Experimentos con Algoritmos Genéticos para resolver un problema real de Programación Maestros-Horarios-Cursos

Pedro FLORES, Ernesto BRAU, Jazmín A. MONTEVERDE, Norman F. SALAZAR, José FIGUEROA, Eliseo CADENA, Caleb A. LIZÁRRAGA.
Departamento de Matemáticas Universidad de Sonora,
Hermosillo Sonora CP 83000 México.

RESUMEN

En las universidades de México se presentan dos tipos de estructuras educativas: de escuelas y de departamentos. Las universidades que están organizadas por escuelas se programan, en cada escuela, todos los cursos de las carreras que ahí se imparten. Por otra parte, en las universidades que están organizadas por departamentos, se programan únicamente los cursos de su área de conocimiento, tanto para las carreras que imparte, como para todas las carreras de los demás departamentos de la institución. Esto origina que en algunos departamentos, por ejemplo matemáticas, el número de cursos a programar sea grande, además de que es necesario considerar más restricciones al elaborar una programación. En este trabajo se presentan los resultados obtenidos, al resolver con Algoritmos Genéticos, un modelo desarrollado para el Departamento de Matemáticas de la Universidad de Sonora que consideramos, es un caso general de todos los departamentos de matemáticas de las universidades que tienen este tipo de organización en México. Para este problema presentamos las restricciones y los objetivos que deben ser cumplidos, así como los resultados que se obtuvieron al aplicar Algoritmos Genéticos para la programación de cursos. Los resultados son muy satisfactorios.

Palabras Claves: Algoritmos, Genéticos, Programación, Maestros, Horarios, Cursos.

1. INTRODUCCIÓN

La programación de cursos, con sus respectivos profesores, en una institución académica es una tarea en la cual una gran variedad de aspectos deben ser considerados, así como el hecho de que dichos aspectos cambian de un ciclo escolar a otro. El tiempo invertido en esta actividad para elaborar una propuesta, que sea lo más completa posible es muy grande. Adicionalmente, la programación de cursos siempre es realizada por varios miembros del cuerpo administrativo de la institución. En el caso del Departamento de Matemáticas de la Universidad de Sonora, se estima que, cada año, un grupo de cuatro administradores requieren de 500 horas-hombre para llevar a cabo tal programación.

El problema de programación maestros-horarios-cursos (PMHC), consiste en elaborar una programación de cursos que logra cumplir, lo mejor posible, los compromisos laborales y académicos de la institución, tomando en cuenta la disponibilidad de horario de los profesores, los posibles cursos que estos pueden impartir, la disponibilidad de espacios físicos, etc., además, en el proceso de construir una programación definitiva aparecen varios conflictos, los cuales es deseable que sean fáciles de resolver.

Es conocido que el problema de PMHC es NP-Hard [5], y, debido al grado de dificultad involucrado en resolver este tipo de problemas, es un área de investigación muy activa. Una prueba de esta última afirmación es la Conferencia Internacional de Practice and Theory of Automated Timetabling (PATAT), realizada bianualmente. Su quinta edición se llevará a cabo el próximo año 2004. Además, existen grupos tales como el Working Group on Automated Time Tabling (Euro-WATT) el cual es parte de la Association of European Operational Research Societies, cuyo objetivo es mantener información actualizada acerca de este tema en la web. Cabe señalar que la Euro-WATT tiene un foro de discusión sobre la implementación de Algoritmos Genéticos (AG) en el problema de PHMC.

La gran diversidad de modelos para este problema se debe al hecho de que cada uno tiene que ser resuelto a partir de un conjunto de características muy específicas, que dependen de cada institución, lo cual obliga a considerar restricciones muy distintas en cada caso. Varias técnicas se han propuesto con el propósito de resolver este tipo de problemas [1, 3, 4, 7, 10], incluyendo el uso de AG [2, 9]. Este artículo presenta un modelo desarrollado para el Departamento de Matemáticas de la Universidad de Sonora, cuyas características principales son: cada uno de los cursos se programa a la misma hora y en la misma aula durante toda la semana y con una base de cinco horas de tiempo de clase a la semana. Adicionalmente, todas las aulas donde se llevan a cabo las clases son del mismo tipo.

