PROGRAMACIÓN ÓPTIMA DE HORARIOS DE CLASE USANDO UN ALGORITMO MEMÉTICO

RESUMEN
El problema de programación óptima de horarios que se propone está conformado por un conjunto de eventos o clases que deben ser programados en 45 bloques de tiempo (5 días de 9 horas de clase cada uno), un conjunto de salones en los cuales se imparten las clases, un conjunto de estudiantes los cuales asisten a las clases y un conjunto de características satisfechas por los salones y requeridas por las clases. Cada estudiante asiste a un determinado número de clases y cada salón tiene un tamaño. El objetivo es elegir un salón y un bloque de tiempo para cada clase de forma que se maximicen las preferencias de los estudiantes sin crear conflictos en la programación de estudiantes o salones. La técnica empleada para resolver este problema se basa en la modificación y adecuación del algoritmo genético propuesto por Chu-Beasley.

PALABRAS CLAVES: optimización de horarios, Programación de cursos, Programación de salones.

ABSTRACT
The Timetabling problem proposed consists of a set of events to be scheduled in 45 timeslots (5 days of 9 hours each), a set of rooms in which events can take place, a set of students who attend the events, and a set of features satisfied by rooms and required by events. Each student attends a number of events and each room has a size. The objective is to choose meeting rooms and times for each event that maximize student preferences without creating student or room schedule conflicts. The used technique to solve this problem is based on the modification and adjustment of the genetic algorithm proposed by Chu-Beasley.

KEYWORDS: *Timetabling problem, Course scheduling, Classroom scheduling.*

MAURICIO GRANADA E.
Ingeniero Electricista, M.Sc.
Docente
Programa de Ingeniería eléctrica.
Universidad Tecnológica de Pereira
magra@utp.edu.co

ELIANA M. TORO OCAMPO
Ingeniera Industrial, M.Sc.
Docente Catedrático
Facultad de Ingeniería Industrial
Universidad Tecnológica de Pereira
eliana@utp.edu.co

JOHN F. FRANCO BAQUERO
Docente Catedrático
Programa de Ingeniería eléctrica.
Universidad Tecnológica de Pereira
jffb@utp.edu.co

1. INTRODUCCIÓN

El problema de programación óptima de horarios de clase que se aborda en este artículo corresponde a una versión simplificada, donde se consideran algunas restricciones de factibilidad y optimalidad de las muchas posibles así cómo un único recurso para ser asignado, los salones. La solución a este problema consiste en establecer una secuencia, en un período determinado de tiempo (típicamente una semana), de sesiones, clases o eventos ofrecidos a los estudiantes de forma que se cumpla un conjunto de restricciones de diferentes tipos. Es considerado un problema combinatorial multidimensional NP-completo de difícil solución.

En la literatura especializada es común encontrar soluciones que dividen el problema para posteriormente resolverlo de forma combinada e iterativa [2,3]. Generalmente, los subproblemas resultantes de esta división corresponden a uno de asignación de recurso único (programación de salones) y a otro de programación de horarios. Hertz [4] describe un algoritmo de búsqueda tabú para resolver los problemas de programación de horarios y asignación de estudiantes a grupos, en donde estos problemas se integran de forma iterativa para encontrar la asignación de profesor y salón. En [4] Carter documenta un método iterativo usado en la universidad de Waterloo para desarrollar programación de cursos.

Otras estrategias de programación de horarios se han basado en la aplicación de técnicas propias del campo de la inteligencia artificial. Mulvey [5] desarrolla un método de programación de asignación múltiple basado en el uso del modelo general de redes. Posteriormente, Dinkel [6] reportó una aplicación basada en el modelo de Mulvey donde se involucran 300 clases, 20 salones y 16 períodos de tiempo, donde la solución fue obtenida usando un aplicativo comercial de programación entera. A medida que el tamaño de los problemas crece, estas técnicas de solución exacta empiezan a presentar problemas de convergencia y tiempos computacionales prohibitivos, lo cual hace necesario el uso de métodos heurísticos de solución [1,7,8,9].

