	6	7
0	6	17	5	10	10	13	13
1	--	11	9	4	6	16	7
2	9	--	8	16	12	17	10
3	3	4	--	3	13	12	3
4	5	11	5	--	7	13	14
5	4	4	10	13	--	13	14
6	6	14	17	14	13	--	5
7	11	4	3	6	10	5	--

Máquina 2

S_{ij}^2	1	2	3	4	5	6	7
0	9	8	8	6	6	16	16
1	--	8	7	17	9	7	13
2	12	--	12	11	8	15	12
3	15	10	--	12	16	6	11
4	8	16	6	--	4	14	14
5	5	12	5	17	--	12	14
6	9	17	7	6	11	--	3
7	3	11	3	9	11	9	--

Tabla N°4: Tiempos de procesamiento ajustado.

Máquina 1

AP^1	1	2	3	4	5	6	7
0	24	41	25	25	36	42	27
1	--	35	29	19	32	45	21
2	27	--	28	31	38	46	24
3	21	28	--	18	39	41	17
4	23	35	25	--	33	42	28
5	22	28	30	28	--	42	28
6	24	38	37	29	39	--	19
7	29	28	23	21	36	34	--

Máquina 2

AP^2	1	2	3	4	5	6	7
0	44	14	43	40	37	23	53
1	--	14	42	51	40	14	50
2	47	--	47	45	39	22	49
3	50	16	--	46	47	13	48
4	43	22	41	--	35	21	51
5	40	18	40	51	--	19	51
6	44	23	42	40	42	--	40
7	38	17	38	43	42	16	--

Tabla N°5: Tiempos de Procesamiento Ajustado después de asignaciones iniciales.

Máquina 1

AP^1	1	2	3	4	5	6	7
0	24	--	--	25	36	--	27
1	--	--	--	19	32	--	21
2	--	--	--	--	--	--	--
3	21	--	--	18	39	--	--
4	23	--	--	--	33	--	28
5	22	--	--	28	--	--	28
6	--	--	--	--	--	--	--
7	--	--	--	--	--	--	--

Máquina 2

AP^2	1	2	3	4	5	6	7
0	44	--	--	40	37	23	--
1	--	--	--	51	40	14	--
2	47	--	--	45	39	--	--
3	--	--	--	--	--	--	--
4	43	--	--	--	35	21	--
5	40	--	--	51	--	19	--
6	--	--	--	--	--	--	--
7	--	--	--	--	--	--	--

Tabla Nº6: Descripción de escenarios.

Escenario	Tiempo de proceso	Tiempo de Setup
1. *Balanceado*	~*Uniforme* [50,100]	~*Uniforme* [50,100]
2. *Proceso dominante*	~*Uniforme* [125,175]	~*Uniforme* [50,100])
3. *Setup dominante*	~*Uniforme* [50,100])	~*Uniforme* [125,175]

Tabla Nº7: Comparación de LACH v/s SAP-SL.

	Escenario 1	Escenario 2	Escenario 3	*Total*
Desviación Media (δ)	0,58%	0,60%	0,52%	0,57%
Desviación Mínima (min δ)	-9,90%	-5,43%	-4,82%	-9,90%
Desviación Máxima (max δ)	9,48%	8,30%	7,23%	9,48%
Mejores LACH	291	315	313	919
Proporción	*58,79%*	*63,64%*	*63,23%*	*61,89%*
Mejores SAP-SL	197	174	170	541
Proporción	*39,80%*	*35,15%*	*34,34%*	*36,43%*
Iguales	7	6	12	25
Proporción iguales	*1,41%*	*1,21%*	*2,42%*	*1,68%*

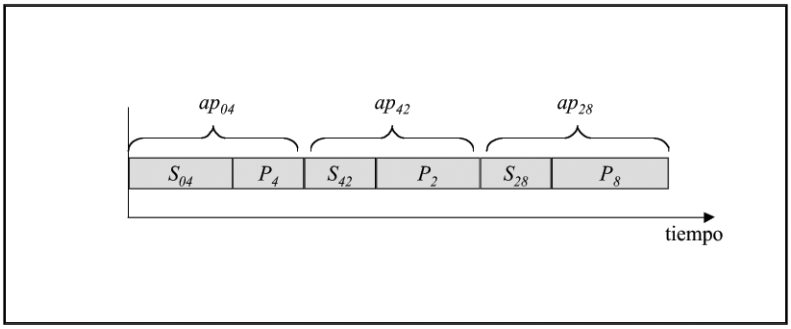

Figura N°1: Tiempos de procesamiento ajustado.

