Una Metodología para Resolver Problemas de Optimización Discreta

Martín G. Martínez-Rangel[1,2], Marco Antonio Cruz-Chávez[1], José Crispín Zavala-Díaz[2], Juan Frausto-Solis[1], David Juárez-Romero[1], Ocotlán Díaz-Parra[1], Alina Martínez-Oropeza[1]

[1]CIICAP-[2]FCAeI, Universidad Autónoma del Estado de Morelos
Avenida Universidad 1001. Col. Chamilpa, C.P. 62210. Cuernavaca, Morelos, México
{mmtzr,mcruz, djuarez, ocotlandp,crispin_zavala, alinam@uaem.mx}

Abstract. El problema de asignación de recursos en un taller de manufactura (JSSP) por su naturaleza misma de intratabilidad en el área de optimización discreta ha sido abordado a través de diversos enfoques. En este artículo, se aplica una metodología a este problema, la cual se utiliza en el modelado de optimización de redes dinámicas híbridas. Esta metodología aplicada a JSSP permite abordarlo de una manera más práctica mediante una secuencia de pasos a seguir para la búsqueda de una solución.

1 Introducción

Muchos de los problemas de optimización combinatoria son clasificados como NP-duros, siendo el problema de calendarizar el trabajo en un taller de manufactura (JSSP o Job Shop Scheduling Problem) uno de los problemas más difíciles de resolver en esta clasificación [2]. JSSP es importante tanto en el área industrial porque tiene que ver con la administración de los recursos, como también en el campo de la investigación que tiene que ver con optimización combinatoria [3][4]. Lo anterior da la razón el porque JSSP a sido abordado a través de diversos enfoques [5] a fin de poder en principio caracterizar el problema, y en segundo proponer un método de solución que lo resuelva. En este documento se aplica al problema de JSSP una metodología utilizada para el modelado de optimización de redes dinámicas híbridas, esta metodología es propuesta por Agarwal y Grossmann en [1], y demuestran que los problemas de optimización pueden ser modelados utilizando técnicas de programación matemática y una de las formas mas comunes de programación matemática para este tipo de problemas es mediante programación entera mixta (MIP), que es el lenguaje de modelación estándar [6]. La metodología aplicada consta de los siguientes pasos: una descripción conceptual del problema a abordar, formulación de un modelo formal, no ambiguo o modelo matemático que exprese el problema, propuesta de un mecanismo de solución, aplicación del mecanismo para resolver el problema planteado y encontrar la solución. En este documento se realiza un análisis del problema de JSSP siguiendo el enfoque de MIP, lo que permite obtener un modelo para representar el problema así como todas las variables involucradas en el mismo. En principio el modelo matemático para el problema de JSSP es simple, ya que solamente se requiere

M.A Cruz-Chávez, J.C. Zavala-Díaz (Eds): CICos 2008, ISBN:978-607-00-0165-9. pp. 11-22, 2008

tener una función objetivo, que es minimizar el tiempo total de procesamiento de todas las operaciones dadas (conocido como makespan o C_{max}), en este modelo sobresalen dos conjuntos de restricciones a considerar por su importancia, uno de los conjuntos tiene variables enteras binarias (de decisión) que son utilizadas para implementar las restricciones disyuntivas o restricciones de capacidad de recursos. El otro conjunto de restricciones tiene que ver con las restricciones de precedencia o de orden en la asignación de las operaciones siguiendo una secuencia lógica.

