ALGORITMO GENETICO MODIFICADO APLICADO AL PROBLEMA DE SECUENCIAMIENTO DE TAREAS EN SISTEMAS DE PRODUCCION LINEAL – FLOW SHOP

RESUMEN
En este trabajo se aborda el problema de producción de Flow Shop que consiste en el secuenciamiento de tareas con el objetivo de minimizar el tiempo total de producción aplicable a las industrias de proceso donde el uso eficiente de los recursos es de vital importancia. El problema es resuelto utilizando una metaheurística denominada algoritmo genético modificado de Chu-Beasley. Para evaluar la confiabilidad de la metodología se usaron datos de prueba que fueron tomados de la literatura especializada obteniendo resultados de alta calidad.

PALABRAS CLAVES: Programación de producción, algoritmo genético, Chu-Beasley, metaheurísticas.

ABSTRACT
This paper presents the Flow shop problem that finds the sequence of tasks that diminishes the process production total time. The Chu-Beasley genetic algorithm is used to solve this problem. The methodology was proven with test cases of specialized Literature.

KEYWORDS: *Flow shop, genetic algorithm, Chu-Beasley, metaheuristics.*

ELIANA MIRLEDY TORO OCAMPO
Ingeniera Industrial, Ms.C
Profesora catedrática.
Facultad de Ingeniería Industrial
Universidad Tecnológica de Pereira
eliana@ohm.utp.edu.co

YOV STEVEN RESTREPO GRISALES
Ingeniero Electricista.
Profesor Catedrático
Programa de Ingeniería Eléctrica
steven@ohm.utp.edu.co

MAURICIO GRANADA ECHEVERRI
Ingeniero Electricista, Ms.C
Docente
Programa de Ingeniería Eléctrica
Universidad Tecnológica de Pereira
magra@utp.edu.co

1. INTRODUCCIÓN

La eficiencia es usualmente expresada como el porcentaje de tiempo en que los recursos son usados productivamente. La utilización improductiva suele ser el resultado de actividades tales como puesta en marcha, paro, reconfiguración de máquinas por cambio de productos, mantenimiento, tiempos muertos e interrupciones por roturas. Todas ellas excepto las interrupciones por roturas pueden ser anticipadas, y su impacto en la eficiencia minimizadas con un buen planeamiento. Las roturas, obviamente, no están programadas y sus consecuencias tienen un doble impacto. Primero, se pierde producción mientras el recurso no esté disponible y segundo, en el caso de una interrupción prolongada, una nueva programación debe ser generada inmediatamente para minimizar las consecuencias de la interrupción.

El proceso del planeamiento y programación de la producción se describe usualmente usando un modelo de tres niveles, como se muestra en la figura 1. A estos tres niveles se los denomina frecuentemente: Planeamiento Estratégico, Planeamiento Táctico y programación de Producción (Planeamiento Operativo).

Los problemas de Planeamiento Estratégico se resuelven con programación lineal, los problemas de Planeamiento táctico se resuelven con métodos exactos como el Branch and Bound y Balas, entre otros. Los problemas de Planeación de Producción se resuelven por medio de técnicas heurísticas debido a la explosión combinatorial del espacio de soluciones [3].

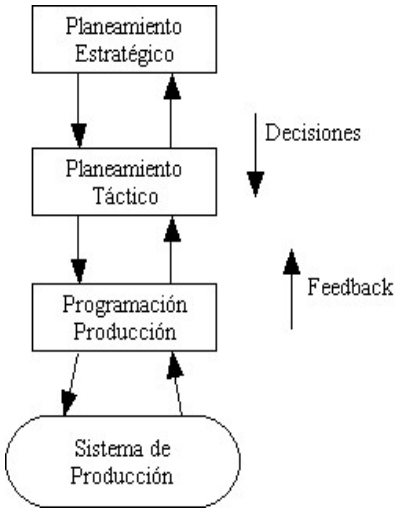

Figura 1. Proceso de planeamiento

En realidad el proceso de planeamiento esta dividido en varios componentes funcionales, con múltiples planes de soporte. El modelo típico de planificación se basa en pronósticos, plan maestro de producción, plan de

Fecha de Recepción: 30 Enero de 2006
Fecha de Aceptación: 13 Febrero de 2006

materiales, plan de producción y programación de operaciones como se detalla en la figura 2.