En la segunda sección se presenta la construcción del modelo, incluyendo las restricciones a considerar. Dadas las características del problema, es posible hacer una representación vectorial de toda la información; esto simplifica la implementación de los AG. En la sección tres presentamos algunos de sus fundamentos y en la cuarta sección describimos los diferentes AG que utilizamos para resolver el problema. Posteriormente, en la sección cinco, se mostrará, por medio de gráficas, los resultados de las diferentes corridas de cada algoritmo. Finalmente, se presentan las conclusiones obtenidas.

2. LA CONSTRUCCION DEL MODELO

En las universidades mexicanas, los departamentos presentan dos tipos de situaciones en su programación de cursos. El primero caso corresponde a un área que llamaremos “área de servicios”, donde los distintos departamentos de la universidad le piden al departamento los cursos que este puede impartir, seleccionando de antemano el aula y el horario en las que serán impartidas; entonces, la tarea de nuestro departamento es asignar un profesor a cada uno de estos cursos. Para el Departamento de Matemáticas de la Universidad de Sonora, esta es la situación de los cursos solicitados por las carreras de: Sociología, Derecho, Químico Biólogo, Psicología,

Comunicación, Economía, Ingeniería Industrial y de Sistemas, Administración y Contabilidad. Por otra parte, existe un área llamada, propiamente: "del Departamento", en donde se consideran los cursos que son impartidos a las carreras que ofrece el Departamento, ó carreras muy afines a esta. Este es el caso de los cursos de las carreras de: Ingeniero Civil, Ingeniero Minero, Ingeniero Químico, Geólogo, Licenciado en Física, Licenciado en Ciencias de la Computación y Licenciado en Matemáticas. En este último caso, los cursos no tienen asignada hora y lugar de antemano, así que esto debe de ser determinado. El Departamento de Matemáticas programa, aproximadamente, 325 clases de 105 cursos distintos con 140 maestros, 2 veces al año.

El modelo que aquí se construye considera los siguientes aspectos:

- La programación de maestros debe ser con el criterio de antigüedad y nivel académico de los maestros.

- Existe, para cada curso, un número preestablecido de veces que debe ser programado.

- Existe, para cada curso, un número fijo de aulas donde deben ser acomodados los cursos.

- Cada profesor puede impartir solamente ciertos cursos y en cierto horario, el cual establece el profesor.

- Algunos cursos deben ofrecerse a una hora establecida de antemano.

- A algunos profesores se les debe asignar ciertos cursos.

- El número de cursos asignados a cada profesor debe estar entre un valor mínimo y uno máximo.

- Hay profesores que pueden impartir clases en diferentes departamentos.

Según la revisión bibliográfica que se ha realizado, este problema de PMHC no ha sido tratado con anterioridad.

3. ALGORITMOS GENETICOS

Los AG fueron descubiertos y formalizados teóricamente por John Holland en 1975. Su aplicación principal es en la solución de problemas en la búsqueda y optimización y están basados en los métodos de evolución biológica: selección natural, reproducción y mutación. Desde sus inicios han sido usados en problemas reales tales como: optimización y control de tuberías de gas, diseño de redes de comunicación, diseño de armaduras metálicas, etc. [6]

Para resolver un problema particular, utilizando AG, es necesario, como se señala en [8], contar con:

1) Una representación genética del problema (el individuo). Esta representación deberá ser codificada con cadenas de longitud finita sobre un conjunto finito de símbolo.

2) Alguna manera de crear la población inicial, la cual puede ser aleatoria o seleccionada por otros métodos.

3) Una función de desempeño que revise que tan adaptado se encuentra un individuo al medio y los separe, por un procedimiento llamado selección, según su desempeño.

4) Operadores genéticos que alteran la estructura de los hijos; normalmente se consideran el cruzamiento y la mutación.