Este trabajo esta compuesto de las siguientes secciones que presentan los pasos a seguir dentro de la metodología a utilizar. La sección 1 introduce el tema. La sección 2 describe de manera conceptual el problema donde se identifican las variables del sistema, se especifican los tipos de datos utilizados (flotantes, enteras, lógicas), también se identifica el tipo de relaciones que se dan en el sistema, así como también se identifica la función objetivo y las restricciones que se tienen para resolver el problema, haciendo uso del modelo de grafos disyuntivo [8] que representa el problema de JSSP. La Sección 3 crea el modelo formal de manera completa (no ambigua) a partir de la descripción conceptual del sistema JSSP, el resultado que se logra es la expresión del problema en un lenguaje matemático, donde se involucran las variables definidas en la descripción conceptual. La Sección 4 describe el funcionamiento del algoritmo Colonia de Hormigas (o algoritmo sistema hormiga) aplicado al problema de JSSP. En la sección 5 se aplica el algoritmo Colonia de Hormigas, se describe el progreso en la refinación de la solución, el valor que va teniendo la función objetivo y el valor de las variables involucradas, así como la forma en que las restricciones de precedencia y de capacidad de recursos se van resolviendo y se muestran los resultados en forma gráfica y en forma tabular. En la Sección 6 se presentan las conclusiones.

2 Primer paso de la metodología: Descripción Conceptual de JSSP

En este paso de la metodología utilizada, se describe de manera conceptual el problema y se identifican las variables del sistema, especificándose sus tipos (flotantes, enteras, lógicas, etc), también se identifica el tipo de relaciones que se dan en el sistema, se identifica la función objetivo y las restricciones a las que se ve sujeto el problema y que se tienen que considerar a fin de resolverlo de manera correcta. Para el caso que nos ocupa, se describirá un problema simple de dos trabajos en dos máquinas, por lo que el problema clásico del JSSP puede establecerse como sigue de manera informal, se tienen dos trabajos que requieren ser procesados en dos máquinas distintas, cada uno de los trabajos tiene dos operaciones, y cada una de las operaciones tiene un tiempo determinado para su terminación. En la tabla 1 se presenta el orden tecnológico o secuencia lógica que deben seguir las operaciones para cada uno de los trabajos. Como se puede apreciar, a partir de la información que se presenta, existen operaciones de distintos trabajos que deben ser ejecutados en la misma máquina, pero además se debe respetar el orden dado. Por ejemplo, no se puede procesar la operación 2 si antes no se ha procesado la operación 1.

Tabla 1. Tabla de datos que describe un problema tipico de JSSP de 2x2

TRABAJOS	Descripción del Proceso {Máquinas} [Operación] (tiempo de proceso)	
1	{1} [1] (3)	{2} [2] (5)
2	{1} [3] (2)	{2} [4] (7)

De los datos presentados en la tabla 1, se identificaron las variables y su tipo que se encuentran involucradas en el problema de JSSP de dos trabajos y dos máquinas, estas variables son descritas en la tabla 2.

Tabla 2. Variables y su tipo para el JSSP de 2x2

Nombre	**Tipo**
Id_de_maquina	Entero
Id_trabajo	Entero
Id_de_operación	Entero
Eventos_de_operacion	Logica (true, false)
Lista_Operaciones_no_ejecutadas	Arreglo de enteros
Lista_Operaciones_a_ejecutar	Arreglo de enteros
Lista_tabu_Operaciones_ejecutada	Arreglo de enteros

La función objetivo que se tiene de manera informal es la siguiente: Encontrar la mejor secuencia de operaciones dentro de un conjunto de máquinas para la realización de un conjunto de trabajos, que minimice el tiempo máximo de procesamiento del último trabajo (makespan). Las restricciones encontradas son dos: a) de precedencia, b) de capacidad de recursos. Las restricciones de precedencia tienen que ver con la secuencia lógica de las operaciones a ser procesadas por trabajo (secuencia indicada por el número de operación), mientras que las restricciones de capacidad de recursos tienen que ver con la disponibilidad de las máquinas, considerando que una máquina solamente puede ejecutar un trabajo en un tiempo determinado. Las relaciones encontradas son las siguientes: Eventos de Operación {true, false} que corresponden al momento en que una máquina puede estar procesando una operación (true) o cuando una máquina no tiene mas operaciones a procesar o requiere de esperar (false) a que se libere un trabajo para iniciar su marcha (true) de nueva cuenta.