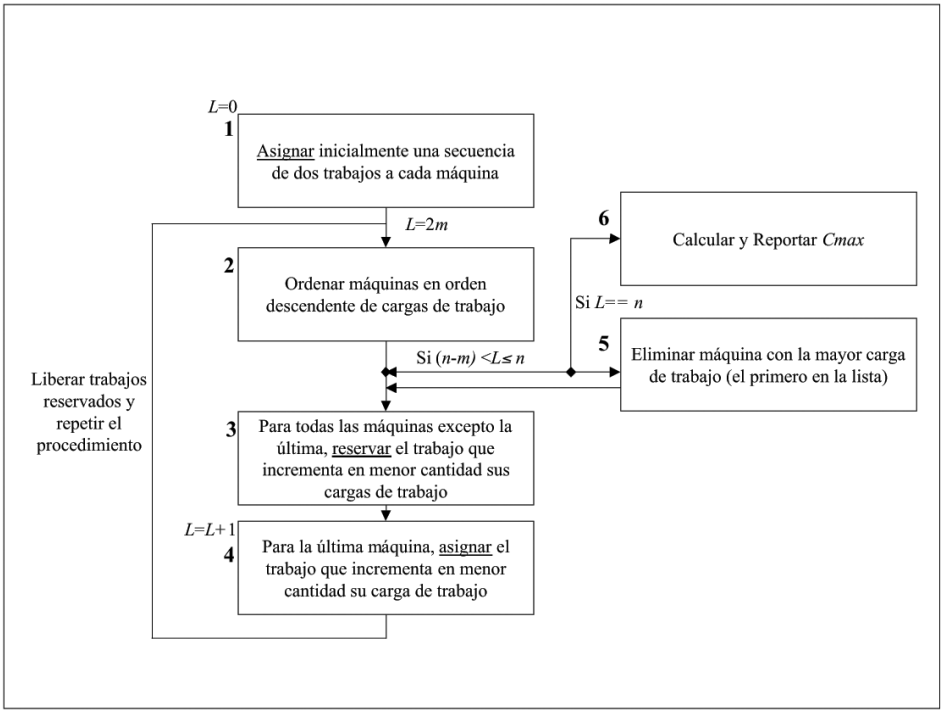

Figura N°2: Diagrama de LACH

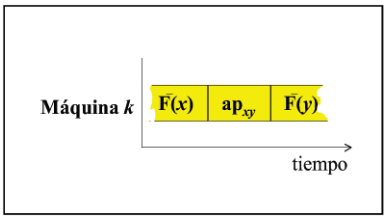

Figura N°3: Criterio de selección de trabajos para la Parte 1.

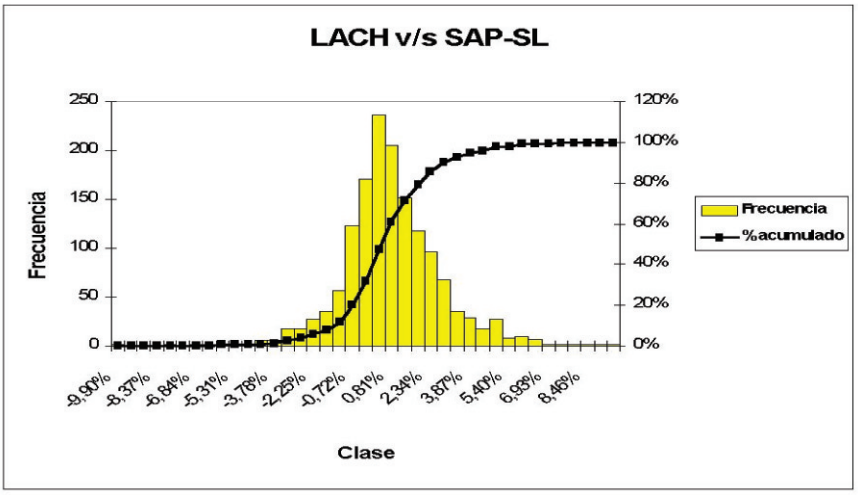

Figura Nº4: Histograma de desviaciones LACH v/s SAP-SL