5) Los valores de varios parámetros que son requeridos por los AG, como son: el tamaño de la población (*pop_size*), probabilidad de cruzamiento (*pc*), y probabilidad de mutación (*pm*).

4. SOLUCIÓN DEL PROBLEMA CON AG

Para nuestro problema se nos dio la información del período escolar agosto-diciembre del 2003 de cada profesor con respecto a los cursos que puede impartir, así como su horario disponible. También, la antigüedad y clasificación (tiempo completo o medio tiempo) de cada profesor es incluida. Por otra parte, los requerimientos académicos, la disponibilidad de espacios físicos y los compromisos laborales por parte de la administración se encuentran establecidos de antemano y toda esta información debe de ser codificada. A continuación presentamos los detalles de la codificación del modelo para que pueda ser resuelto con AG.

Representación de los individuos.
Para que sea factible resolver el problema, se necesita tener la representación de una posible solución, la cual es llamada individuo. En este problema en particular, el individuo es representado por un vector de números enteros, el cual se determina por otros tres vectores: vector de profesores, vector de horarios y vector de cursos. Estos tres vectores son definidos por todas las posibles combinaciones de profesores, horarios y cursos. Por ejemplo, supongamos tener dos profesores, tres horarios y tres cursos, asumiremos que el primer profesor puede impartir el curso 0 y el curso 1, el segundo profesor puede impartir el curso 0 y el curso 2. Asimismo, el profesor 0 esta disponible en los horarios 0 y 1, mientras que, el profesor 1 tiene disponibles los 3 horarios. Si los cursos 0, 1 y 2 pueden ser programados en estos tres horarios, los vectores se verían así:

Vector de profesores:	(0, 0 | 0, 0 | 1, 1 | 1, 1 | 1, 1)
Vector de Horarios	(0, 0 | 1, 1 | 0, 0 | 1, 1 | 2, 2)
Vector de cursos:	(0, 1 | 0, 1 | 0, 2 | 0, 2 | 0, 2)

Cuando los vectores se encuentran alineados, tenemos que: el profesor 0 puede impartir en el horario 0 los cursos 0 y 1, así como en el horario 1. Hay que hacer hincapié en como estos vectores están separados en bloques, los cuales juegan el rol de genes. Cada bloque representa el horario disponible para cada profesor. El vector de individuos es llenado de ceros y unos, siendo un 1 si el profesor es asignado a ese curso en ese horario, y 0 si no. Se puede observar que en cada bloque solo puede haber un solo 1; de lo contrario significaría que el profesor tiene asignado dos cursos en el mismo horario. El vector de individuos se podría ver de la siguiente manera:

Vector de profesores:	(0, 0 | 0, 0 | 1, 1 | 1, 1 | 1, 1)
Vector de Horarios:	(0, 0 | 1, 1 | 0, 0 | 1, 1 | 2, 2)
Vector de cursos:	(0, 1 | 0, 1 | 0, 2 | 0, 2 | 0, 2)
Vector de individuos:	(0, 1 | 1, 0 | 0, 0 | 1, 0 | 0, 1)

Esto significa que el profesor 0 fue asignado en el horario 0, al curso 1 y en el horario 1 al curso 0. El profesor 1 fue asignado, en el horario 1, curso 0 y en el horario 2, el curso 2, dejando en blanco el tiempo 0.

Con este modelo nuestro problema tenia 827 variables y 336 bloques y 754 restricciones. En la versión final del código, usamos una representación vectorial diferente, la cual es mas compacta para el tratamiento de la información.

Los vectores de Individuos pasan a constituir los renglones de dos matrices llamadas, respectivamente, padres e hijos, cada una con el mismo número de individuos. A este número es al que llamamos tamaño de la población, ó *pop_size*.

Generando la Población Inicial
Este procedimiento es para iniciar la matriz de padres. Para esto, se generan los renglones de acuerdo al desarrollo que se presenta a continuación.