Definición del problema de manera gráfica

Como se puede apreciar en la figura 1, se tienen dos trabajos a ser procesados en dos máquinas diferentes, en esta representación se puede observar que cada uno de los trabajos tiene dos operaciones (indicadas por las flechas que parten de cada uno de los trabajo), y en donde cada operación requiere de un tiempo especifico de cada una de las máquinas que intervienen en el proceso.

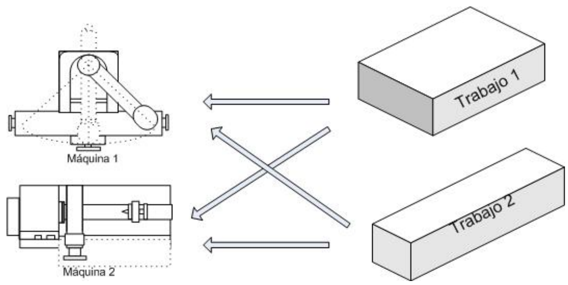

Fig. 1. Descripción gráfica del problema de JSSP de 2x2

Otra forma práctica de representar el problema para una instancia de dos trabajos en dos máquinas es mediante el modelo de grafos disyuntivo propuesto por Roy and Sussmann [8], como se puede apreciar en la figura 2, el trabajo 1 tiene las operaciones O_1, y O_2, el trabajo 2 tiene las operaciones O_3 y O_4.

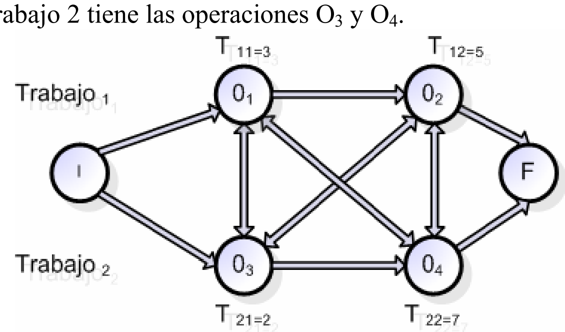

Fig. 2. Modelo de grafos disyuntivos paral problema de JSSP de 2x2.

Las máquinas pueden realizar una operación a la vez, la máquina 1 puede realizar las operaciones O_1 y O_3, y por otro lado las operaciones que puede realizar la máquina 2 son O_2 y O_4. Cada una de las operaciones requiere de un tiempo t de procesamiento. El problema que se tiene es que respetando las restricciones de precedencia (arcos conjuntivos) y las restricciones de capacidad (arcos disyuntivos) buscar la forma en que los trabajos se procesen en el menor tiempo posible, mediante una secuencia de las operaciones a realizar por las máquinas.

3 Segundo paso de la metodología: Representación del Problema Mediante un Modelo Formal

En este paso, se crea a partir de la descripción conceptual el modelo formal de manera completa (no ambigua) del problema en estudio en un lenguaje matemático, en este caso del sistema JSSP, donde se involucran las variables definidas en la descripción conceptual antes desarrollada.

Como ya ha sido reconocido por muchos investigadores, los problemas de optimización pueden ser modelados utilizando técnicas de programación matemática y una de las formas mas comunes de programación matemática para este tipo de problemas es mediante MIP, que como ya se dijo, es el lenguaje de modelación estándar, el cual es utilizado para describir de manera formal el problema de JSSP. La función objetivo que se plantea es minimizar el tiempo total de procesamiento de todas las operaciones dadas (Cmax), considerando las restricciones de precedencia y de capacidad de recursos, por lo que el modelo matemático que lo representa es el siguiente:

(1)

Min C_{max}

donde

Sujeto a:

$C_{jK}-t_{jk} \geq C_{jh}$ j=1,2,..,n h,k=1,2,...,m (2)

$C_{jK}-C_{ik} + M(1-x_{ijk}) \geq t_{jk}$ i,j=1,2,..,n k=1,2,...,m (3)

i=1,2,.,n k=1,2,.,m (4)