2. DESCRIPCIÓN DEL PROBLEMA

Para particularizar el problema a las condiciones establecidas en la Universidad Tecnológica de Pereira,

Fecha de Recepción: 30 Enero de 2006
Fecha de Aceptación: 7 Abril de 2006

éste se describe de la siguiente manera: se debe programar un conjunto de asignaturas en 45 bloques o períodos de tiempo, un conjunto de salones en los cuales se imparten las asignaturas, un conjunto de estudiantes que asisten a las clases de cada asignatura y un conjunto de características requeridas por las asignaturas y ofrecidas por los salones. Cada estudiante tiene programado un número determinado de asignaturas y cada salón posee una capacidad limitada de estudiantes. Un horario factible es aquel en el cual todas las asignaturas han sido asignadas a bloques de tiempo y a salones de clase, considerando que:

- Los estudiantes no deben recibir más de una asignatura al mismo tiempo (estudiantes con cruces de horario).
- La capacidad del salón debe ser suficiente para atender todos los estudiantes programados y debe satisfacer todos los requerimientos de la materia. Algunas materias requieren salones especiales como laboratorios y aquellas asistidas por computador, entre otras.
- Sólo una asignatura es impartida en cada salón en cualquier bloque de tiempo (cruce de salones).

Adicionalmente, se debe procurar que:

- Un estudiante no tenga clase en el último bloque del día.
- Un estudiante no reciba 2 eventos consecutivos.
- Un estudiante no reciba sólo una clase al día.

Las consideraciones anteriores constituyen dos diferentes tipos de restricciones denominadas *blandas y duras* y pueden ser ampliadas según las particularidades del problema que se quiera resolver. Por ejemplo, se puede considerar una restricción adicional que involucre la preferencia de horarios de una materia, la preferencia de un profesor por un salón y la distancia de desplazamiento de profesores y estudiantes, entre otras. Las restricciones duras corresponden a aquellas que deben cumplirse para que la alternativa de solución sea factible. Las restricciones blandas son aquellas que se desean minimizar (o maximizar) para buscar la optimalidad pero que no son de estricto cumplimiento.

La información involucrada en la construcción de un caso comprende los siguientes arreglos:

- Conjunto de n_S salones: $S = \{S_j : j \in J\}$, con $J = \{1, 2, \ldots n_S\}$.
- Conjunto de n_E eventos: $E = \{E_i : i \in I\}$, con $I = \{1, 2, \ldots, n_E\}$. Cada clase que se debe dictar es tratada como un evento. Un evento existe ó está activo cuando ha sido asignado al menos a un estudiante.
- Conjunto de n_{Es} estudiantes: $Es = \{Es_K : k \in K\}$, con $K = \{1, 2, \ldots n_{Es}\}$. Un estudiante existe ó está activo cuando tiene asignado al menos un evento.
- Conjunto de n_C características: $C = \{C_l : l \in L\}$, con $L = \{1, 2, \ldots, n_C\}$.
- Conjunto de n_T períodos o bloques de tiempo: $P = \{P_t : t \in T\}$, con $T = \{1, 2, \ldots, n_T = 45\}$.

Se define una matriz "*Evento_salón*" en la que se relaciona qué salones son propicios para un evento en particular, de la siguiente forma:

$$Evento\_Salón[E_i, S_j] = \begin{cases} 1 & \text{si el salón } S_j \text{ tiene las características requeridas por el evento } E_i \text{ incluyendo la capacidad necesaria para atender a todos los estudiantes asociados al evento } E_i \\ 0 & \text{en cualquiera de los otros casos} \end{cases}$$