Un problema de programación de operaciones en un ambiente Flow Shop es un problema en el cual *n* tareas deben ser procesadas por un conjunto de *m* máquinas distintas y las tareas deben tener el mismo flujo de procesamiento o secuencia tecnológica en las máquinas.

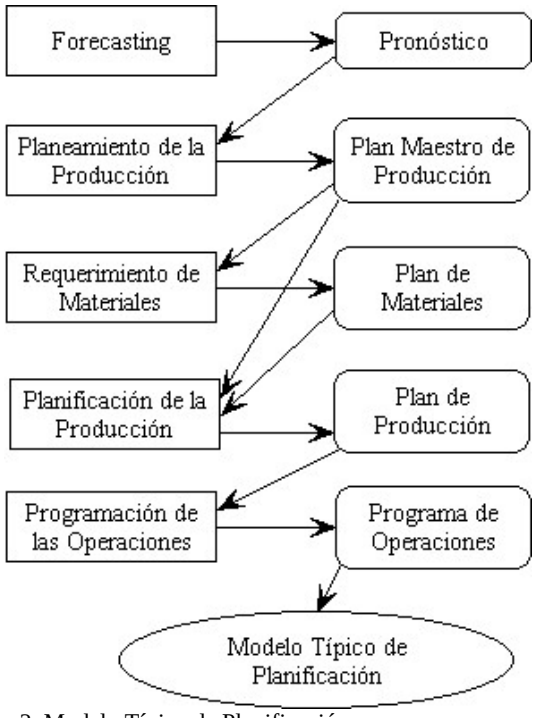

Figura 2. Modelo Típico de Planificación

2. PLANTEAMIENTO DEL PROBLEMA

La solución del problema consiste en determinar una secuencia de tareas entre las *n!* secuencias posibles pasando por todas las máquinas (Programación Permutacional) procurando optimizar una determinada medida de desempeño. En este trabajo se medirá la duración total de programación, esto se refiere al intervalo de tiempo transcurrido desde la ejecución de la primera tarea en la primera máquina hasta la ejecución de la última tarea en la última máquina [1][2] .

Las consideraciones usuales de un problema de programación de tareas Flow Shop son:

1) Cada máquina esta disponible continuamente y sin interrupciones.
2) Cada máquina puede procesar una tarea por vez.
3) Cada tarea sólo puede ser procesada por una máquina cada vez.
4) Los tiempos de procesamiento de las tareas en las diferentes máquinas son determinados y fijos.
5) Las tareas tienen la misma opción de ser programadas.
6) Los tiempos de preparación de las operaciones en las distintas máquinas están incluidos en los tiempos de procesamiento.
7) Las operaciones en las máquinas, una vez iniciadas no deben ser interrumpidas.

Según la teoría que estudia la complejidad matemática este problema es clasificado como NP- completo [4].

Un ejemplo del problema de secuenciación en línea es el que se muestra en la figura 3, donde en un ambiente de línea de flujo, se tiene un conjunto de 3 tareas que deben ser procesadas en un conjunto de 4 máquinas, cada tarea tiene el mismo orden de secuencia tecnológica a través de las máquinas, es decir, cada una de las tareas debe ser procesada primero en la máquina 1, luego en la máquina 2, y así sucesivamente hasta llegar a la máquina 4. Las secuencias se denominan técnicamente secuencias de permutación. El tiempo de procesamiento de la tarea *j* en la máquina *i* se denota por p_{ij} y el tiempo total de procesamiento o de ejecución se denomina $C_{\max}$ (*Makespan*).