$C_{iK}, C_{jK}, \geq 0$ i=1,2,.,n h,k=1,2,.,m (5)

i=1,2,.,n h,k=1,2,.,m (6)

x_{ijk} =1 o 0

Donde la restricción de precedencia (2) asegura que la secuencia de procesamiento de las operaciones de cada Job corresponda a la orden determinada, y la restricción de capacidad de recursos (3) nos asegura que cada máquina procese un trabajo a la vez. Para las restricciones de precedencia, para una secuencia dada se denota C_{jk} como el tiempo de terminación del trabajo j en la máquina k y a t_{jk} como el tiempo de procesamiento del trabajo j en la máquina k, por lo que, para un trabajo j, si el procesamiento en una máquina h precede al de una máquina k se cumple (2).

Ahora se considerara la restricción de no-traslape de operaciones para una máquina dada. Para dos trabajos i y j, donde ambos necesitan ser procesados en la máquina k, si el trabajo i se procesa primero que el trabajo j necesitamos la restricción que viene dada en (7):

$$C_{jK} - C_{ik} \geq t_{jk} \quad (7)$$

De otra forma si j se procesa primero que i, entonces se requiere que se cumpla la restricción (8):

$$C_{iK} – C_{jk} \geq t_{ik} \quad (8)$$

Para el manejo de estas restricciones, es útil definir un coeficiente indicador como sigue:

X_{ijk}=1, si el trabajo i precede al trabajo j en la máquina k, o en su defecto

X_{ijk}=0, si el trabajo j precede al trabajo i en la máquina k.

Entonces se puede reescribir las restricciones anteriores como se señala en (3). Donde M es un número positivo grande. La desigualdad anterior representa claramente la restricción de no traslape. Si el el trabaja i se debe hacer primero, x_{ijk} es uno, lo que al restarse M(1- 1) es cero , quedando la desigualdad verdadera. De otra manera si el trabajo j es primero x_{ijk} es cero, lo que al restarse M(1-0) se hace un número muy grande haciendo la desigualdad verdadera.

4 Tercer paso de la metodología: Selección de un método de solución para JSSP

En este paso de la metodología, se describe el método que fue seleccionado para resolver el problema en estudio. Para el caso que nos ocupa, el método utilizado es mediante un algoritmo inteligente. El algoritmo utilizado para resolver el problema JSSP es el denominado Colonia de Hormigas (o algoritmo sistema hormiga), donde el estudio desde el punto de vista biológico de una colonia de hormigas presenta que su comportamiento es altamente estructurado y además se conoce que una hormiga tiene capacidades limitadas [7]. La metaheurística se compone de 3 pasos fundamentales

- Actividad de las hormigas.
- Evaporación de la feromona.
- Decisiones globales (opcionales).

El sistema de Hormigas simula el comportamiento de las hormigas reales, en los que iterativamente se añaden componentes de soluciones parciales tomando en cuenta entre otros elementos la información del problema a resolver, que consiste en identificar trazos de feromona artificial que cambia dinámicamente durante la ejecución del algoritmo para reflejar la experiencia adquirida por las hormigas en la búsqueda de cada solución. Donde el principio básico del algoritmo son los siguientes parámetros.

- Una población de L hormigas artificiales que construyen de manera cíclica soluciones para problemas de optimización combinatoria.
- El algoritmo impone una definición del problema de optimización en un grafo.
- Las hormigas para construir rutas que representan soluciones utilizan una regla de transición. Esta regla esta representada por la ecuación (9).

$$P_{ij} = \frac{\left[\tau_{ij}(t)\right]^{\alpha}\left[1/d_{ij}\right]^{\beta}}{\sum_{j \notin denodospermitidos}\left[\tau_{ij}(t)\right]^{\alpha}\left[1/d_{ij}\right]^{\beta}} \tag{9}$$

Donde

τ_{ij} - Es la cantidad de feromonas en el arco que conecta a un $nodo_i$ y a un $nodo_j$

d_{ij} - Distancia Heurística entre un $nodo_i$ y a un $nodo_j$

p_{ij} - Es la probabilidad de ramificarse desde un $nodo_i$ y a un $nodo_j$

Los parámetros α y β sintonizan la importancia relativa en probabilidad de el monto de feromonas versus la distancia heurística entre dos nodos.

