# COLONIA DE HORMIGAS APLICADA A LA PROGRAMACIÓN ÓPTIMA DE HORARIOS DE CLASE

Application of Ant Colony Optimization to course timetabling problem

**RESUMEN**
Se presenta el problema de la programación óptima de horarios de clase, así como el modelo matemático que lo representa y la codificación empleada. Se muestra la aplicación de la técnica de optimización combinatorial Colonia de Hormigas en la solución del problema y se usa la metodología desarrollada en casos de prueba de la literatura especializada.

**PALABRAS CLAVES:** Colonia de Hormigas, metaheurísticas, optimización combinatorial, programación de horarios de clase.

**ABSTRACT**
*Course timetabling problem is presented, as well as its mathematical model and the codification that was used. Application of Ant Colony is shown in order to solve the problem and the methodology developed is tested in several cases.*

**KEYWORDS:** *Ant Colony, combinatorial optimization, course timetabling, metaheuristics.*

**CESAR AUGUSTO PEÑUELA**
Ingeniero Electricista.
Profesor Catedrático
Programa de Ingeniería Eléctrica
Universidad Tecnológica de Pereira
cesar_penuela@utp.edu.co

**JOHN FREDY FRANCO B.**
Ingeniero Electricista, M.Sc.
Profesor Catedrático
Programa de Ingeniería Eléctrica
Universidad Tecnológica de Pereira
jffb@utp.edu.co

**ELIANA M. TORO O.**
Ingeniera Industrial, M. Sc.
Profesor Asistente
Facultad de Ingeniería Industrial
Universidad Tecnológica de Pereira
elianam@utp.edu.co

## 1. INTRODUCCIÓN

La asignación óptima de horarios en instituciones educativas ha sido reconocida como un problema de difícil solución, el cuál puede ser definido como un problema de asignación de un número de eventos en un número limitado de períodos de tiempo. Wren en [1] lo define como: "Un problema de asignación, sujeto a un conjunto de restricciones y recursos disponibles en un espacio de tiempo, de tal manera que se satisfaga en la medida de lo posible un conjunto de objetivos deseables".

Dependiendo del número de eventos (cursos o exámenes a programar) el espacio de soluciones del problema crece de forma exponencial, y por lo tanto su solución puede convertirse en una tarea difícil. Este tipo de problemas ha atraído la atención de la comunidad científica de una serie de disciplinas, incluyendo la Investigación operativa y la Inteligencia artificial desde hace 40 años.

La comunidad científica ha desarrollado un concurso denominado *International Timetabling Competition* donde se aborda el problema de programación horaria dividido en dos categorías principales: programación de exámenes y programación de horarios, sujetos a dos clases de restricciones: "duras" o de estricto cumplimiento y "blandas" que es preferible que se cumplan pero que pueden no hacerlo.

Como ejemplo de las restricciones duras puede mencionarse que dos clases o dos exámenes no pueden programarse en igual horario en un mismo salón. Las restricciones blandas o suaves son deseables, pero en soluciones reales pueden llegar a ser difíciles de satisfacer completamente; por ejemplo programarle a un docente todas sus clases en un aula de su preferencia o que los estudiantes no deberían tener dos exámenes consecutivos. Un análisis más detallado del tipo de restricciones que se pueden suponer en la programación de exámenes se puede revisar en [2].

Una gran variedad de métodos se han descrito en la literatura para resolver el problema, habiendo sido probados con casos reales. Estos métodos se dividen en cuatro tipos: métodos secuenciales [3],[4], [5], métodos de clusterización [6], [7], métodos basados en restricciones [8], y los métodos meta-heurísticos [9], contando estos últimos con un gran desarrollo en la últimas dos décadas, con técnicas como recocido simulado, Búsqueda Tabú, Genéticos, métodos Híbridos, entre otros [10,11]. A la fecha no se encuentra en la literatura especializada un desarrollo basado en Colonia de Hormigas como método de solución para resolver este tipo de problema.

