Revista Cubana de Ciencias Informáticas
Vol. 6. No. 4, Octubre –Diciembre, 2012
ISSN: 2227-1899 | RNPS: 2301
Pág. 37-43
http://rcci.uci.cu

Tipo de artículo: Artículo original
Temática: Técnicas de programación
Recibido: 29/11/2012 | Aceptado: 18/12/2012

Aplicación de algoritmos genéticos en la generación automática de horarios docentes en la Facultad Regional de Granma

Application of genetic algorithms in the automatic generation of class schedules at the Regional School in Granma

Karel Rodríguez Varona

Vicedecanato de Formación. Facultad Regional Granma de la Universidad de las Ciencias Informáticas, Ave Camilo Cienfuegos s/n, Manzanillo, Granma, Cuba

karel@grm.uci.cu

Resumen

En este trabajo se propone una solución basada en Algoritmos Genéticos, al problema de la generación de horarios que presenta la Facultad Regional Granma de la Universidad de las Ciencias Informáticas. Se realiza el modelado de la solución y la implementación del algoritmo propuesto en el lenguaje de programación Java acordes a las necesidades de la Institución. Entre sus particularidades se encuentra la presencia de restricciones que deben ser satisfechas. Finalmente, el algoritmo se somete a una prueba, con un juego de datos real, con el objetivo de observar su comportamiento y verificar su factibilidad.

Palabras clave: algoritmos genéticos, horarios docentes, meta-heurísticas.

Abstract

This paper proposes a solution, based on genetic algorithms, to the problem of generating schedules in the Granma Regional School of the University of Informatics Sciences. Modeling of the solution is performed and the implementation of the proposed algorithm in the Java programming language. One of the particularities of this problem is the presence of constraints that must be met. The tricky thing when it comes to generalize, is that each educational center is different in terms of available resources and therefore about the restrictions that may occur (also have "general" restrictions that are applicable to any institution). Finally, the algorithm undergo a test, with an actual data set, in order to observe its behavior and to verify its feasibility.

Keywords: *educational timetabling, genetic algorithm, meta-heuristics.*

Introducción

Cuando se habla de planificación, en el ámbito de las ciencias de la computación, situaciones muy complejas vienen a la mente. La generación automática de horarios docentes es un subproblema dentro de ese espectro de optimizaciones. Diversas han sido las técnicas empleadas a lo largo de la historia para acometer los problemas de *educational timetabling*, como se le conoce en inglés. Las meta-heurísticas sobresalen entre ellas.

Uno de los procesos más importantes en la organización de un curso académico, en una institución docente, es la conformación de los horarios de actividades lectivas para cada grupo perteneciente a la institución en cuestión.

Confeccionar un horario que se ajuste a los intereses de profesores y alumnos, que utilice de la mejor manera posible los recursos disponibles, es una parte primordial en la planificación de cualquier nivel de enseñanza. Se trata de un problema común que se aborda, generalmente, de manera manual o con soluciones empíricas ajustables a las necesidades de la institución que lo presenta, dadas las diferencias entre las planificaciones de los distintos tipos y niveles educativos.

El problema de la generación de horarios es uno de los clásicos de las ciencias de la computación, específicamente de la Inteligencia Artificial, referido como *Timetabling* definido como: "la asignación, sujeta a restricciones, de un grupo de recursos a objetos ubicados en tiempo y espacio, de tal manera que se satisfagan un conjunto de objetivos deseados" (Wren, 1996) y aparece en diversos ámbitos incluyendo: la planificación de transporte, programación de eventos deportivos, horarios laborales (Qu, 2002).

Existen varios abordajes que utilizan diversas técnicas y, usualmente, se adaptan o condicionan a las características de la entidad que se disponga a obtener una solución. Un caso particular es la planificación de horarios docentes (*Educational Timetabling*), que a su vez se puede dividir en planificación de actividades lectivas y planificación de exámenes (Willemen, 2002). El primero de los casos será el que se trate en este trabajo.