Cuando todas las hormigas tienen construido una solución completa, esto es una secuencia de nodos visitados que resultan en una solución completa para el problema, el ciclo es completado y las feromonas actualizadas mediante una regla llamada Regla Global de Actualización (10).

$$\tau_{ij}\left(t+n\right) = (1-\rho).\tau_{ij}\left(t\right) + \rho.\Delta\tau_{ij}\left(t+n\right) \tag{10}$$

Donde:

$$\Delta\tau_{ij}(t+n)=\begin{cases} Q/f_{Evaluación}(\text{de\_lo\_mejor\_que\_hay}) \\ 0 \qquad \text{en otro caso}\end{cases}$$

ρ - Es el coeficiente de evaporación

Q - Es la cantidad de feromonas por unidad de distancia

5 Cuarto paso de la metodología: Aplicación del método Colonia de Hormigas para el JSSP

En esta parte se describe paso a paso la aplicación del método seleccionado, se describe el progreso en la refinación de la solución, el valor que va teniendo la función objetivo y el valor de las variables involucradas y se muestran los resultados en forma gráfica y en forma tabular, así como la forma en que las restricciones son tomadas en cuenta a fin de obtener una solución factible. El método Colonia de Hormigas es aplicado para resolver una instancia pequeña de JSSP de dos trabajos y dos máquinas. A continuación se muestra el progreso en la refinación de la solución aplicando el algoritmo de colonia de hormiga a la información ya mostrada en la tabla 1, que corresponde a los datos del problema ya presentado en el modelo conceptual.

La figura 3 muestra los posibles caminos que puede seguir una hormiga *K*, en el modelo de grafos presentado, los arcos conjuntivos corresponden a las restricciones de precedencia, mientras que los arcos disyuntivos corresponden a las restricciones de capacidad de recursos.

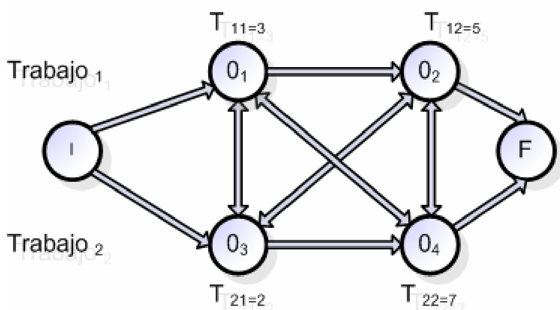

Fig. 3. Posibles caminos que puede seguir una hormiga *K*

La hormiga *K* comienza su recorrido desde su nido (Operación I) pasando por todos los estados (Operaciones) una sola vez. Una vez realizado este recorrido, la última operación a la que llegue la hormiga será la fuente donde este la comida. Para que la hormiga pueda realizar un recorrido correcto, debe guardar la siguiente información en los arreglos (listas):

G_k.-Una lista de los nodos que no ha visitado todavía;

S_k.- Una lista de los nodos que tiene permitido visitar partiendo de un nodo determinado, respetando las restricciones de precedencia (no se puede visitar O_2 si antes no se

ha visitado O_1) ecuación (2).
T.- Una lista tabú de los nodos ya visitados y que no puede visitar dos veces. El último nodo almacenado en T, es el nodo en que la hormiga se encuentra actualmente. Al final del recorrido la lista tabú tendrá la solución.
Cmax.- Guarda el valor de la función objetivo durante el proceso de construcción de la solución.

La sintonización de los parámetros α y β se logra mediante el incremento de β y manteniendo constante α por cada serie de pruebas. Los valores asignados a ambos parámetros son positivos y se recomienda asignarlos en incrementos de una unidad. Así mismo, el parámetro de la constante de evaporización ρ se fija por sintonización $0 < 0.001 < 1$, el número de hormigas que pueden realizar el trabajo se fija de acuerdo al número de nodos a ser visitados, en este caso L=n*m=4.