En este trabajo se propone la técnica de Colonia de Hormigas [12] para resolver el problema de programación óptima de horarios de clase. En la sección 2 se explica el tipo de problema que se está resolviendo. En la sección 3 se describen las distintas estructuras

Fecha de Recepción: 25 de Enero de 2008
Fecha de Aceptación: 9 de Abril de 2008

usadas para el manejo de los datos, la codificación empleada y se presenta el modelo matemático. En la sección 4 se muestra la metaheurística Colonia de Hormigas y su aplicación al problema. En la sección 5 se presenta la aplicación de la metodología desarrollada en casos de prueba de gran tamaño que pueden ser encontrados en [13]. Finalmente, se presenta una sección con conclusiones y recomendaciones para trabajos futuros.

## 2. PROBLEMA DE PROGRAMACIÓN ÓPTIMA DE HORARIOS DE CLASE

El problema de generación de horarios consiste en programar un conjunto de eventos de clase en 45 horas repartidas en 9 horas para 5 días a la semana, conocido un conjunto de salones en los cuales tomarán lugar los eventos, un conjunto de estudiantes que asisten a los eventos, y un conjunto de características que presentan los salones y requieren los eventos. Cada estudiante asiste a cierto número de eventos y cada salón tiene una determinada capacidad. Un horario factible es aquel en el que todos los eventos han sido asignados a una hora y a un salón de tal manera que las siguientes restricciones duras se satisfacen:

- En un periodo de tiempo dos o más eventos no pueden estar programados en el mismo salón.
- El salón seleccionado cumple con las características requeridas por el evento y tiene la capacidad suficiente para todos los estudiantes que asisten al evento.
- Un estudiante no puede asistir a más de un evento en cada periodo de tiempo.

Además es preferible que se evite la violación de las siguientes restricciones blandas:

- Un estudiante tenga programado más de dos eventos consecutivos.
- Un estudiante tenga un único evento en un día.
- Un estudiante tenga un evento en la última hora del día.

De esta forma el problema se puede formular como la programación de los horarios de clase definiendo la hora y el salón para cada evento de tal forma que cumpliendo las restricciones duras se minimicen las restricciones blandas.

Para el primer día se toma desde la hora 1 hasta la hora 9 y así sucesivamente hasta el quinto día, al que le corresponden las horas 37 hasta la 45 de la semana.

## 3. CODIFICACIÓN Y MODELO MATEMÁTICO

Para el manejo de los datos del problema de programación óptima de horarios de clase se usan matrices para almacenar la información suministrada.

La información de las características requeridas por los eventos se almacena en la matriz características_eventos, en la que para cada evento (filas), respecto a cada característica (columnas), aparece un ‘1’ si el evento requiere la característica o aparece un ‘0’ en caso contrario.

La información de las características que tienen los salones se almacena en la matriz *características_salones*, en la que para cada salón (filas), respecto a cada característica (columnas), aparece un ‘1’ si el salón tiene la característica o aparece un ‘0’ en caso contrario.

Se tiene un vector que indica la capacidad de los salones: capacidad_salones, en el que en la posición ‘i’ se almacena la capacidad del salón ‘i’.

La información de los estudiantes que están matriculados en cada evento se representa en la matriz eventos_estudiantes, en la que en la posición (i,j) tiene un ‘1’ si el estudiante ‘j’ (columna) asiste al evento ‘i’ (fila) o tiene un ‘0’ en caso contrario.

A partir de las matrices características_eventos, características_salones y del vector capacidad_salones se puede crear la matriz eventos_salones que en la posición (i,j) tendrá un ‘1’ si el evento ‘i’ (filas) se puede programar en el salón ‘j’ (columnas) o tendrá un ‘0’ en caso contrario. Esta matriz resume las condiciones de características y capacidad de los salones frente a los eventos.

La programación de los eventos se puede representar en una matriz (evento_hora_salon) con número de filas igual al número de eventos y con dos columnas. Para cada evento, en la primera columna se ubica la hora en la que se programa y en la segunda columna se almacena el salón donde se realiza.

Se puede formar la matriz horario_eventos con número de filas igual al número de eventos y con 45 columnas que corresponde al total de horas. La matriz horario_eventos en la posición (i,j) tiene un ‘1’ si el evento ‘i’ se programa a la hora ‘j’ o tiene un ‘0’ en caso contrario.