Para generar los individuos de la población inicial, se inicializan los individuos con todos sus valores iguales a 0, después, para cada individuo, se seleccionan un número diferente de bloques, para cada individuo, igual al número de cursos a ser programados; posteriormente, se asigna un 1 aleatoriamente a cada uno de estos bloques. El resultado de esta selección es que los individuos de la población inicial representan una programación en la cual el número de cursos es igual al del resultado deseado.

Función de Desempeño y Mecanismo de Selección
La selección es el procedimiento a partir del cual se construye la matriz de hijos tomando como origen la matriz de padres y representando el fenómeno que ocurre en la naturaleza de la supervivencia del más apto. La forma de medir la aptitud es por medio de una función de desempeño.

En nuestro problema se construyen funciones como en [3], las cuales calculan el error de no cumplir con las condiciones requeridas por la programación. Estas funciones comparan los valores obtenidos al tomar la programación de un individuo con los requisitos que se deben de cumplir en la programación deseada. Estos errores son:

1) Errores en los espacios físicos.

2) Errores en los cursos que imparten los maestros.

3) Errores en los requerimientos de los cursos. Estos errores consideran todos los diferentes requisitos que los cursos deben tener.

4) Errores en los horarios de los cursos.

Además, un valor constante es asignado a cada profesor, el cual es utilizado para modelar la prioridad en la programación. Para los profesores con la prioridad más alta, el valor de la constante es bajo, mientras los profesores con prioridad baja tienen un valor más alto para esta constante. La prioridad de una programación, denotada por *pe*, se mide sumando las constantes de los profesores que fueron seleccionados en el vector solución.

Este problema requiere la selección de constantes por cada error y constante para *pe*. Estas constantes permiten darle más importancia a un error sobre otro, de los presentados en esta sección. Para resolver este problema se utiliza una versión de AG, la cual se encarga de minimizar una función de desempeño igual a la suma de las respectivas constantes multiplicadas por los errores y *pe*, es seleccionada.

En este trabajo se probaron dos métodos distintos como mecanismo de selección: la ruleta y el torneo binario

El método de la ruleta: consiste, como su nombre los sugiere, en simular una ruleta. Esto se logra creando un espacio de probabilidad, en el cual los individuos más fuertes ocupan un mayor rango, de tal forma que tienen una mayor probabilidad de reproducirse.

El torneo binario es un método que consisten en seleccionar aleatoriamente dos individuos y dejar que compitan, es decir ambos errores son comparados y el individuo con el menor error pasa a la siguiente generación. Esto se repite hasta que la matriz con los hijos este lleno.

Operadores Genéticos.
Los operadores genéticos que utilizamos son el cruzamiento y la mutación.

En cuanto al cruzamiento, se probó con tres métodos: cruzamiento simple, doble y uniforme. Para todos estos cruzamientos se van tomando parejas de individuos y, con probabilidad *pc*, se decide si se cruzan ó no de acuerdo al tipo de cruzamiento que se trate. Si se decide no cruzar dos individuos, estos pasan directamente a formar parte del conjunto de hijos.

El cruzamiento simple consiste en escoger un bloque aleatoriamente del vector de individuos y llevar acabo el cruzamiento desde ese bloque en adelante. El cruzamiento doble es básicamente el mismo proceso que el de cruzamiento simple, excepto que este es repetido; esto significa que dos bloques son elegidos aleatoriamente y dos cruzamientos son llevados acabo. El último método utilizado fue el de cruzamiento uniforme. En este método, cada bloque (gen) del individuo descendiente tiene la misma probabilidad de venir de alguno de los dos individuos padres de que se trate, esto significa que cada gen es elegido aleatoriamente de un ancestro.

El operador de mutación consiste en seleccionar aleatoriamente un bloque del vector solución y dependiendo del caso aplicar una de las siguientes. Si el bloque consiste de puros ceros, uno de ellos se cambia a 1 aleatoriamente. Esta acción simboliza asignarle a un profesor un curso en un horario que tenga disponible. Si el bloque contiene un 1, entonces un lugar es seleccionado aleatoriamente y el 1 es cambiado a ese lugar – a menos que el lugar seleccionado es aquel donde el 1 se encuentra localizado, en tal caso el 1 se cambia a 0.