INICIO DE PROCESO

Iteración 1 (de la figura 3)
Nodo donde se encuentra la hormiga=I
Nodos aun no visitados Gk={O_1,O_2,O_3,O_4}
Nodos permitidos a visitar Sk={O_1,O_3}
Nodos tabú T={I}
Cmax=0

De acuerdo a la ecuación (9), se evalúa la probabilidad de pasar a un nodo permitido, para este problema, elegimos de manera aleatoria de la lista S_k el nodo O_1. El resultado se aprecia en la figura 4, los nodos ya visitados se muestran en gris.

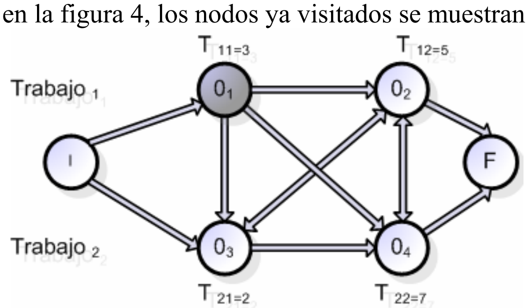

Fig. 4. Primer nodo visitado en la iteración 1

Haciendo que C_{jK} =3; t_{jk} =3; $C_{jh;}$=0 se cumple con la restricción de precedencia (2) $C_{jK}-t_{jk} \geq C_{jh;}$ por lo que 3-3≥0 para j=1;k=1 y h=2; Cmax=3. Por otra parte para la restricción de capacidad de recursos se cumple haciendo que C_{jK} =3, C_{iK} =0, t_{jk} =3 se cumple ya que 3-0≥0. El modelo de Gantt para la iteración 1 se aprecia en la figura 5.

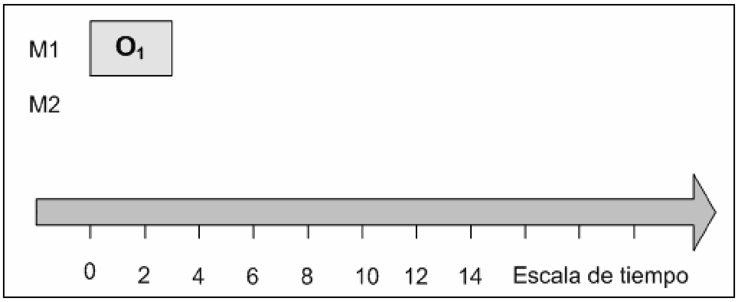

Fig. 5. Asignación de la primera operación

Iteración 2 (de la figura 4)
Nodo donde se encuentra la hormiga=O_1
Nodos aun no visitados $G_k=\{O_2,O_3,O_4\}$
Nodos permitidos a visitar $S_k=\{O_2,O_3\}$
Nodos tabú $T=\{I,O_1\}$
Cmax= 3

De acuerdo a la ecuación (9) se realiza la misma operación que en la iteración 1, de seleccionar de manera aleatorio el siguiente nodo a partir de la lista S_k de nodos permitidos respetando en todo momento la restricción de precedencia y de capacidad de Recursos. En este caso se elige el nodo O_3. El resultado se aprecia en la figura 6.

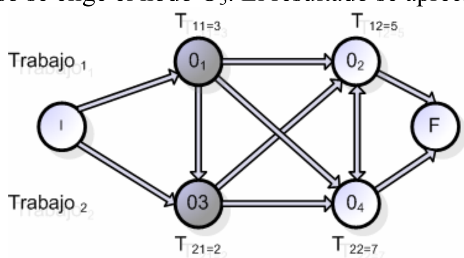

Fig. 6. Segundo nodo visitado en la iteración 2

Iteración 3 (de la figura 6)
Nodo donde se encuentra la hormiga=O_3
Nodos aun no visitados Gk=$\{O_2,O_4\}$
Nodos permitidos a visitar Sk=$\{O_2,O_4\}$
Nodos tabú $T=\{I,O_1,O_3\}$
Cmax=3

La O_3 es asignada en la máquina 2 al encontrarse esta libre, por lo que el valor de Cmax continua siendo igual a 3, como se puede apreciar en la figura 7.