El horario de los estudiantes se puede generar a partir de las matrices evento_hora_salon y eventos_estudiantes, con lo que puede conocer la hora y el salón programados para cada estudiante según los eventos que tenga matriculados.

Para facilitar la comprobación de las restricciones blandas se forma la matriz horario_estudiante con número de filas igual al número de estudiantes y con 45 columnas que corresponde al total de horas. La matriz horario_estudiante en la posición (i,j) tiene un ‘1’ si el estudiante ‘i’ tiene programado un evento en la hora ‘j’ o

tiene un '0' en caso contrario. La matriz horario_estudiante se puede calcular como:

$$HA = EA' \cdot ET$$

*donde* :

*HA : matriz horario de los estudiantes*

*ET : matriz horario de los eventos*

*EA : matriz evento estudiante*

A continuación se presenta el modelo matemático para el problema de programación óptima de horarios de clase.

*Variables de decisión :*

$$X_{est} : \begin{cases} 1 & \text{si el evento 'e' se programa en el salón 's'} \\ & \text{en el bloque de tiempo 't'} \\ 0 & \text{si el evento 'e' no se programa en el salón 's'} \\ & \text{en el bloque de tiempo 't'} \end{cases}$$

$$\text{minimizar } z = hf + hu + hc$$

*s.a.*

$$\sum_{s \in N_S} \sum_{t \in N_T} X_{est} = 1 \qquad e \in N_E \qquad (1)$$

$$\sum_{e \in N_E} X_{est} \le 1 \qquad s \in N_S, t \in N_T \qquad (2)$$

$$X_{est} \left[ ES(e,s) - X_{est} \right] \ge 0 \qquad e \in N_E, s \in N_S, t \in N_T \qquad (3)$$

$$HA(t,a) \le 1 \qquad t \in N_T, a \in N_A \qquad (4)$$

$$hf = \sum_{t \in H} \sum_{a \in N_A} HA(a,t) \qquad (5)$$

$$hu = \sum_{a \in N_A} \sum_{d=1}^{5} e^{-\left\{1 - \sum_{t=H(d)-s}^{H(d)} HA(a,t)\right\}^2} \qquad (6)$$

$$hc = \sum_{a \in N_A} \sum_{d=1}^{5} \sum_{k=1}^{7} \prod_{t=H(d)+k}^{H(d)+k+2} HA(a,t) \qquad (7)$$

$$ET(e,t) = \sum_{s \in N_S} X_{est} \qquad e \in N_E, t \in N_T \qquad (8)$$

$$HA = EA' \cdot ET \qquad (9)$$

*donde* :

$N_A$ *: conjunto de estudiantes*

$N_E$ *: conjunto de eventos*

$N_S$ *: conjunto de salones*

$N_T$ *: bloques de tiempo*

*ES : matriz evento – salón*

*ET : matriz horario de los eventos*

*HA : matriz horario de los estudiantes*

*EA : matriz evento estudiante*

$H = [9, 18, 27, 36, 45]$

La restricción (1) asegura que cada evento se programa una sola vez. La restricción (2) representa la condición de que en un salón, en un bloque de tiempo, no se puede programar más de un evento. La restricción (3) permite garantizar que el salón asignado cumple con la capacidad y las características requeridas por el evento correspondiente. La restricción (4) previene que se presenten cruces de horarios para los estudiantes; esta restricción usa la matriz *HA*, que contiene los horarios de los estudiantes, construida según las ecuaciones (8) y (9).

La ecuación (5) define la variable *hf* que representa el número de eventos programados al final de cada día para todos los estudiantes. La ecuación (6) define la variable *hu* que representa el número de eventos únicos en cada día para todos los estudiantes. La ecuación (7) define la variable *hc* que representa los casos en que se programan más de dos eventos consecutivos en un día para todos los estudiantes.