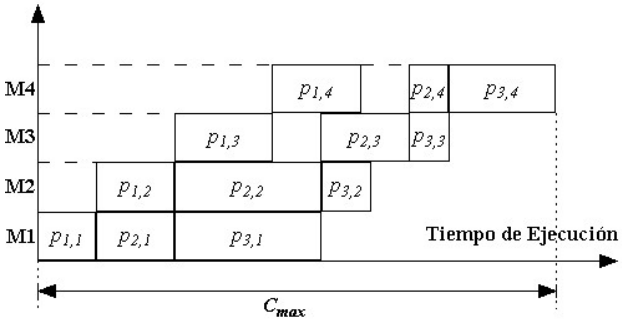

Figura 3. Secuenciación en línea de tres tareas en cuatro máquinas.

3. MODELAMIENTO MATEMÁTICO

El objetivo es encontrar la secuencia de los trabajos a programar la cual debe minimizar el *Makespan* que corresponde al tiempo en el cual el último trabajo es finalizado en la máquina m. El problema investigado usualmente es denotado por *n/m/P/* $C_{\max}$ y definido de la siguiente forma:

Si se tienen los tiempos de procesamiento *p(i,j)* para el trabajo *i* en la máquina *j* y la secuencia de trabajo $\{J_1, J_2, ..., J_n\}$ donde se calcula el tiempo completo de procesamiento o *Makespan* $C(J_i, j)$, entonces el modelo puede ser planteado así [7]:

$$C(J_1,1) = p(J_1,1) \quad (1)$$

Con la ecuación (1), se contabiliza el tiempo que tarda en ejecutarse el trabajo 1 de la secuencia de trabajo

$\{J_1, J_2, ..., J_n\}$ en la máquina 1. Partiendo de este valor y utilizando la ecuación (2), se contabilizan los tiempos de ejecución de la tarea J_i $(i \neq 1)$ en la máquina 1, adicionándole el tiempo acumulado de la secuencia en esta máquina en el momento del inicio de la ejecución del trabajo J_i.

$$C(J_i,1) = C(J_{i-1},1) + p(J_i,1) \quad for\ i = 2,...,n \tag{2}$$

La ecuación (3) es utilizada para contabilizar el tiempo de ejecución del trabajo J_1 en las demás máquinas del sistema de producción, teniendo en cuenta que al tiempo de ejecución de esta tarea en la máquina *j* se le debe sumar el tiempo que tardó en ejecutarse esta misma tarea en la máquina anterior (máquina *j-1*).

$$C(J_1, j) = C(J_1, j-1) + p(J_1, j) \quad desde\ j = 2,...,m \tag{3}$$

El tiempo acumulado de la secuencia de producción hasta que finaliza la ejecución de la tarea J_i $(i \neq 1)$ en la máquina j $(j \neq 1)$ se obtiene mediante la ecuación (4):

$$C(J_i, j) = \max\left\{C(J_{i-1}, j), C(J_i, j-1)\right\} + p(J_i, j) \quad desde\ i = 2,...,n \quad desde\ j = 2,...,m \tag{4}$$

Finalmente el tiempo total que tarda la secuencia de producción, contabilizado desde que ingresa el trabajo J_1 en la máquina 1 hasta que el trabajo J_n sale de la máquina *m* se expresa por medio de la siguiente ecuación:

$$C_{\max} = C\left(J_{n,m}\right) \tag{5}$$

4. METODOLOGIA DE SOLUCIÓN

El algoritmo genético propuesto por Chu-Beasley [8] en 1997 a ganado mayor importancia en el sector académico en los últimos años debido a la calidad en las respuestas y en el alto desempeño computacional obtenido al adaptarse a nuevos problemas. La principal característica del Algoritmo Genético de Chu-Beasley consiste en mantener constante el tamaño de la población de alternativas de solución, de manera que en cada iteración se reemplaza una alternativa de la población usando un eficiente mecanismo de modificación de la misma. En cada iteración la población es reemplazada sistemáticamente por un único descendiente generado. Esta estrategia tiene la ventaja de permitir encontrar múltiples soluciones y además conservar la diversidad del conjunto de alternativas.