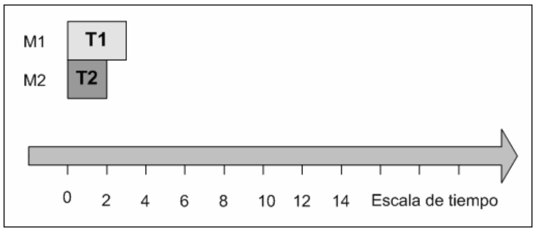

Fig. 7. Asignación de la segunda operación

De acuerdo a la ecuación (9) se realiza la misma operación que en las iteraciones pasadas, de seleccionar de manera aleatoria el siguiente nodo a partir de la lista S_k de nodos permitidos. En este caso se elige el nodo O_2. El resultado se aprecia en la figura 8.

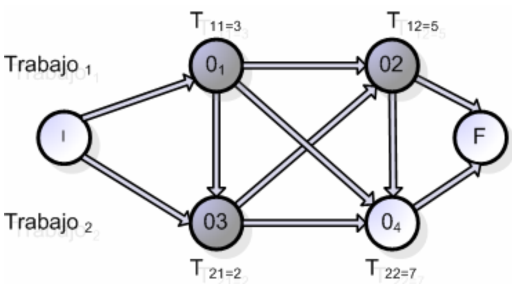

Fig.8 Tercer nodo visitado en la iteración 3

Iteración 4 (de la figura 6)

Nodo donde se encuentra la hormiga=O_2

Nodos aun no visitados G_k={O_4}

Nodos permitidos a visitar S_k={O_4}

Nodos tabú T={I,O_1,O_3,O_2}

Cmax=8

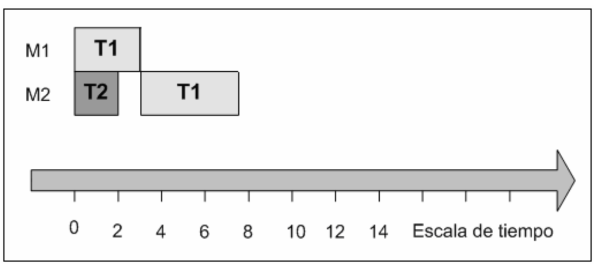

Fig.9 Asignación de la tercer operación

De acuerdo a la figura 9 y haciendo que C_{jK} =8; t_{jk} =5; $C_{jh;}$=3 se cumple con la restricción de precedencia (2) $C_{jK}-t_{jk} \geq C_{jh;}$ por lo que 8-5≥3 para j=1;k=1 y h=2; Cmax=8.

Por otra parte para la restricción de capacidad de recursos C_{jK} - C_{ik} $\geq$ t_{jk} se cumple haciendo que C_{jK} =8, C_{iK} =2, t_{jk} =5, ya que 8-2≥5. Dado que únicamente queda un nodo por visitar de la lista S_k, se elige el nodo O_4. El resultado se aprecia en la figura 10.

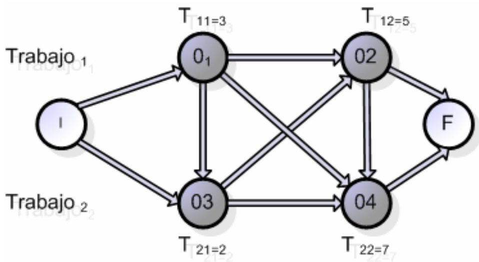

Fig.10 Cuarto nodo visitado en la iteración 4

Recorrido Final:
Nodo donde se encuentra la hormiga=O_4
Nodos aun no visitados G_k={Φ}
Nodos permitidos a visitar S_k={ Φ }
Nodos tabú T={I,O_1,O_3,O_2,O_4}
Cmax=10

De la lista tabú T, se obtiene la solución final. De acuerdo al orden de precedencia marcado en la lista tabú, se tiene que la secuencia de operaciones es:

En máquina 1: primero ejecutar la operación O_1 y después la operación O_4

En máquina 2: primero se ejecuta la operación O_3 y después la operación O_2.