## 4. COLONIA DE HORMIGAS

El algoritmo por colonia de hormigas es una metaheurística capaz de encontrar soluciones de buena calidad a problemas de optimización altamente complejos. Se basa en la habilidad que poseen las hormigas naturales para encontrar el alimento a partir de individuos relativamente simples pero con una estructura social altamente eficiente. La base en ambos sistemas (natural y artificial) es la comunicación indirecta entre todos los individuos a partir de rastros de feromonas.

El algoritmo construye una alternativa de solución agregando paso a paso un elemento $i$ de entre un conjunto alternativas $N_i^k$ conocido como vecindad del elemento $k$. Cada elemento tiene consigo una probabilidad $p_i$ de ser elegido dada por:

$$p_i = \frac{\tau_i^{\alpha} \cdot \eta_i^{\beta}}{\sum_{l \in N^k_i} \tau_l^{\alpha} \cdot \eta_l^{\beta}} \qquad (10)$$

Donde $\tau_i$ es una cantidad numérica almacenada en una matriz de feromonas y define el grado de participación del elemento $i$ en alternativas de solución de buena calidad encontradas con anterioridad. El factor $\eta_i$ es un valor dado por un análisis de sensibilidad del sistema, y define el impacto que produce el elemento $i$ al ser adicionado a la alternativa que se construye. Los parámetros de inicialización $\alpha$ y $\beta$ definen el grado de importancia de la información utilizada. Si $\alpha > \beta$ se da mayor importancia a la información proveniente de la matriz de feromonas, y en caso contrario prima la información heurística.

El proceso para construir una alternativa de solución se repite hasta completar varios individuos. Cada uno de ellos produce un incremento o un decremento de feromona sobre los elementos que los forman en forma proporcional a la calidad de la solución. Con base a esta información adicional se genera una nueva población con mejores características.

Con el fin de evitar el estancamiento del algoritmo en alternativas de baja calidad, el sistema colonia de hormigas efectúa un proceso de evaporación en dos etapas distintas. La primera etapa, conocida como evaporación local se realiza durante la construcción de cada solución, aplicando la ecuación (11) al elemento recientemente adicionado a la alternativa en

construcción. La segunda etapa, conocida como evaporación global se ejecuta luego de realizar los depósitos de feromona, aplicando a cada elemento de la alternativa de solución la ecuación (12). En ambos casos, $\varepsilon$ y $\rho$ son parámetros de calibración y están en el intervalo [0,1].

$$\tau_i = \tau_i \cdot (1-\varepsilon) \quad (11)$$

$$\tau_i = \tau_i \cdot (1-\rho) \quad (12)$$

*Codificación del Problema*

*a. Matrices de Feromonas*

Para la formación de las alternativas de solución se recurre a dos tipos matrices de feromonas. La primera matriz es mostrada en la figura 1 y relaciona la deseabilidad aprendida de seleccionar el salón *i* para dictar el evento j, por tanto es de dimensiones *SxE* (donde *S* es el número total de salones y *E* la cantidad de eventos del problema).

$$[\tau_{salones}] = \begin{bmatrix} \tau_{11} & \tau_{12} & \dots & \tau_{1j} & \dots & \tau_{1E} \\ \tau_{21} & \tau_{22} & \dots & \tau_{2j} & \dots & \tau_{2E} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \tau_{i1} & \tau_{i2} & \dots & \tau_{ij} & \dots & \tau_{iE} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \tau_{s1} & \tau_{s2} & \dots & \tau_{sj} & \dots & \tau_{sE} \end{bmatrix}$$

Figura 1. Matriz de feromonas Evento_Salón para la selección de los salones

La segunda matriz, mostrada en la figura (2), relaciona la deseabilidad aprendida de seleccionar el bloque de tiempo *i* para dictar el evento *j*, por tanto es de dimensiones BT*xE* (donde BT es el número total de bloques de tiempo).

$$[\tau_{BT}] = \begin{bmatrix} \tau_{11} & \tau_{12} & \dots & \tau_{1j} & \dots & \tau_{1E} \\ \tau_{21} & \tau_{22} & \dots & \tau_{2j} & \dots & \tau_{2E} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \tau_{i1} & \tau_{i2} & \dots & \tau_{ij} & \dots & \tau_{iE} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \tau_{s1} & \tau_{s2} & \dots & \tau_{sj} & \dots & \tau_{sE} \end{bmatrix}$$

Figura 2. Matriz de feromonas Evento_Tiempo para la selección de los bloques de tiempo.