En cada inicio de período se presenta la necesidad de organizar y distribuir los horarios de clases de profesores y alumnos. Sin embargo, se presentan condiciones particulares originadas por la dinámica de la institución, cambios en los programas de las asignaturas y algunos otros factores como la disponibilidad de profesores y locales, que complican su exitosa planificación.

En la Facultad Regional Granma (FRG) de la Universidad de las Ciencias Informáticas (UCI), los horarios docentes de cada semestre se realizan de forma manual. Para esto se emplea mucho tiempo (supera las dos horas por cada semana que se desee planificar). Esta tarea es realizada por el Vicedecano de Formación, quien debe tener en cuenta muchas restricciones para terminar con una propuesta que se ajuste, a las exigencias del centro y de los balances de carga. Frecuentemente, el horario presenta inconsistencias, introducidas por el error humano, que imposibilitan su cumplimiento y hacen obligatoria la redistribución (volver a planificar) de actividades docentes.

Se requiere que, al inicio de cada semestre, sean publicados los horarios de cada una de las semanas, que no son iguales entre ellos debido a las diferencias en cuanto a tipologías de clases y frecuencias de cada asignatura a impartir, en la mayoría de los casos. El cambio frecuente en las actividades de la facultad hace que a menudo sea necesario alterar el horario de alguna semana en particular y esto se traduce en otra inversión de tiempo para adecuar la nueva distribución.

Se hace necesario minimizar las inconsistencias en los horarios docentes y el tiempo empleado en su generación en la Facultad Regional Granma de la UCI

Luego de haber realizado un estudio del estado del arte de la temática expuesta, se valoraron diferentes meta-heurísticas para solucionar el problema de la generación automática de horarios docentes. Se consideraron algunas herramientas propuestas por diferentes autores.

El objetivo del trabajo es implementar algoritmos que, aplicando técnicas de inteligencia artificial, puedan ser utilizados para generar automáticamente los horarios docentes de la Facultad Regional Granma de la Universidad de las Ciencias Informáticas.

Metodología computacional

Con el ánimo de ayudar en la consecución del objetivo general planteado, se definieron siguientes tareas:

- Describir el problema de la generación de horarios docentes.

- Caracterizar algunas de las principales meta-heurísticas que se utilizan para encontrar soluciones a problemas de *educational timetabling.*
- Diseñar una propuesta de solución al problema de la generación de horarios apoyada en una (o en una combinación) de las meta-heurísticas caracterizadas.
- Implementar el algoritmo propuesto que sea capaz de generar, de manera automática, los horarios docentes de la FRG, cumpliendo un conjunto de restricciones preestablecidas.
- Evaluar las ventajas de la propuesta sobre la confección manual en cuanto al tiempo empleado en la obtención y las inconsistencias que puedan presentar las soluciones.

Entre los **métodos científicos** utilizados en este trabajo se destacan:

- Métodos generales.

El método hipotético-deductivo para elaborar la idea a defender y trazar estrategias para cumplirla; el método sistémico para el desarrollo de la aplicación y lograr que los elementos que formen parte de la misma sean un todo que funcione de manera armónica; el método histórico-lógico y el dialéctico para el estudio crítico de los trabajos anteriores, y para utilizar estos como punto de referencia y comparación de los resultados.

- Métodos lógicos.

El método analítico-sintético al descomponer el problema de investigación en elementos por separado, para luego sintetizarlos en la solución de la propuesta; el método inducción-deducción como vía de la constatación teórica durante el desarrollo de la investigación; el método de modelado para el desarrollo de los algoritmos.

- Métodos empíricos.

El método coloquial para la presentación y discusión de los resultados; el método experimental para comprobar la utilidad de los resultados.

Herramientas utilizadas

Para el modelado y la implementación de los algoritmos se utilizaron las herramientas y lenguajes que se relacionan en este epígrafe.

Como lenguaje de modelado se utilizó el que quizás sea el más utilizado desde hace muchos años, UML.

El lenguaje de programación seleccionado fue Java en su versión 1.6.0_32. Su carácter multi-plataforma desempeñó un papel fundamental en la elección. La máquina virtual empleada fue la Java HotSpot Client VM, de Oracle.