El resultado se aprecia en la figura 10, donde ya existe una secuencia marcada por los arcos conjuntivos. De acuerdo a la solución encontrada por el sistema de hormiga, el resultado de la funciôn objetivo es=l0, el Makespan se puede apreciar en el diagrama de Gantt de la figura 11 (Schedule resultante). El Makespan lo determina la Operación O_4 del trabajo 2.

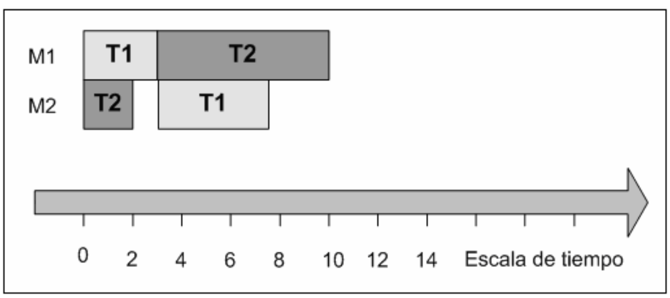

Fig. 11. Solución final del JSSP de 2x2

6 Conclusiones

Este trabajo se desarrollo siguiendo una metodología para el modelado de optimización de sistemas de redes híbridas. La aplicación de esta metodología muestra que los problemas de optimización pueden ser modelados utilizando técnicas de programación matemática como es el modelado de programación entera mixta. Los resultados obtenidos presentan que el enfoque planteado permite modelar sin ambigüedades el sistema que permite solucionar el problema de asignación de tareas en un taller de manufactura (JSSP) mediante la selección y aplicación de un método como es el de Colonia de hormigas.

Referencias

1. A. Agarwal and I. E. Grossman. Modeling for Optimization of Hybrid Dynamic Networks. Submitted to Computers Chemical Engineering, 2004.
2. Garey, M. R. Jonson, D.S and Shethi, R; "The complexity of Flor Shop and Job Shop Scheduling, in Mathematics Of Operation research, vol. 1; No.2 (1976) pag. 117-129.
3. M. A. Cruz-Chávez, M.G. Martinez-Rangel, J. A. Hernandez-Perez, J. C.-Zavala-Diaz, O. Díaz-Parra, An Algorithm of Scheduling for the Job Shop Scheduling Problem, Proceeding of CERMA2006, IEEE-Computer Society, ISBN, pp, 25-28 September, México, 2007.
4. Martín G. Martínez-Rangel, Marco Antonio Cruz-Chávez, J. Crispín Zavala-Díaz, David Juárez-Romero and Ocotlán Díaz-Parra, Analysis of the Simulated Annealing Convergence in Function of the Standard Deviation and the Boltzmann Quotient for Scheduling Problems, Research in Computing Science, IPN, ISSN 1870-4069, Vol., No., pp.,282-293 2007.
5. A. Jain S. Meeran.: A State of the Art Review of JOBSHOP Scheduling Techniques. Technical Report. Department of Applied Physics, Electronic and Mechanical Engineering University of Dundee, Dundee, Scotland, K,DD1 4HN,1998.
6. J. KallRath., Mixed integer optimization in the chemical process industry, experience, Potential and Future Perspectives .Edited by. ICE The Institution of Chemical Engineers. September 2000. Trans IChemE. Vol. 78. Part A.
7. M. Dorigo, V. Maniezzo, A. Colorni, Ant system: optimization by a colony of cooperating agents, IEEE Trans. Syst. Man Cybern. B 26 (1) (1996) 29–41.
8. Roy and Sussman, Les problemes d'ordonnancement avec contraintes disjonctives, Note D.S. no 9 bis, SEMA, Paris, France, December 1964.