4.1 Representación del Cromosoma

La codificación utilizada para representar cada una de las alternativas de solución del problema de flow shop consiste en construir un vector de tamaño *m (que corresponde al número de trabajos a ejecutar).* Por lo tanto, la *k-ésima* posición del vector representa el trabajo que se hará en el *k-ésimo* lugar. La población de alternativas de solución se conforma por un número determinado de cromosomas como el mostrado en la figura 4. El cromosoma representa la secuencia natural en la que se programaran los trabajos requeridos.

1	2	4	6	3	7	5	8

Figura 4. Secuenciación de 6 tareas.

4.2 Selección

En el proceso de selección se definen el número de alternativas que serán escogidas de la población de manera aleatoria a fin de definir las alternativas padre, en este caso se utilizó la selección por torneo que consiste en escoger la mejor de las *k* alternativas elegidas en forma aleatoria, el proceso se repite dos veces para obtener los dos padres [5]. Según diagrama de flujo de la figura 5.

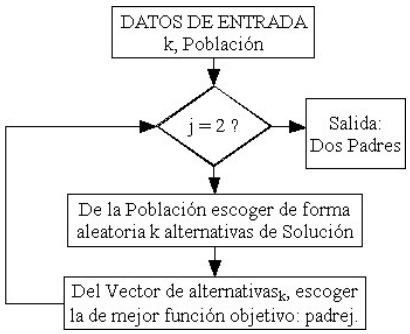

Figura 5. Diagrama de flujo de selección de padres

4.3 Recombinación

Con los dos padres seleccionados, el paso a seguir es recombinarlos de tal forma que se generen dos descendientes. En el proceso de recombinación es necesario definir el número *p* de puntos de recombinación; estos se eligen de forma aleatoria sobre el cromosoma y luego se realiza un cruzamiento de porciones del cromosoma como se muestra en la figura 6. Para conservar la legitimidad de las configuraciones obtenidas se aplicó una variación a este operador, llamada recombinación PMX (Partially Mapped Crossover) [6],de manera que se garantice la legitimidad de las configuraciones obtenidas y que consiste en lo siguiente:

1. Elegir aleatoriamente dos puntos de cruce.

2. Intercambiar estos 2 segmentos en los hijos que se generan.
3. El resto de las cadenas se obtienen haciendo mapeos entre los 2 padres:

    a. Si un valor no está contenido en el segmento intercambiado, permanece igual.

b. Si está contenido en el segmento intercambiado, entonces se sustituye por el valor que tenga dicho segmento en el otro padre.

Por ejemplo, si tenemos dos padres P1={1,2,4,6,3,7,5,8} y P2={5,4,1,7,2,6,8,3}
y el segmento seleccionado al azar de P_1 para ser insertado en P_2 es el {4,6,3} esto establece una relación con el segmento {1,7,2} que ocupa las mismas posiciones en P_2. Entonces la secuencia de operaciones transformarían P_2 en {5,4,4,6,3,6,8,3}, y luego, eliminado las repeticiones, quedaría {5,*,4,6,3,*,8,*} donde los asteriscos corresponden a los elementos repetidos que deben retirarse de la configuración. Reemplazando queda {5,1,4,6,3,*,8,*} ya que el 4 había ocupado el lugar del 1 en el segmento, continuando se obtiene {5,1,4,6,3,7,8,2}.