Parámetros utilizados por los Algoritmos genéticos.
Los parámetros son los siguientes: tamaño de población, probabilidad de cruzamiento y probabilidad de mutación. Además, un parámetro por cada error es utilizado más *pe* (para modelar la prioridad). Estos parámetros deben ser encontrados para cada problema en particular; esto explica por que son probados diferentes tamaños de población, probabilidades de cruzamiento y mutación.

5. RESULTADOS DE LA PROGRAMACIÓN POR COMPUTADORA.

Como se mencionó antes, los cursos de matemáticas del Departamento de Matemáticas de la Universidad de Sonora están separados en dos áreas: Área de Servicio y Área del Departamento. Para nuestro modelo se consideran al mismo tiempo las dos áreas y se resuelve el problema para ambas simultáneamente.

Para llevar a cabo la programación por computadora, se procedió de la siguiente manera: la información que es usada por el Departamento de Matemáticas para programar los cursos de los otros departamentos y de sí mismo fue codificada en un formato que el programa pudiera entender. Los datos fueron alimentados en el modelo con las especificaciones mencionadas anteriormente y fue implementado en una computadora personal con micro procesador Pentium(R) 4 de 2.4 GHz. Los resultados fueron obtenidos en menos de un segundo.

Para las corridas se estableció un valor igual a 1 para todas las constantes que multiplican a los errores, y de 0 para *pc*. Con estos valores una programación adecuada para el Departamento se representaría por un individuo con valor de función de desempeño igual a 0.

Los experimentos que se realizaron están divididos en tres etapas. En la primera etapa, se encontraron los mejores parámetros para un solo tipo de algoritmo que fue el que utiliza la ruleta y el cruzamiento simple. De esta etapa no se presentan resultados. En la segunda etapa, se realizaron corridas con los parámetros encontrados en la primera etapa con el fin de encontrar el tipo de algoritmo que tenía mejor desempeño. En la última etapa se trató de afinar más los resultados de los parámetros para el algoritmo con torneo y
cruzamiento uniforme. En los experimentos realizados en esta etapa (no se presentan resultados) se vio que no es necesario modificar los parámetros para mejorar el desempeño del algoritmo.

Durante la primera etapa para el AG se fijó como base el cruzamiento simple y el método de la ruleta, y se buscaron los mejores parámetros para este algoritmo. Los parámetros que se probaron fueron los siguientes: tamaño de población (*pop_size*), para lo cual se hicieron pruebas con valores de 50, 80, 100, 150 y los mejores resultados que se obtuvieron fueron con un tamaño de 100; la probabilidad de cruzamiento (*pc*) fue variada entre 0.50 y 0.90, con incrementos de 0.05, obteniendo mejores resultados con 0.85; la probabilidad de mutación (*pm*) fue originalmente fijada a 0.001 y se hicieron pruebas con incrementos de 0.01 hasta el valor de 0.17, el valor que produjo mejores resultados fue 0.15. En resumen, los parámetros con los que se obtuvieron los mejores resultados son: *pc* = 0.85, *pm* = 0.15 y *pop_size* = 100.

Con los parámetros mencionados se realizaron grupos de 50 corridas del sistema, las cuales fueron hechas con cada par de selección y cruzamiento posible –torneo binario con cruzamiento simple, ruleta con cruzamiento simple, etc.- y fue tomada la media del error. Los resultados obtenidos se muestran en las siguientes gráficas:

Las figuras 1,2 y 3 muestran los resultados usando ruleta con tres diferentes tipos de cruzamiento.