Para que un salón haga parte de la solución del problema debe ser apto, al menos, para un evento. Esto quiere decir, que para un salón *s* se debe cumplir que:

$$\sum_{i=1}^{n_E} Evento\_Salon_{i,s} \neq 0 \qquad (1)$$

Se define una matriz "*Evento_Estudiante*" que relaciona a cada uno de los eventos con cada uno de los estudiantes. Es decir, relaciona las clases que el estudiante debe cursar en el semestre actual. Esta matriz es de la siguiente forma:

$$Evento\_Estudiante_{E_i, Es_k} = \begin{cases} 1 & \text{si el estudiante } Es_k \text{ tiene matriculado el evento } E_i \\ 0 & \text{en caso contrario} \end{cases}$$

Para que el estudiante *k* exista se debe cumplir que:

$$\sum_{i=1}^{n_E} Evento\_Estudiante_{i,k} \neq 0 \qquad (2)$$

Para que el evento *i* exista se debe cumplir que:

$$\sum_{k=1}^{n_{Es}} Evento\_Estudiante_{i,k} \neq 0 \qquad (3)$$

La figura 1 ilustra la codificación utilizada para representar una alternativa de solución del problema. Esta codificación consiste en construir n_E vectores fila de 45 posiciones cada uno, correspondientes a cada bloque de tiempo de la semana. Cada uno de estos vectores está asociado a un evento en particular y los valores existentes en cada posición corresponden a los salones asignados.

$$Alternativa=\begin{array}{c} \\ Evento_1 \\ Evento_2 \\ \vdots \\ Evento_{nE} \end{array}\begin{array}{c} \begin{array}{ccccc} Bloque\ 1 & Bloque\ 2 & Bloque\ 3 & \cdots & Bloque\ 45 \end{array} \\ \begin{array}{|ccccc|} \hline 3 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & 0 \\ 1 & 0 & 0 & \cdots & 0 \\ \hline \end{array} \end{array}$$

Figura 1. Codificación de la alternativa de solución.

En la figura 1, se puede observar que el evento 1 se llevará a cabo en el salón 3 en el bloque de tiempo 1. Las posiciones iguales a 0 indican que no se ha realizado ninguna asignación para un evento y bloque de tiempo determinado. Adicionalmente para que una alternativa sea factible, en este modelo en particular, sólo se asigna un salón por cada evento.

El modelo matemático que se propone, penaliza en la función objetivo todas las restricciones (duras y blandas). Por lo tanto, el problema puede ser presentado formalmente de la siguiente manera:

$$\text{minimizar} \quad \text{F.O.}$$

$$\underbrace{\left(w_1 \cdot \mathrm{Es}_C + w_2 \cdot \mathrm{S}_{\mathrm{RNS}} + w_3 \cdot \mathrm{S}_C\right)}_{\text{Restricciones duras}} + \underbrace{\left(w_4 \cdot \mathrm{Es}_{\mathrm{UB}} + w_5 \cdot \mathrm{Es}_{\mathrm{EC}} + w_6 \cdot \mathrm{Es}_{\mathrm{UE}}\right)}_{\text{Restricciones blandas}}$$

$$\begin{aligned} &\text{s.a.} \\ &Es_C = 0 \\ &S_{RNS} = 0 \\ &S_C = 0 \\ &\sum_{i=1}^{n_E} Evento\_Estudiante_{i,k} \neq 0 \\ &\sum_{k=1}^{n_{Es}} Evento\_Estudiante_{i,k} \neq 0 \end{aligned} \tag{4}$$

Donde:

Es_C es el número total de cruces de horario que presentan los estudiantes en sus materias (estudiantes que deben atender más de un evento al mismo tiempo).

$$Es_C = \sum_{t=1}^{45}\sum_{k=1}^{n_{Es}}\left(\left(\sum_{i=1}^{n_E} Num\_Cruces\_Est(t,k,i)\right)-1\right) \tag{5}$$

$$Num\_Cruces\_Est(t,k,i) = \begin{cases} 1 & \text{si } Evento\_Estudiante_{i,k} \times Alternativa_{i,t} \geq 1 \\ 0 & \text{En cualquier otro caso} \end{cases}$$

S_{RNS} es el número de asignaciones de eventos a salones que no cumplen con los requerimientos exigidos (requerimientos no satisfechos).

