Sistema automático para

Asignación de aulas y distribución de espacios

Pablo Cababie, Facundo Cancelo, Daniela López De Luise
Universidad de Palermo (TE. 5199 4520 - FAX 4963 1560),
ITLab, AIGROUP, aigroup@palermo.edu

Abstract—El problema de asignación de aulas y optimización de espacios ha sido desarrollado e investigado hasta hoy utilizando diversas herramientas y tecnologías. En este trabajo se propone encontrar una solución al problema de asignación de aulas, profesores y recursos como primer paso para diseñar un sistema que pueda ser adaptado a problemas similares. El enfoque aquí presentado se basa en algoritmos genéticos, cuyos resultados relevantes se presentan y evalúan. Dada la esencia del problema, la investigación se centra en un algoritmo MOEA (Multi Objective Evolucionary Algorithm) que optimice varios objetivos al mismo tiempo, que competirán para llegar al mejor resultado posible.

I. Introducción

El presente trabajo corresponde al proyecto llevado a cabo en el ITLab de la Universidad de Palermo, bajo el nombre Gdarim, para optimizar y mejorar el sistema de asignación de aulas. Este proceso suele realizarse manualmente, estimando y estipulando los recursos como mejor parecen ajustarse.
A simple vista parece la optimización de algunos pocos parámetros (cantidad de alumnos por clase, capacidad de las aulas y cantidad de recintos disponibles) pero requiere el modelado de relaciones complejas entre los mencionados parámetros. Entre las versiones de software, algunas soportan visualmente la tarea de asignar aulas manualmente; las que no suelen implementar inteligencia alguna, ni optimizan la distribución[2][4][5]. Existen soluciones que recurren a métodos matemáticos, como el simplex [8] y en otras se llega a utilizar inteligencia artificial.[9]
El resto de este trabajo se organiza como sigue: en la sección II se presentan algunos productos y alternativas vigentes en el mercado; en la sección III se explica la estructura y arquitectura general que tiene Gdarim, tanto a nivel de implementación y código como a nivel de su diseño; en la sección IV se compara Gdarim y todas las opciones que se ofrecen en el mercado. La sección V define las conclusiones y el trabajo propuesto para el futuro.

II. Productos vigentes

No parece existir aún un producto con administración y asignación de recursos y aulas totalmente automatizado. Aplicaciones como Matrícula y Classroom scheduler[5], ayudan a la visualización del esquema temporal y la distribución de espacios, pero carecen de inteligencia alguna en la asignación de las aulas y recursos. Visual Classroom Scheduler, [2] distribuye y acomoda las clases en los ámbitos disponibles. Matrícula[1], se limita a integrar la información académica en un sistema permitiendo la interacción entre diferentes áreas. El más prometedor sería Softaula [3], que provee una interfaz amigable para crear un esquema de clases y horarios, pero no asigna las aulas con inteligencia ni ayuda a mejorar la performance en la distribución. El open source Classroom Scheduler [5] permite una visualización de la distribución de los salones de clase. Otro open source, phpScheduleIt [4], no asigna