Como entorno integrado de desarrollo (EID) se utilizó Netbeans en su versión 6.9.

Los modelos de UML se diseñaron utilizando, precisamente, el módulo que para este lenguaje posee Netbeans.

Las imágenes que se aparecen en el informe fueron creadas con la versión 0.48.2 del editor de gráficos vectoriales Inkscape.

Todo el trabajo se realizó sobre el sistema operativo Linux Mint 11, corriendo en una computadora personal con 1Gb de RAM y un procesador Intel Core Duo a 1.86 GHz. Este sistema es derivado de Ubuntu 11.04, una de las distribuciones GNU Linux más populares en la actualidad.

Modelado de la solución

Las clases que intervienen en esta solución se agruparon en paquetes de acuerdo a su finalidad. Se pueden identificar cuatro paquetes fundamentales:

1. *genetic* : Clases relacionadas directamente con el algoritmo genético implementado.
2. *model* : Clases utilizadas para modelar los recursos que intervienen en la planificación de horarios.

Revista Cubana de Ciencias Informáticas
Vol. 6. No. 4, Octubre –Diciembre, 2012
ISSN: 2227-1899 | RNPS: 2301
Pág. 37-43
http://rcci.uci.cu

3. *constraints*: Clases utilizadas para modelar las restricciones que pueden incluirse en el horario.
4. *aux* : Clases auxiliares que se utilizan en funciones muy específicas.

El algoritmo

Se utilizó una variante de la propuesta realizada por (Koizumi, Yamamori, & Yoshihara, 2002). La propuesta en este trabajo es semejante en cuanto a varios aspectos internos, aunque también difiere en varios.

Composición del cromosoma

Toda vez que los cromosomas son soluciones candidatas cuando se habla de algoritmos genéticos, en esta propuesta los cromosomas se constituyen de secuencias de actividades ubicadas, cada una de ellas, en un espacio de tiempo. En un cromosoma válido deben incluirse todas las actividades a ubicar en el horario.

Cada cromosoma se divide en segmentos (habrá tantos segmentos como grupos docentes). Los segmentos se componen, a su vez, por identificadores de actividades.

La cantidad máxima de actividades permitidas (CMA) en cada segmento viene dada por la diferencia entre el producto del número de días hábiles (d) y cantidad de horas diarias de actividades (h) y la cantidad de espacios de tiempo deshabilitados (CED) para el grupo correspondiente.

$$CMA = d \times h - CED$$

Operadores genéticos

Cada uno de los operadores implementados funcionan a nivel de segmentos. El cruce entre dos cromosomas consiste en cruzar cada uno de los segmentos de uno de ellos con su homólogo correspondiente en el otro.

Para efectuar el cruce entre dos cromosomas se utilizó una variante del cruce en un punto (one-point crossover) (Sastry, Goldberg, & Kendall, 2005).

El **cruce** se realiza a nivel de segmentos, cada segmento de un progenitor se cruza con su homólogo correspondiente en el otro progenitor.

Como punto de cruce se tomó la casilla ubicada en la mitad del segmento. De esta forma cada cruce entre dos progenitores origina dos descendientes:

1. el primero con la primera mitad del progenitor p1 y la segunda mitad del progenitor p2,
2. el segundo con la primera mitad del progenitor p2 y la segunda mitad del progenitor p1.

Este tipo de cruce (para este problema en particular) puede generar lo que se denomina cromosoma letal. Un cromosoma letal es aquel en el que se puede dar una de las siguientes situaciones o ambas.

1. Genes duplicados. Aparecen más de una vez en el cromosoma.
2. Genes en falta. No aparecen en el cromosoma.

Para palear esta situación, se lleva a cabo un proceso de “eliminación de cromosomas letales” cada vez que se realiza un cruce. Contrario a lo que podría pensarse, estos cromosomas no son eliminados como tal, sino que son “reparados” mediante la localización de sus genes duplicados y sustitución por los que están en falta. Al terminar este proceso, los cromosomas contienen todos los genes que les corresponden y ninguno está duplicado.