$$S_{RNS} = \sum_{t=1}^{45}\sum_{i=1}^{n_E} Num\_Salones(t,i) \tag{6}$$

$$Num\_Salones(t,i) = \begin{cases} 1 & \text{si } Evento\_Salon_{i,Alternativa_{i,t}} = 0 \\ 0 & \text{En cualquier otro caso} \end{cases}$$

S_C es el número de cruces de horarios entre salones, es decir aquellos que tienen más de un evento asignado al mismo tiempo.

$$S_C = \sum_{t=1}^{45}\sum_{j=1}^{n_S} Num\_Cruces\_Salones(t,j) \tag{7}$$

$$Num\_Cruces\_Salones(t,j) = \begin{cases} \sum_{i=1}^{n_E}\left(Contador(t,j,i)\right)-1 & \text{si } \sum_{i=1}^{n_E}\left(Contador(t,j,i)\right) \geq 1 \\ 0 & \text{en los otros casos} \end{cases}$$

$$Contador(t,j,i) = \begin{cases} 1 & \text{si } S_j = Alternativa_{i,t} \\ 0 & \text{En cualquier otro caso} \end{cases}$$

Es_{UB} es el número total de estudiantes que tienen programado un evento en el último bloque del día. Estos bloques pueden ser almacenados en un vector como el siguiente:

$$Ultimos\_bloques\_dia = \{9,18,27,36,45\}$$

En este vector también pueden ser almacenados algunos otros bloques en los que preferiblemente no se deban programar eventos.

$$Es_{UB} = \sum_{t=1}^{días=5}\sum_{k=1}^{n_{Es}}\sum_{i=1}^{n_E} Est\_Ultimo\_Bloque(t,k,i) \tag{8}$$

$$Est\_Ultimo\_Bloque(t,k,i) = \begin{cases} 1 & \text{si } Evento\_Estudiante_{i,k} \times \\ & \quad Alternativa_{i,Ultimos\_bloques\_dia_t} \geq 1 \\ 0 & \text{En cualquier otro caso} \end{cases}$$

Es_{EC} es el número total de estudiantes que reciben 2 eventos consecutivos.

$$Es_{EC} = = \sum_{t=1}^{44}\sum_{k=1}^{n_{Es}} \text{Eventos\_Consecutivos}(t,k) \tag{9}$$

$$\text{Eventos\_Consecutivos}(t,k) = \begin{cases} 1 & \text{si } \left(\sum_{i=1}^{n_E} Evento\_Estudiante_{i,k} \times Alternativa_{i,t} \geq 1\right) \text{ y} \\ & \quad \left(\sum_{i=1}^{n_E} Evento\_Estudiante_{i,k} \times Alternativa_{i,t+1} \geq 1\right) \\ 0 & \text{En cualquier otro caso} \end{cases}$$

Es_{UE} es el número total de estudiantes que tienen programado, en algún día de la semana, sólo un evento.

$$Es_{UE} = = \sum_{i=1}^{n_E}\sum_{k=1}^{n_{Es}} \text{Evento\_Unico\_Dia}(i,k) \tag{10}$$

$$\text{Evento\_Unico\_Dia}(i,k)=\begin{cases}1 & \text{si } \sum_{t=\text{dia\_inicio}}^{\text{dia\_fin}} \text{Suma\_Eventos}(i,k,t)=1\\ 0 & \text{en cualquier otro caso}\end{cases}$$

$$\text{Suma\_Eventos}(i,k,t)=\begin{cases}1 & \text{si } \textit{Evento\_Estudiante}_{i,k}\times\textit{Alternativa}_{i,t}\geq 1\\ 0 & \text{En cualquier otro caso}\end{cases}$$

donde dia_inicio={1,10,19,28,37} y dia_fin={9,18,27,36,45}

$w_1,\ w_2,\ w_3, w_4, w_5$ y w_6 son parámetros de ponderación encargados de dar mayor o menor importancia al cumplimiento de una restricción en particular. Por lo tanto, para garantizar el cumplimiento de las restricciones duras, los valores de $w_1,\ w_2$ y w_3 deben ser altos (≥ 1000) y los parámetros asociados a las restricciones blandas se pueden considerar alrededor de la unidad.