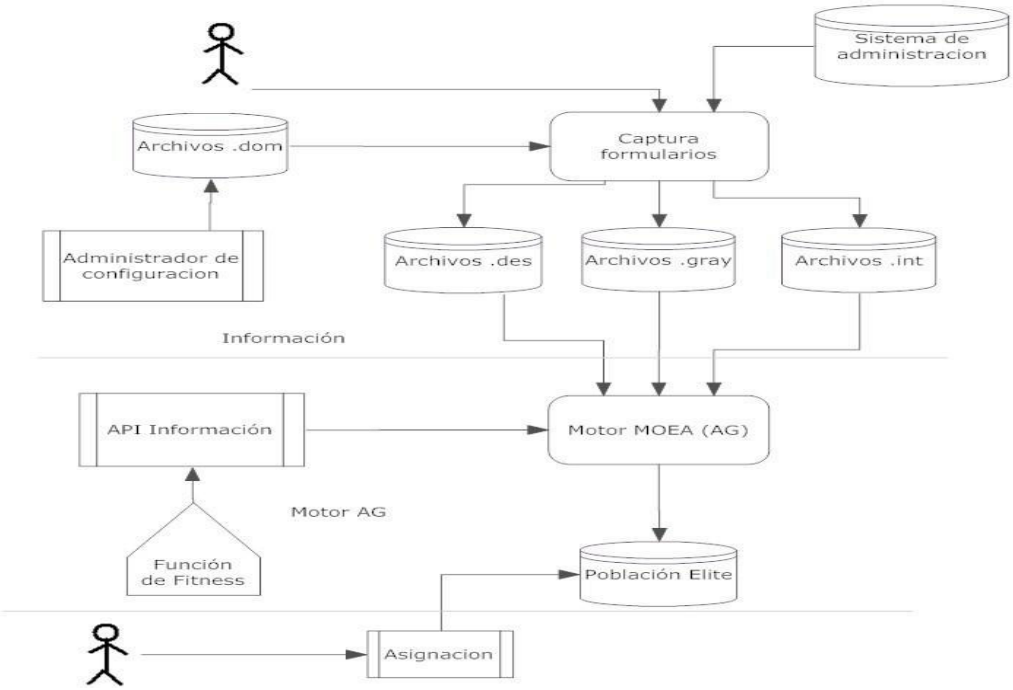
los recursos eficientemente pero gestiona los cambios de aula ante eventos especiales y ayuda a gestionar los cambios para que la repercusión de los mismos sea mínima dentro del esquema propuesto. Algunas universidades incorporaron la asignación manual a los sistemas informáticos. Como en la Universidad nacional del Comahue, con su sistema SIU Guaraní [6]. Dicho sistema realiza la gestión académica de alumnos, gestiona aulas, mesas de exámenes, jurados, etc.
Las alternativas mencionadas aportan un avance considerable en el proceso de asignación de aulas sin embargo, ninguna de estas consigue resolverlo por completo.

III. ESTRUCTURA DE GDARIM

El problema de asignación de aulas podría reducirse a una asignación de recursos, con una cantidad considerable de parámetros interdependientes. La propuesta de Gdarim es usar algoritmos genéticos [7], tecnología apropiada para este tipo de problema. Este fue desarrollado en Java, por su portabilidad y su característica de open source. El prototipo consta de tres módulos (ver Fig. 1):

- Información: contiene y administra la información del problema y sus características.
- Motor AG: corazón del algoritmo genético y la inteligencia de la solución.
- Asignación: registra y administra las posibles asignaciones de recursos que genera el algoritmo.

Figura 001. Arquitectura.



Se carga la información requerida en varios formularios. La misma es almacenada en archivos que luego son accedidos y "entendidos" por la aplicación. La información ingresada es codificada en genomas, que son evaluados, mutados y cruzados de acuerdo a las operaciones de AG definidas para el problema[7] y a funciones de aptitud que serán las encargadas de determinar que alternativa resuelve mejor el problema.
El proyecto cuenta con archivos de configuración, de dominio de valores para las variables en texto y en código gray, de descripción de las interrelaciones entre las variables (resultado de los formularios), los códigos e identificadores de cada entidad.

El proceso empieza definiendo el tipo de Problema Gdarim que consta de 21 variables binarias y una cantidad de restricciones que, en principio, dependen de los formularios y otras restricciones implícitas (por ejemplo, que las materias no se superpongan). Con ésto, se invoca el motor MOEA.
Se crea una población inicial que está formada por un conjunto finito de individuos para un día determinado. Cada uno es una asignación completa para todas las aulas ingresadas y los recursos correspondientes. En la definición del gen se especifica qué parámetros se deben evaluar para la representación del problema. Algunos de los parámetros son: nro. De aula, materia, profesor, turno, etc. Una vez definido el Gen, se genera la población inicial con n_0 individuos. Se eliminan las asignaciones incorrectas y se eligen los mejores individuos, resultando una población de trabajo de n_1 individuos.
En el MOEA propuesto se optimiza no sólo la asignación de aulas, sino además la asignación de los profesores a pesar de potencialmente ser objetivos contrapuestos [11][12]. Se aplica la función de fitness **f1** para la asignación de aulas y la función **f2** para la asignación de profesores. Ambos criterios deben ser optimizados, y con este algoritmo podrán competir en función de la incidencia sobre el beneficio y desventaja alcanzados en ambas funciones [10].
A fin de acotar el espacio de búsqueda, el algoritmo trabaja con una o más funciones que modelan las restricciones al problema.