La **mutación** se realiza según una probabilidad que puede ser variada de una corrida a otra. Al terminar un cruce, a los descendientes se les aplica este operador que, en dependencia de la probabilidad de mutación, selecciona dos genes de manera aleatoria e intercambia sus respectivos valores.

Este proceso se realiza con el objetivo de buscar variedad en las posibles soluciones. Cabe resaltar que la probabilidad

Revista Cubana de Ciencias Informáticas
Vol. 6. No. 4, Octubre –Diciembre, 2012
ISSN: 2227-1899 | RNPS: 2301
Pág. 37-43
http://rcci.uci.cu

de mutación no debe ser muy alta, de otro modo le estaría restando protagonismo al cruce (operador más importante en este algoritmo).

La **selección** de los individuos para cruzar en cada una de las generaciones se realiza de forma totalmente aleatoria. Esta política rompe con el elitismo a la hora del cruce, dando así una mayor diversidad en cuanto a los descendientes que se generan en cada iteración.

Los individuos que formarán parte de la nueva población se seleccionan de acuerdo a su aptitud. Incluso los progenitores compiten por mantenerse en la siguiente población. De esta forma, con una cantidad de individuos fija, se lleva a cabo el proceso de reemplazo para obtener la nueva población.

Evaluación de aptitud

Para evaluar la aptitud de cada cromosoma se tiene una función que recibe dicho cromosoma como parámetro y chequea si este cumple con cada una de las restricciones presentes. Si una restricción no se ve cumplida en el cromosoma, le aplica una penalización a dicho individuo.

$$Aptitud_x = \frac{1}{\sum_{i=1}^{i=n} P(R_i, x)}$$

x es el cromosoma que se está evaluando,

n es el número de restricciones presentes,

R_i es la restricción que ocupa la posición i y

P es la función mediante la cual R_i penaliza a *x*.

Paralelamente a esta aptitud que se le calcula al cromosoma, existen algunas restricciones de obligatorio cumplimiento (delo inglés, *Hard Constraints*) las cuales, además de sumar su penalización (en caso de verse incumplidas), ponen en falso un atributo de los cromosomas indicando que no son cromosomas factibles.

Condiciones de parada

El algoritmo debe detenerse en cualquiera de las siguientes situaciones:

1. Se ha generado un individuo con aptitud 1, garantizando que se cumplen todas las restricciones.
2. Se ha alcanzado el número máximo de generaciones. En este caso no se garantiza haber cumplido con todas las restricciones.

En cualquiera de los casos descritos anteriormente, la salida del algoritmo generador es la última población generada con los individuos ubicados en orden descendente de su aptitud.

Resultados y discusión

Luego de haber concluido con la implementación del algoritmo, este se sometió a una prueba de funcionamiento con un juego de datos real. El experimento se describe a continuación.

Se tomaron los datos (actividades) de la semana número uno del segundo semestre del curso 2011- 2012 en la FRG. La inclusión de estas actividades (68 en total) hizo necesaria la incorporación de un total de 41 restricciones que las relacionaban (asegurando la presencia de al menos una de cada tipo soportado).

Se planificó una semana de actividades con cinco días hábiles y seis espacios de tiempo en cada día hábil, lo que deja un máximo de 30 espacios de tiempo para cada recurso.

Se realizaron 10 corridas, para observar el comportamiento del algoritmo en cada una, con dos configuraciones

diferentes de los parámetros genéticos.

La primera configuración con valores: 500 individuos, 200 cruzamientos por generación, 1000 generaciones y una probabilidad de mutación de 0,5 arrojó un promedio de efectividad del 100 % con 236,6 generaciones y 44 segundos.

La segunda configuración: 1000 individuos, 500 cruzamientos por generación, 1000 generaciones y probabilidad de mutación 0,6 tuvo 100 % de efectividad con 116,7 generaciones y 55,1 segundos como promedio.

La configuración 1 resultó ligeramente menos positiva que la configuración 2 en cuanto al número de generaciones necesarias, como promedio, para obtener la óptima calidad (236.6 vs. 116.7). En cuanto al tiempo empleado en obtener la solución, resultó mejor la configuración 1 (44 vs 55.1).