3. ALGORITMO MEMÉTICO

El algoritmo que se propone se basa en la modificación del algoritmo genético planteado por Chu-Beasley [1]. La principal característica de este algoritmo 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 mecanismo eficiente de modificación. Dicho mecanismo busca beneficiar las alternativas con menor índice de infactibilidad y de mejor calidad, de forma que en cada iteración la población es reemplazada sistemáticamente por un único descendiente generado. Esta estrategia tiene la ventaja de encontrar múltiples soluciones y mantiene alta diversidad en los individuos de la población. El resto del algoritmo obedece, en su estructura básica, a un algoritmo genético tradicional cuyas secciones se explican a continuación.

3.1 Cromosoma y población inicial

La codificación utilizada para representar cada una de las alternativas de solución o cromosoma se muestra en la figura 1. La población inicial de alternativas de solución se conforma por un número determinado de estos cromosomas y se puede construir a partir de técnicas de inicialización como metodologías constructivas y análisis de sensibilidades, entre otras, o realizando un proceso aleatorio de asignación de salones y bloques de tiempo. En este artículo se utiliza esta última opción para la construcción de la población inicial.

3.2 Proceso de selección

En el proceso de selección es necesario definir el número k de alternativas que serán escogidas de forma aleatoria de la población para que compitan a fin de seleccionar una alternativa padre. El proceso utilizado en este caso es la selección por ruleta y debe realizarse dos veces a fin de obtener dos padres [9].

3.3 Recombinación

Con los dos padres seleccionados, el siguiente paso consiste en combinarlos de forma tal que se obtenga solamente un descendiente. En el proceso de recombinación es necesario definir el número p de puntos de recombinación. Dichos puntos son escogidos de forma aleatoria sobre el cromosoma. Posteriormente, se combinan las características de los padres haciendo un cruzamiento de las porciones de cromosoma existentes entre cada punto de recombinación como se muestra en la figura 2 para un cromosoma de 4 bloques de tiempo, 5 eventos, 4 salones y 2 puntos de recombinación.

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

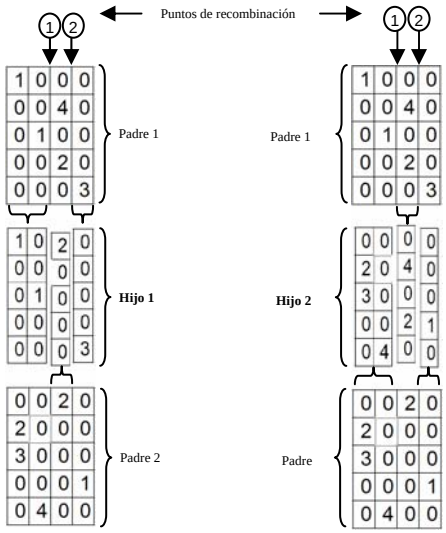

Figura 2. Recombinación de dos puntos.

Nótese que el hijo 1 mostrado en la figura anterior presenta una asignación infactible ya que el evento 1 está asignado a dos salones (salones 1 y 2). Adicionalmente, a los eventos 2 y 4 no se les ha asignados salón. Lo mismo sucede con el hijo 2, donde los eventos 2 y 4 tienen dos salones asignados y el evento 1 no tiene salón. Por tal motivo se hace necesario someter los hijos obtenidos a un proceso de factibilización en el cual, entre los eventos con 2 o más asignaciones, se conserva uno de los salones asignados de forma aleatoria y los demás se descartan, es decir, la asignación toma el valor de 0. Para el caso de los eventos que no tienen asignado salón, se procede de la siguiente manera: se escoge de forma aleatoria tanto un bloque de tiempo como un salón, considerando que el salón escogido no haya sido asignado en ese bloque a otros eventos. Si ningún salón puede ser asignado, entonces se debe escoger otro bloque de forma aleatoria. De esta manera se garantiza una recombinación factible.