- Se define un problema como un conjunto finito de objetivos a resolver, que por la naturaleza del algoritmo serán típicamente más que uno. En este caso son dos, la optimización de aulas y de profesores.
- Se crea una población inicial en forma azarosa, pero en base a los posibles valores del dominio.
- Se calcula el fitness de la población para cada función objetivo.
- Se seleccionan de la población a aquellos individuos que por su tipo de dominancia pasan a formar parte de la elite.
- A continuación se itera una cantidad de veces estipuladas por configuración.
- En cada iteración se cruzan, se mutan, y se seleccionan a los individuos para crear la nueva población.
- Al final de cada iteración se actualiza la población elite de manera que se mejore paulatinamente el fitness de los objetivos del problema.

IV. GDARIM Y SISTEMAS ALTERNATIVOS

En particular Gdarim, pretende resolver el problema de la asignación automatizada y eficiente de aulas. Intenta contemplar parámetros y alternativas posibles para adaptarse a cualquier institución educativa y poder ser implementado con cambios mínimos, ahorrando el arduo de trabajo de buscar, y probar diferentes opciones de distribución hasta llegar a la más convincente.
La mayoría del software registrado hasta el momento fue creado con el fin de facilitar la tarea de asignación manual de aulas. Sin embargo, GDarim, propone resolverlo por completo. Tan solo es necesario ingresar la información requerida, para que sea procesada y posteriormente devuelve las mejores alternativas. Este desarrollo no se encuentra integrado ni asociado con otra aplicación, sin embargo puede configurarse para operar mediante el uso de JODBC o archivos de texto intermedios, lo que brinda aun más flexibilidad y consistencia en la información administrativa.
Gdarim a diferencia de SoftAula [3], por ejemplo, incorpora la inteligencia y capacidad de optimizar el espacio sin necesidad de intervención del operador. El mismo puede ser adaptado para cualquier institución intentando optimizar la distribución eficiente de recursos con solo modificar la función de ajuste (aptitud) y los parámetros de acuerdo a las reglas de negocio dadas.

Aplicaciones como PhpScheduleIt [4], Classroom scheduler [5], representan un modelo de la información en forma concisa y clara. No obstante su lógica es muy rudimentaria en comparación con el prototipo de Gdarim.

V. Estado y trabajo a futuro

Esta propuesta pretende resolver un problema difícil hasta ahora y presentar un aporte para la administración del espacio en las instituciones educativas, resultando muy probablemente un fuerte ahorro de tiempo y dinero. Las aplicaciones existentes constan de desarrollos maduros en ciertos aspectos, pero en ningún caso se ha verificado la asignación completamente automática de recursos. Gdarim es innovador desde la perspectiva técnica porque propone un algoritmo inteligente altamente paramétrico y multi-objetivo como es el Epsilon-MOEA.
Actualmente el proyecto tiene implementado un prototipo inicial que recibe información administrativa y propone hasta 8 alternativas de asignación ponderadas. En los próximos meses, se depurarán los algoritmos para refinar parámetros sensibles a los resultados.

VI. Referencias

[1] “Sistema Matricula”, Pereira Educa, Colombia, http://www.pereiraeduca.gov.co
[2] “Visual Scheduling System,” http://www.vss.com.au/index.asp
[3] “Softaula”, http://www.softaula.com/es/prod/prod_comparativa.asp
[4] “PHPScheduleIt”, software opensource, http://sourceforge.net/projects/phpscheduleit/
[5] “Classroom Scheduler”, software opensource, http://sourceforge.net/projects/cr-scheduler/
[6] “SIU Guaraní”, Univ. Nac. de Comahue, Tecnologías de la Informac., http://www.uncoma.edu.ar
[7] “Introducción a la computación evolutiva”, Anselmo Perez Serrada, 1996
[8] “Modelos de despacho eléctrico económico -ambiental”, Pablo Pizarro. Univ. de Mendoza, 2006
[9] “Modelado de la Distribución de Espacios Físicos mediante Algoritmos Evolutivos”, C.A. Delrieux, IX WICC. 2007.
[10] “Optimización Multiobjetivos del proceso de Torneado”, Ing. R. Quiza Sardiñas, Matanzas. 2004.
[11] “Twenty Years of Evolutionary Multi-Objective Optimization: A Historical View of the Field”, Carlos A. Coello Coello. Mexico, D. F., Nov. 11, 2005.
[12] “Visualization and Data Mining of Pareto Solutions Using Self-Organizing Map”, S. Obayashi et al., 980-8577. Japón.
[13] “Introducción a la Optimización Evolutiva Multiobjetivo”, Carlos A. Coello Coello Sept. 2002, http://neo.lcc.uma.es/