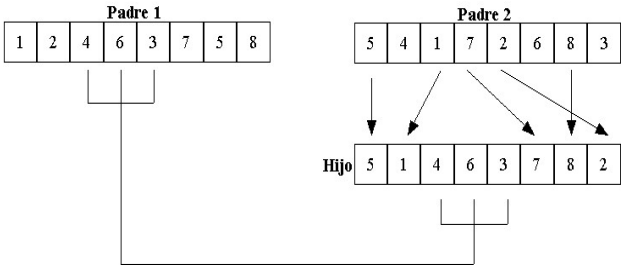

Figura 6. Ejemplo de aplicación del operador PMX

El resultado de la recombinación produce dos hijos de los cuales uno es eliminado de forma aleatoria. Otra estrategia es hacerlos competir por torneo o por ruleta.

4.4 Mutación

El algoritmo de mutación permite crear una amplia gama de propuestas, y su aceptación depende exclusivamente de si mejoran o no el valor de la función objetivo de la propuesta inicial. Este proceso esta ligado fuertemente al concepto de vecindad.[5].

En este trabajo, el cromosoma hijo obtenido en el proceso de recombinación es sometido al proceso de mutación. La mutación aplicada consiste en escoger, de manera aleatoria, dos trabajos e intercambiarlos en la secuencia de producción. En la figura 6, se muestra el algoritmo de mutación, donde el número de permutaciones o mutaciones j es una cantidad que el usuario puede escoger a voluntad. La cantidad recomendada de permutaciones oscila entre 10 y 20 para los problemas que son tratados en este artículo, una mayor cantidad puede hacer caer en redundancias al algoritmo, es decir que se llegue a un punto de saturación donde las permutaciones no aporten ninguna mejora a la secuencia evaluada.

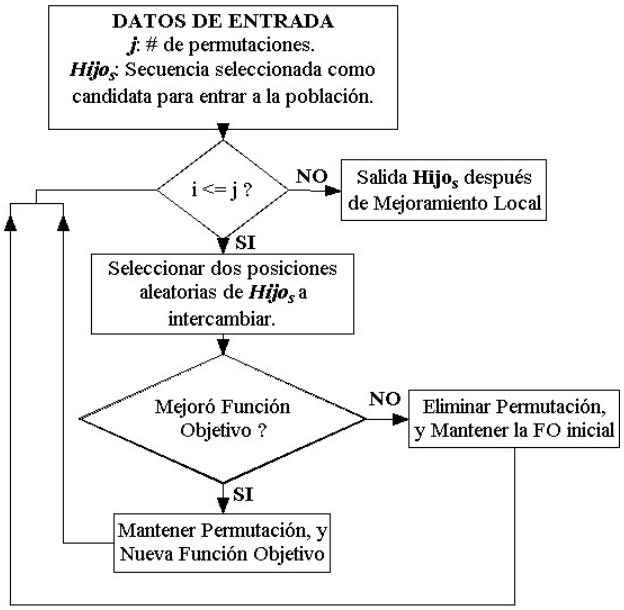

Figura 6. Proceso de Mutación.

4.5 Modificar la población actual

El algoritmo presentado en este trabajo tiene algunas características especiales asociadas a la población actual.

El algoritmo completo después de generada la población inicial consiste en repetir un número determinado de iteraciones los pasos descritos a continuación:

1. Se obtiene dos alternativas padre por selección de la población actual.

2. Se obtiene una alternativa hijo aplicando recombinación a los padres obtenidos en el paso anterior.

3. Se obtiene una alternativa modificada aplicando mutación.

4. Si la configuración es infactible se mejora la infactibilidad y se obtiene una alternativa menos infactible. De lo contrario ir al paso 5.

5. Se mejora la optimalidad de la alternativa en estudio.

6. Si la alternativa resultante de aplicar los pasos anteriores no se encuentra en la población, entonces aplicar estrategia de modificación de la población de lo contrario volver al paso 1.

Para modificar la población se propone la siguiente estrategia:

1. Si la alternativa actual es infactible y a su vez es menos infactible que la peor infactible de la población, entonces reemplazar la peor infactible por la alternativa actual.