3.4 Mutación

El proceso de mutación se encuentra fuertemente ligado al concepto de vecindad, lo cual permite una amplia gama de propuestas. En este trabajo, el cromosoma hijo obtenido en el proceso de recombinación es sometido al proceso de mutación, el cual consiste en escoger, de forma aleatoria, dos eventos e intercambiar los salones asignados a cada uno de ellos. Para realizar mutaciones más fuerte es posible aplicar dos o más veces el proceso descrito al mismo hijo.

3.5 Disminuir infactibilidad

Las alternativas infactibles son aquellas que presentan violaciones en las restricciones duras. El algoritmo que se propone incluye un proceso que busca disminuir la infactibilidad del hijo mutado anteriormente de forma gradual. Este proceso es aplicado a las alternativas infactibles y consiste en disminuir ya sea el número total de cruces en los horarios de los estudiantes, el número total de cruces en los horarios de los salones o el número de requerimientos no satisfechos. El proceso consiste en escoger, de forma aleatoria cualquiera de las tres restricciones duras anteriores y realizar el siguiente proceso según el caso:

i. Disminución del número total de cruces en los horarios de los estudiantes: para el bloque de tiempo en el que se produzca un mayor número de cruces entre estudiantes, se escoge aleatoriamente uno de los eventos con salón asignado para reasignarlo al bloque de tiempo con menos cantidad de cruces. Si existen varios bloques candidatos para la reasignación se escoge uno aleatoriamente.
ii. Disminución del número total de cruces en los horarios de los salones: para el bloque de tiempo en el que se produzca un mayor número de cruces entre salones, se escogen los eventos pertenecientes al salón que más se cruce, y entre estos se escoge uno de forma aleatoria para ser reasignado a otro salón.
iii. Disminución del número de requerimientos no satisfechos: se debe identificar un salón que no cumpla con los requerimientos de algún evento en particular y reasignarlo aleatoriamente a uno de los salones que si cumpla.

Después de aplicar cualquiera de las tres estrategias anteriores se debe evaluar la factibilidad de la alternativa a través de la expresión:

$$\underbrace{\left(w_1 \cdot \mathrm{Es}_{\mathrm{C}} + w_2 \cdot \mathrm{S}_{\mathrm{RNS}} + w_3 \cdot \mathrm{S}_{\mathrm{C}}\right)}_{\text{Restricciones duras}} \qquad (11)$$

Si la factibilidad no disminuye es necesario aplicar de nuevo cualquiera de las estrategias al hijo mutado original. El proceso se repite un número determinado de veces o hasta disminuir la infactibilidad.

3.6 Mejorar optimalidad

Este proceso busca mejorar el valor de la función objetivo asociado a las restricciones blandas sin desmejorar el valor de la infactibilidad y puede ser realizado consecutivamente dos o más veces para acelerar la convergencia. Lo anterior puede traer como consecuencia convergencias a soluciones de mala calidad, por lo cual se debe elegir valores adecuados para cada caso en particular. El proceso consiste en evaluar la alternativa actual haciendo que en cada bloque de tiempo cada uno de los salones sea asignado a cada uno de los eventos, teniendo en cuenta realizar una reasignación a la vez. Las reasignaciones que mejoren la función objetivo y que mejoren o no empeoren la infactibilidad son almacenadas. Finalmente, de todas las reasignaciones almacenadas se escoge la que mas favorezca la función objetivo. Es importante notar que la alternativa resultante se diferencia de la alternativa original solamente en un elemento.

3.7 Modificar la población actual

Las principales características del algoritmo memético que se presenta en este documento están asociadas a la modificación de la población actual. El algoritmo completo, después de generada la población inicial, es el siguiente: Repetir un número determinado de iteraciones los pasos del 1 al 6.