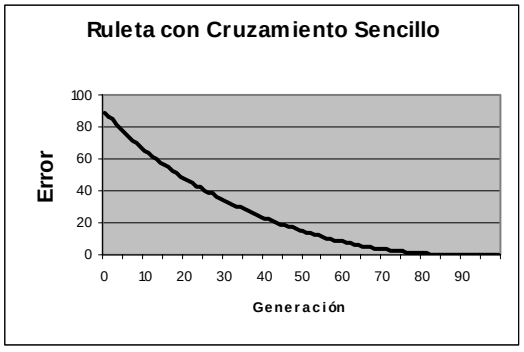

Figura 1

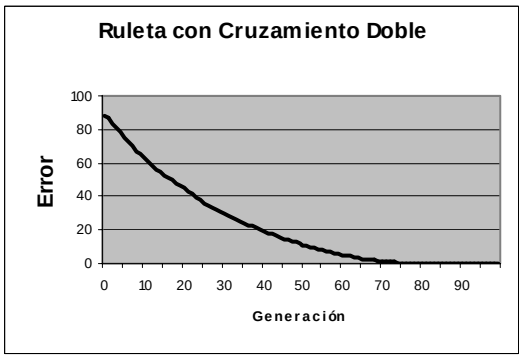

Figura 2

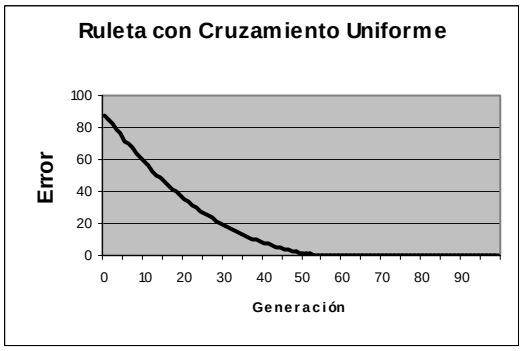

Figura 3

Las figuras 4, 5 y 6 muestran los resultados usando torneo binario con tres diferentes tipos de cruzamiento.

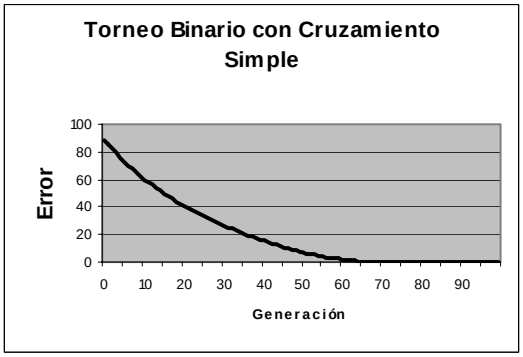

Figura 4

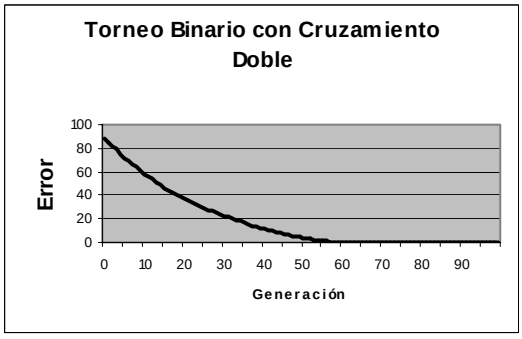

Figura 5

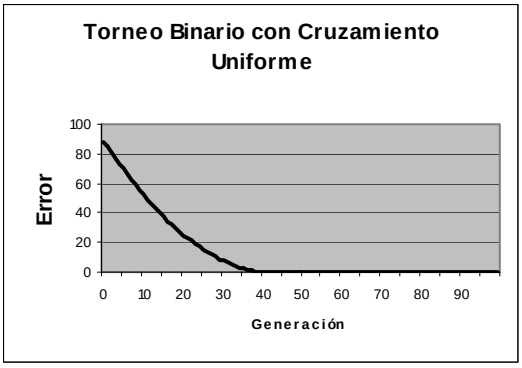

Figura 6

Finalmente, en la figura 7 mostramos los resultados de este problema usando torneo binario con cruzamiento uniforme y probabilidades de mutación de 0.01 y 0.15.