2. Si la configuración es factible y existe por lo menos una infactible en la población actual, entonces reemplazar la peor infactible por la alternativa actual.

3. Si la configuración es factible y todas las alternativas de la población actual son factibles, entonces reemplazar la alternativa con peor función objetivo por la alternativa actual. Lo anterior se realiza sólo si la alternativa actual es de mejor calidad que la peor de la población.

La estrategia de modificación de la población actual se realiza cambiando sólo una alternativa por iteración y teniendo en cuenta que no se admiten alternativas repetidas. Lo anterior evita convergencias prematuras y asegura una exploración detallada de la región de soluciones. Adicionalmente se pueden obtener múltiples soluciones de un mismo problema.

Esta estrategia busca preservar las mejores alternativas, asegurando factibilidad y optimalidad. Estas características constituyen la principal diferencia con respecto al algoritmo propuesto por Chu-Beasley, en el cual la alternativa más infactible es reemplazada. A diferencia de los algoritmos genéticos tradicionales, no se modifica la población de forma aleatoria.

4.6 Los problemas

Muchos investigadores han probado las metodologías de solución para problemas de secuenciación de máquinas usando casos que se han generado aleatoriamente, sin embargo la gran mayoría de ellos no han sido publicados. Debido a esta situación aparece un conjunto de problemas de alta dificultad denominados problemas Taillard [9] que sirven como casos de prueba. El rango de los problemas comprende casos desde 20 trabajos y 5 máquinas hasta casos de 500 trabajos y 20 máquinas, donde los tiempos de procesamiento de cada tarea (incluyen tiempo de preparación de la máquina j para realizar la tarea i, más el tiempo de ejecución de dicha tarea en la presente máquina) fueron obtenidos aleatoriamente de una distribución uniforme U(1,100).

5. RESULTADOS

Para verificar la metodología propuesta, se desarrolló una aplicación en Delphi 7.0, en la cual se ejecutaron los casos Taillard que aparecen en la tabla 1, donde se puede distinguir para cada caso el número de tareas y la cantidad de máquinas disponibles respectivamente, además un intervalo dentro del cual debe estar el valor de la función objetivo de la secuencia óptima.

Cabe anotar que el límite inferior, es un valor que se obtiene al relajar el problema, es decir, se idealiza suponiendo que las máquinas inmediatamente terminan una tarea están listas para realizar la siguiente, lo que permite disminuir el tiempo de ejecución de una tarea en cada máquina, debido a que a los tiempos de procesamiento de una determinada tarea en una máquina cualquiera se le restan los tiempos de preparación de dicha máquina para realizar la tarea asignada.

Caso	# de Tareas	# de Máquinas	Cmax	
			Límite Inferior	Límite Superior
ta_20_5_01	20	5	1232	1278
ta_20_5_02	20	5	1290	1359
ta_20_20_01	20	20	1911	1297
ta_50_5_01	20	5	2712	2724

Tabla 1. Casos de Prueba.

Seguidamente en la tabla 2, se encuentran los resultados obtenidos con la aplicación desarrollada para cada uno de los casos de prueba. En esta tabla se puede observar que los resultados obtenidos fueron satisfactorios, permitiendo que para cada problema en el peor de los casos se alcanzara el valor máximo de la función objetivo exigido.

El buen desempeño del algoritmo depende de una buena calibración de los parámetros: tamaño de la población inicial, rango PMX, el valor de *k* para la elección de los descendientes (selección por torneo), número de permutaciones (mutación). El valor asignado para cada parámetro depende exclusivamente de la complejidad del problema.

CASO	Tamaño Población	Constante k para Selección	# de Permutaciones	Cmax
ta_20_5_01	100	3	20	1278
ta_20_5_02	100	3	15	1329
ta_20_20_01	500	3	20	1297
ta_50_5_01	500	3	10	2724

Tabla 2. Resultados de la Metodología Propuesta.