1. Obtener 2 alternativas padre de la población actual usando el proceso de selección.
2. Se obtiene una alternativa hijo aplicando Recombinación a los padres obtenidos en el paso anterior.
3. Se obtiene una alternativa modificada aplicando Mutación (hijo mutado).
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, sino, 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 toda 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.

4. PRUEBAS Y RESULTADOS

Los casos de prueba pueden ser descargados de la dirección de Internet:
http://www.idsia.ch/Files/ttcomp2002/

La tabla 1 muestra valores de funciones objetivo para varios casos de prueba resueltos por diferentes métodos encontrados en la literatura especializada. Se observa que los mejores resultados han sido registrados por la metaheurística basada en Simulated Annealing, sin embargo, el algoritmo memético propuesto supero los resultados obtenidos por las otras tres metodologías.

	Número de eventos	Número de salones	número de características	Número de estudiantes	Valor de la función objetivo usando: Algoritmo memético propuesto	Heurística basada en Simulated annealing	Búsqueda tabú	Algoritmo de búsqueda local	Búsqueda con memoria adaptiva
Caso 1	350	10	5	300	42	13	92	199	133
Caso 2	350	10	5	350	118	44	118	194	161
Caso 3	400	10	5	250	54	29	66	128	131
Caso 4	440	11	6	220	109	17	51	126	126
Caso 5	400	10	5	200	81	61	81	147	147

Tabla 1. Resultados obtenidos

Los tiempos computacionales (en segundos) registrados para cada caso son respectivamente: 255, 293, 307, 322, 512. Los otros métodos no registran en la literatura especializada los tiempos computacionales, por lo cual no se hacen comparaciones al respecto.

5. CONCLUSIONES Y RECOMENDACIONES

El método heurístico propuesto proporciona resultados interesantes en la solución del problema de optimización de horarios en cuanto a calidad de la respuesta y velocidad de convergencia.

La estrategia de modificación de la población actual evita convergencias prematuras y asegura un alto factor de diversidad. 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. A diferencia de los algoritmos genéticos tradicionales, no se modifica la población de forma aleatoria.

La solución a este tipo de problemas puede mejorar significativamente si se consideran metodologías multiobjetivo las cuales cobran mayor importancia cuando aumenta la complejidad del problema al crecer en tamaño y al incorporar restricciones adicionales tales como: preferencias de horarios, de materias y de salones por parte de los profesores y distancias de desplazamiento por parte de estudiantes, entre otras.

6. BIBLIOGRAFÍA

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

[2]. FERLAND, J.A. and ROY, S., Timetabling Problem for University as Assignment of Avtivities to Resources. Computers and Operations Research, 12:2. 1985, 207-218.

[3]. AUBIN, J. and FERLAND, J.A., A large Scale Timetabling Problem. Computers and Operations Research, 16:1. 1989, 67-77.

[4]. CARTER, M.W., A Lagrangian Relaxation Approach to the Classroom Assignment Problem. Computers and Operations Research, 27:2. May 1989, 230-246

[5]. MULVEY, J.M., A Classroomtime Assignment Model. European Journal of Operational Research. 9, 1983, 64-70.

[6]. DINKEL, J.J., MOTE, J. and VENKATARAMANAN, M.A. An Efficient Decision Support System for Academic Course Scheduling. Operations Research, 37:6. 1989, 853-864

[7]. WHITE, G.M. and WONG, K.S., Interactive Timetabling in Universities. Computers in Education, 12:4, 1988, 521-529.

[8]. GRANADA M., TORO E. M., TABARES P. Método de Colonia de Hormigas Aplicado a la Solución del Problema de Asignación Generalizada. Revista Tecnura No 15, Universidad Distrital F.J.C., II-2004, Colombia.

[9]. GRANADA M., TORO E. M., ROMERO R, Algoritmo memético aplicado al problema de asignación generalizada, Revista Tecnura No 16, Universidad Distrital F.J.C., I-2005, Colombia.