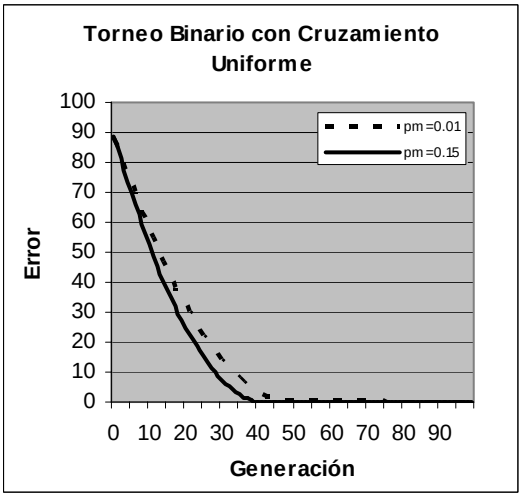

Figura 7

Por último, se presentaron para su evaluación las programaciones obtenidas con AG a los responsables del Departamento de Matemáticas y su opinión fue que se podían considerar como equivalentes a la programación elaborada manualmente por ellos.

6. CONCLUSIONES

De acuerdo a las graficas obtenidas de las pruebas llevadas a cabo, uno puede observar el comportamiento de cada uno de los métodos para resolver el problema, de donde lo primero que observamos es que todos los métodos resuelven el problema con los parámetros aquí presentados.

Es importante indicar que el método en el cual la mayoría de los errores convergieron a cero más rápidamente fue el de torneo binario con cruzamiento uniforme, por lo que concluimos que este método es el mejor para este problema. En contraste, cuando usamos ruleta con cruzamiento sencillo, aunque la solución con error cero fue obtenida, no es la mejor opción ya que fue la que consumió mayor tiempo de procesamiento.

Por último, es de resaltar que, no obstante el tamaño y la dificultad del problema, la convergencia es sumamente rápida. Además, los resultados obtenidos para los valores de *pc* y *pm* son altos comparados con los obtenidos en diversos problemas tratados con AG.

7. AGRADECIMIENTOS

Se agradece a Alina García y Gabriela Saralegui por su colaboración en la codificación del problema; al profesor René Leyva por su información de la calendarización del departamento y a los profesores Guillermo Dávila y Francisco Salazar por sus observaciones a este artículo.

8. REFERENCIAS

[1] Aubin, J. & Ferhand J.A. “A large Scale Timetabling Problem”. **Computers Operationals Research**, Vol. 16 No. 1, 1989. Pergamon Press, pp.67-77.

[2] Carrasco M. P., Pato M. V. “A multiobjetive Genetic Algorithm for the Class/Teacher Timetabling Problem”. **Proceedings of the 3nd International Conference for the Practice and Theory of Automated Timetabling (PATAT 2000)**, Springer-Verlag Lecture Notes in Computer Science Volume 1408, pp. 3-17.

[3] Costa, D. “A Tabu Search Algorithm for Computing and Operational Timetable”. **European Journal of Operational Research** 76 (1994) pp. 98-100.

[4] De Werra, D. “An Introduction to Timetabling”. **European Journal of Operational Reserach** (1985) pp 151-162.

[5] Even, Sl, Atai, A., and Shamir, A. “On the commplexity of timetable and multicommodity flow problems”. **SIAM Journal on Computing** 5(1976) pp. 691-703.

[6] Goldberg D.E.: Genetic **Algorithms in Search, Optimization and Machine Learning**. Addison-Wesley Publishing Company, Inc. 1989.

[7] Lawrie, N.L. “An Integer Linear Programming Model of a School Timetabling Problem”. **Computer Journal** 12 (1969) pp 307-316

[8] Michalewicz, Z.: **Gentic Algoritthms + Data Structures = Evolution Programs**. New York, USA: Springer Verlag. Second, extended edition. 1994.

[9] Ross P., Hart E., Corne D. “Some Observations about GA based Timetabling”. **Proceedings of the 2nd International Conference for the Practice and Theory of Automated Timetabling (PATAT '97)**, Springer-Verlag Lecture Notes in Computer Science Volume 1408, pp. 115-129.

[10] Wood, D.C. “A technique for Coloring a Graph Applicable to Large Timetabling Problems”. **Computer Journal**, Vol. 12, 1968, pp. 317-319.