*b. Primera fase: Asignación de Salones.*

El orden en el que se da la asignación depende del número de salones donde es factible que un evento se desarrolle, dando prioridad a los eventos con un menor número de opciones factibles. La probabilidad de escoger el salón *i* para el evento *j* se calcula por medio de la ecuación (10), usando los elementos de la columna *j* de la matriz de feromonas Evento_Salón. La información heurística en este caso se calcula de acuerdo a la ecuación (13).

$$\eta_{ij} = f * S_a^{NES} \quad (13)$$

Donde $S_a$ es un valor en el intervalo [0.9,1] con el cual se decrementa la probabilidad de escoger el salón *i* por la asignación de otros eventos con anterioridad, mientras que la variable *NES* indica el Número de Eventos asignados al Salón *i* durante la alternativa de solución en construcción. La variable *f* toma el valor de uno si el salón *i* es apto para el evento *j*, y cero en caso contrario.

Los eventos asignados a cada salón son almacenados en una matriz de *EventosxSalon* y utilizados durante la segunda fase. Con el fin de prevenir cruces de salones, la cantidad de asignaciones hechas a cada salón debe ser inferior al número de bloques de tiempo.

*c. Segunda Fase: Asignación de los bloques de tiempo*

Para completar la alternativa de solución, se deben asignar los bloques de tiempo en los cuales se dictarán los eventos que son impartidos en un mismo salón. De esta manera se previene el cruce de salones en la formación de la solución. La probabilidad de que un evento *j* se dicte en el bloque de tiempo *i* se calcula por medio de la ecuación (*), usando los elementos de la columna *j* de la matriz de feromonas Evento_Tiempo. La información heurística se determina por la ecuación:

$$\eta_{ij} = (1 - Asig) * Cr^{NC} * Ec^{NCont} \quad (14)$$

Donde *Asig* es un vector en el que cada posición caracteriza los bloques de tiempo, y puede tomar el valor de uno si el salón ha sido programado en el bloque de tiempo *i,* en caso contrario tiene un valor de cero. *Cr* y *Ec* son factores que reducen la probabilidad de aceptar el bloque de tiempo *i* en caso de que se produzca un cruce de horario en el primer caso, o un evento continuo en el segundo caso. *NC* es el número de estudiantes a los cuales se les asignó previamente el asistir a otro evento durante el bloque de tiempo *i*. *NCont* es el número de eventos continuos que se producen en los horarios de los estudiantes matriculados al evento *j* si es seleccionado el bloque de tiempo *i*.

A menor valor de *Cr*, y *Ec* cercano a uno, se logra rápidamente una solución factible, pero con un amplio número de restricciones blandas insatisfechas. Asimismo, con un valor de *Cr* cercano a uno y *Ec* tendiendo a cero, se logran soluciones con bajas restricciones blandas pero infactibles. Por tal motivo se deben ajustar cuidadosamente estos parámetros dependiendo de las características y el tamaño del problema con el fin de obtener respuestas de buena calidad en tiempos computacionalmente eficientes.

*d. Evaporación y Depósito de Rastros de Feromona*
Cada vez que a un evento se le asigna un salón o un bloque de tiempo durante la construcción de la alternativa de solución, se disminuye levemente la cantidad de feromona que caracteriza al elemento adicionado. De esta manera se previene el estancamiento sobre soluciones de baja calidad y se amplía la diversidad de soluciones construidas en cada iteración. Este procedimiento corresponde a una fase de evaporación local, en la que se aplica la ecuación (11), donde el factor de evaporación Є se calibró en 1%.

La evaporación global se aplica al final de cada iteración y afecta todos los elementos de todas las alternativas, excepto a la que posea las mejores características. Para efectos del problema, se define que una alternativa es mejor que otra si:

- Posee un menor grado de infactibilidad, o
- Posee la misma infactibilidad, pero un menor número de restricciones blandas insatisfechas.