6. AGRADECIMIENTOS

Los autores expresan su agradecimiento a la Universidad Tecnológica de Pereira por su apoyo al grupo de desarrollo en Investigación de Operaciones - DINOP.

7. CONCLUSIONES

La metodología propuesta del algoritmo genético modificado de Chu-Beasley, muestra resultados de alta calidad y se adapta apropiadamente a los problemas de flow shop estudiados en este artículo.

Con la metodología implementada fue posible encontrar múltiples soluciones diferentes, característica propia del

algoritmo de Chu-Beasley que garantiza la diversidad de la población durante todo el proceso.

Este método se proyecta como una alternativa interesante en la solución de problemas de alta complejidad matemática, y con el cual se podrían resolver problemas de mayor dificultad que los estudiados en este artículo, empleando menor tiempo computacional y mejor calidad de respuesta comparado con otras metaheurísticas.

En las empresas dedicadas a la producción de bienes, el mayor capital esta concentrado en las materias primas y maquinarias, el uso eficiente de estos recursos mide el grado de rendimiento del negocio. La solución del problema del flow shop abre un panorama para empresas con producción en línea.

En el algoritmo planteado no fue necesario implementar una etapa de factibilización, debido a que la población inicial se genera con secuencias factibles y en el operador de recombinación se utiliza el método PMX que garantiza secuencias legítimas [6]. Sin embargo, como una recomendación para futuras investigaciones, es posible considerar alternativas de solución infactibles lo cual justificaría una etapa de factibilización y permitiría contar con un espacio de soluciones mayor.

Existen diversos objetivos que pueden ser evaluados al buscar una secuencia óptima, como por ejemplo: cumplir con las fechas de entrega, minimizar la tardanza de trabajos, minimizar horas extras, maximizar la utilización de máquinas, minimizar el tiempo ocioso, o minimizar inventario de trabajo en proceso. La selección del objetivo a optimizar depende de la necesidad del usuario, en este trabajo se minimizó el tiempo en el sistema (*Cmax*).

8. BIBLIOGRAFÍA

[1]. HILLER F. y LIEBERMAN G. J.. Introducción a la Investigación de Operaciones. 3ra Edición, Mc Graw Hill, México, 1982

[2]. SALAZAR María Angélica. RÍOS Roger. Minimización Heurística del número de tareas tardías al secuenciar Líneas de Flujo. Revista de la facultad de Ingeniería Mecánica y Eléctrica de la Universidad autónoma de Nuevo León México.Vol. VII, No 23, abril-junio 2004.

[3]. HOYOS, Mario. Memorias del Primer Encuentro Nacional de Automática. Corporación Universitaria Autónoma de Occidente. Noviembre 24-27 de 1993. Cali. Colombia.

[4]. MOCELLIN. Joao Vitor, BUZZO Walter Rogerio. Programacao da Producao em sistemas Flow-Shop utilizando um Método Heurístico Híbrido Algoritmo Genético- Simulated Annealing. Revista Gestao & producao. Brasil. VII. No 3, diciembre 2000.

[5]. TORO E., GRANADA M., ROMERO R. Algoritmo Memético aplicado a la solución del problema de Asignación Generalizada.Revista Tecnura. Año 8 No 16 semestre I -2005.

[6]. DIAZ Adenso.GLOVER Fred y otros. Optimizaciòn Heurística y Redes Neuronales. Editorial Paraninfo s.a. Magallanes España 1996.

[7]. REEVES Colin. A generis Algorithm for Flowshop Sequencing. Pergamon. Computers Ops Res. Vol 22 No 1, pp 5-13, 1995 Great Britain.

[8]. BEASLEY, J.E. CHU, P.C. A Genetic Algorithm for the Generalized Assignment Problem. Computers and Operations Research, 24(1), pp 17-23, 1997.

[9]. SCHEDULING INSTANCES. http://ina.eivd.ch/Collaborateurs/etd/problemes.dir/ordonnancement.dir/ordonnancement.html