Por su promedio de efectividad y menor consumo de tiempo, la configuración 1 fue seleccionada para establecerse como configuración predefinida.

Si se observan los tiempos empleados en obtener una solución, utilizando cualquiera de las dos configuraciones, se puede apreciar que son pequeños comparados con los tiempos necesarios para generar un horario de forma manual, según el reporte del autor.

Lo hasta aquí expuesto corrobora la idea que se defiende en esta investigación. La utilización de un generador automático de horarios contribuirá a utilizar menos tiempo en esta tarea y reducirá el número de inconsistencias en los horarios generados.

El comportamiento del algoritmo fue adecuado en relación con el tiempo empleado en encontrar una solución al problema planteado. La calidad de las soluciones encontradas en todas las corridas realizadas fue buena en el sentido de cumplir todas las restricciones introducidas.

Conclusiones

Con el desarrollo de este trabajo científico, se pudo constatar la actualidad del tema y la diversidad de técnicas de computación que se utilizan en este tipo de problemáticas. Las características, recursos y condiciones en general de cada entorno, hacen de este un problema para el cual, encontrar una solución generalizable, es muy difícil.

Se realizó una presentación y caracterización del problema de generación de horarios docentes. Se hizo énfasis en el interés que ha despertado en la comunidad científica internacional y se relacionaron las diferentes clasificaciones del problema de acuerdo al objetivo final.

Se realizó una búsqueda en internet de la evolución de las técnicas meta-heurísticas aplicadas al problema de *educational timetabling* y se realizó una caracterización de cada una de las técnicas revisadas, valorando su grado de éxito en el campo. Como resultado de esa revisión, se eligió la técnica de Algoritmos Genéticos para el desarrollo de la solución propuesta.

Se diseñó e implementó una propuesta de solución, basada en Algoritmos Genéticos, con un conjunto de restricciones adecuadas a la realidad de la FRG. La solución propuesta establece las restricciones que, en la experiencia del autor, son más utilizadas en la institución para acomodar las actividades de una semana y respetar la disponibilidad de los diferentes recursos que intervienen en el proceso docente. El algoritmo, implementado en Java, utiliza un conjunto de clases, definidas al efecto, para modelar cada uno de los recursos y restricciones que deben ser incluidos en la planificación de una semana de actividades docentes, y acomodar cada actividad en un espacio de tiempo con la condición de que se cumpla la mayor cantidad de restricciones posibles.

Se llevó a cabo un experimento, consistente en una serie de corridas del algoritmo con diferentes configuraciones de los parámetros (probabilidad de mutación, número de generaciones, número de cruces por generación), para verificar su efectividad y eficacia, utilizando un juego de datos real (una planificación de actividades de una semana en la

FRG). El experimento arrojó resultados positivos en cuanto al tiempo empleado por el algoritmo para encontrar una solución factible al problema planteado y las soluciones encontradas cumplen, en su mayoría, con todas las restricciones introducidas.

En sentido general, se cumplieron los objetivos propuestos al inicio de esta investigación.

Referencias

- KOIZUMI, N., YAMAMORI, K., & YOSHIHARA, I. Genetic algorithm approach to university timetabling. Asia-Pacific Conference on Simulated Evolution and Learning (SEAL) 2002, p. 1–5. Singapur.
- QU, R. Case-based reasoning for course timetabling problems. Search. University of Nottingham. 2002.
- SASTRY, K., GOLDBERG, D., & KENDALL, G. Genetic Algorithms. In E. K. Burke & G. Kendall (Eds.), Search Methodologies. 2005, p. 97–125. Springer US. Disponible en: [http://dx.doi.org/10.1007/0-387-28356-0_4].
- WILLEMEN, R. School timetable construction. Algorithms and complexity. Technische Universiteity Eindhoven. 2002.
- WREN, A. Scheduling, timetabling and rostering - A special relationship? In E. Burke & P. Ross (Eds.), Practice and Theory of Automated Timetabling. Vol. 1153, 1996, p. 46–75. Springer Berlin / Heidelberg. Disponible en: [http://dx.doi.org/10.1007/3-540-61794-9_51].