El factor de evaporación global se calcula por medio de la siguiente ecuación:

$$\rho = 0.99^{(a+b)} \qquad (15)$$

Donde *a* es la diferencia entre el número de restricciones duras violadas por la alternativa actual contra las restricciones violadas por la mejor alternativa encontrada durante todo el algoritmo. En forma análoga es calculado *b* pero se tiene en cuenta sólo las restricciones blandas.

La mejor solución construida en cada iteración es comparada con la mejor encontrada durante todo el algoritmo. Si la mejora es reemplazada y se mantiene como incumbente. Ambas alternativas están habilitadas para depositar feromona en los elementos que las forman, de acuerdo a la ecuación (16).

$$\Delta = t \cdot 0.99^{(Rduras+Rblandas)} \qquad (16)$$

Donde $\Delta$ es el incremento en cada elemento de las matrices de feromonas que pertenecen a las alternativas de solución habilitadas para los depósitos. $R_{duras}$ es el número de restricciones duras que origina la alternativa y $R_{blandas}$ el número de restricciones blandas. El parámetro de calibración t equivale a un 5% del valor de la feromona inicial en caso de evaluar la mejor alternativa de la iteración, y de un 10% de la feromona inicial en caso de evaluar la incumbente.

## 5. PRUEBAS Y RESULTADOS

Se desarrolló un programa computacional usando lenguaje Delphi que implementa la metodología para la solución del problema de programación óptima de horarios de clase usando Colina de Hormigas. El programa fue validado a través de tres casos de prueba cuyos datos del sistema se encuentran disponibles en la página *International Timetabling Competition* [13].

En cada uno de los casos de prueba considerados se requiere programar 400 eventos de una hora de duración en un espacio físico formado por 10 salones. Se dispone de cinco días hábiles en los cuales la institución labora nueve horas, por lo que se cuenta con un total de 45 bloques de tiempo. Existen 200 estudiantes matriculados, y se tienen un conjunto de 10 características diferentes que describen las condiciones específicas de cada aula, así como las necesidades requeridas por cada evento, lo que se pretende es que el evento sea programado en un salón que cumpla con las características necesarias para que se ejecute el evento de interés.

El algoritmo fue calibrado para efectuar una búsqueda del óptimo de mejor calidad en un número máximo de 300 iteraciones, donde la población se formaba por 10 hormigas o alternativas de solución.

Al inicio del algoritmo se asignaron los siguientes valores a los parámetros de la ecuación (14): Cr=0.9 y Ec=0.2. De esta manera la búsqueda se centra en reducir las restricciones blandas violadas, aunque la infactibilidad inicial generada sea alta. El valor de Cr disminuye gradualmente con el fin de forzar el algoritmo hacia soluciones factibles y se logre una función objetivo de buena calidad.

En cada caso de prueba se lograron alternativas de solución factibles, es decir que todos los eventos fueron asignados en salones adecuados sin que se presenten cruces de salones o de horarios en los estudiantes. Sinembargo se debe mencionar que se violaron un cierto número de restricciones blandas, con lo cual la función objetivo es diferente de cero.

En el primer caso de prueba, el valor de la función objetivo es de 386, debido a la siguiente secuencia de restricciones blandas insatisfechas en los horarios de los estudiantes: 209 eventos continuos, 147 eventos en la hora 9 del día y 30 eventos de asistencia única por día.
En el segundo caso de prueba, el valor de la función objetivo es de 365, que corresponde a la siguiente secuencia de restricciones blandas insatisfechas en los horarios de los estudiantes: 188 eventos continuos, 147 eventos en la hora 9 del día y 30 eventos de asistencia única por día.

Por último, en el tercer caso de prueba, el valor de la función objetivo es de 434, ya que se presentan 207 eventos continuos, 206 eventos en la hora 9 del día y 21 eventos de asistencia única por día en los horarios de los estudiantes.

| | Caso de Prueba | | |
|---|---|---|---|
| **# estudiantes con** | **1** | **2** | **3** |
| eventos continuos | 209 | 188 | 207 |
| eventos finales | 147 | 121 | 206 |
| eventos únicos | 30 | 30 | 21 |
| Función Objetivo | 386 | 339 | 434 |

Tabla 1. Resultados encontrados para cada caso de prueba.

## 6. CONCLUSIONES Y RECOMENDACIONES

Durante el desarrollo de la aplicación y puesta a prueba del programa computacional, se observó que resulta conveniente asignar inicialmente los salones a los eventos y posteriormente los bloques de tiempo. Adicionalmente, es crítica la asignación en primera instancia de los eventos que cuentan con un reducido conjunto de salones apropiados, con lo cual se reduce drásticamente el número de infactibilidades generadas.

Se ha mostrado que la metaheurística de optimización por Colonia de Hormigas resuelve el problema de asignación óptima de salones, encontrando una solución factible y de buena calidad, considerando el tamaño y complejidad de las instancias asumidas.

Estudiando el comportamiento del aplicativo con los casos de prueba se observó que una vez alcanzada la solución factible, es difícil que esta sea mejorada, lo cual denota una convergencia acelerada del algoritmo. Lo anterior se debe a la definición adoptada para comparar dos alternativas distintas y a que siempre se almacena como incumbente la que produzca la menor infactibilidad. De esta manera se renuncia al interés sobre cualquier solución con bajas restricciones blandas y un grado de infactibilidad reducido, que puede resultar tan importante como una solución factible. Se pueden almacenar varias incumbentes, siguiendo el criterio de que cada una de ellas minimice una de las restricciones blandas, con lo que se consigue mayor diversidad de las configuraciones.

Como trabajos futuros se propone implementar una técnica hibrida entre Colonia de Hormigas y Simulated Annealing de forma que con este último se obtengan soluciones iniciales diferentes y de buena calidad, para que con Colonia de Hormigas se realice la exploración del vecindario. Además se deben considerar aspectos adicionales como el desplazamiento de los estudiantes de un salón a otro, el que se tengan eventos que duren más de una hora y materias que deban distribuirse en diferentes días de la semana. También se puede considerar el problema de asignación de carga docente, resolviéndolo simultáneamente con el de programación de clases.

## 7. BIBLIOGRAFÍA

[1] Wren A. Scheduling, timetabling and rostering – A special relationship In: Burke and Ross, pp. 46–75, 1996.
[2] Burke, E.K., Elliman, D.G., Ford, P., Weare, R.F. Examination timetabling in British Universities – A survey. In: Burke and Ross, pp. 76–92, 1996.
[3] Carter, M.W., Laporte, G., Recent developments in practical examination timetabling. In: Burke and Ross, pp. 3–21, 1996.
[4] Carter, M.W., Laporte, G., Recent developments in practical course timetabling. In: Burke and Carter, pp. 3–19, 1996.
[5] Werra, An introduction to timetabling. European Journal of Operational Research, 19, pp. 151–162, 1985.
[6] White, GM and Chan P.W. Towards the construction of optimal examination timetables. INFOR 17, pp. 219–229, 1979.
[7] Fisher, J.G., Shier, D.R., A heuristic procedure for large-scale examination scheduling problems. Technical Report 417, Department of Mathematical Sciences, Clemson University.
[8] White G.M. Constrained satisfaction, not so constrained satisfaction and the timetabling problem. In: A Plenary Talk in the Proceedings of the 3rd International Conference on the Practice and Theory of Automated Timetabling, University of Applied Sciences, Konstanz, August 16–18, 2000, pp. 32–47.
[9] Burke, E.K., Newall, J.P., Weare, R.F., 1996b. A memetic algorithm for University exam timetabling. In: Burke and Ross, pp. 241–250, 1996.
[10] Granada M., Toro E. M., Franco J. F., Programación óptima de horarios de clase usando un algoritmo memético. Sicentia et technica Año XII no. 30 Mayo de 2006.
[11] Burke, McCollum, Meisels, A graph-based hyper-heuristic for educational timetabling problems. European Journal of Operational Research 176, 2007.
[12] Domingo M., Stützle T. Ant Colony Optimization, A Bradford Bokk, Massachusetts Institute of Technology, 2004.
[13] International Timetabling Competition. Disponible en: http://www.idsia.ch/Files/ttcomp2002/