Tesis de grado

Estudio del problema de school timetabling utilizando técnicas de IA

Pablo Ezequiel Sánchez

Feb.2009

Contenido

Presentación
Estado del Arte
Modelado e implementación
Resultados
Conclusiones
Referencias

Presentación

Definiciones
Caso de estudio
Objetivos

Definiciones

Timetabling

El problema del school timetabling consiste en asignar clases de distintas materias escolares en ciertos períodos de tiempo, satisfaciendo diferentes restricciones.

Restricciones

Comprenden hechos tales como evitar los conflictos de horas, capacidad de aulas, carga de trabajo y disposición para estudiantes y profesores, entre otros.

Definiciones

Tipos de Timetabling

- School timetabling: asignación de horas semanales para los cursos de un colegio secundario, evitando que los profesores dicten en dos cursos al mismo tiempo, y viceversa.

- University timetabling: asignación de horas semanales de un conjunto de clases de diferentes materias de la Universidad, minimizando superposición de clases entre años que comparten alumnos.

- Examination timetabling: organización de exámenes de materias de la Universidad, evitando solapar exámenes del mismo año.

Caso de estudio

La motivación de este trabajo surge de la generación del horario semanal de clases en el colegio secundario E.E.M. Nro 2 de la localidad de Henderson, Bs. As.

El mismo se realiza actualmente en forma manual por el encargado. Esta tarea suele llevarle al empleado todo un verano, pues debe tener en consideración múltiples condiciones de bandas horarias y superposiciones de cursos.

Objetivos

Objetivo Principal

- Analizar la utilidad de mecanismos automáticos existentes para la generación de horarios adecuados a la escuela que nos convoca, sugiriendo alternativas y eventuales perfeccionamientos a los mismos.

Objetivos

Objetivos específicos

- Modelar como caso de estudio la escuela citada de modo que el resultado de esa modelización pueda ser utilizado como entrada para el cálculo y estudio de algoritmos de *timetabling*.

- Estudiar cómo se comportan diferentes algoritmos y métodos cuando son aplicados sobre el caso de estudio.

- Analizar los resultados en función de métricas y estándares de la comunidad científica.

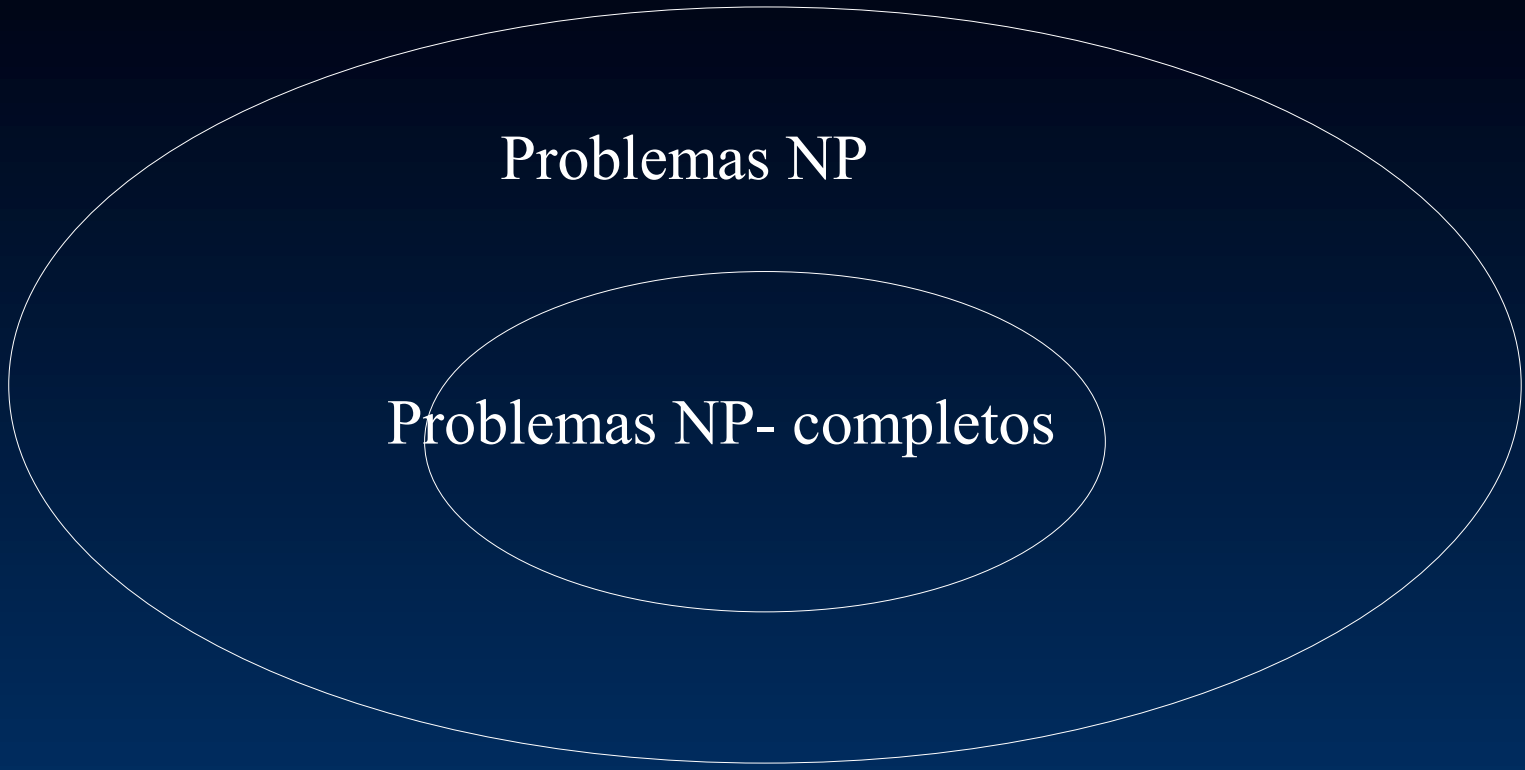
- Obtener conclusiones que permitan decidir el uso de una u otra técnica de Inteligencia Artificial para la resolución del Timetabling Problem.

Estado del Arte

Complejidad del problema

Técnicas de resolución

Complejidad del problema



Complejidad del problema

Problemas NP

- NP significa tiempo polinómico no determinista (nondeterministic polynomial-time) [Weiss, 1997]

- Una forma sencilla de comprobar si un problema está en la clase NP es expresar el problema como una pregunta si/no.

El problema está en NP si , en tiempo polinómico, se puede demostrar que cualquier ocurrencia “si” es correcta.
No tenemos que preocuparnos de las ocurrencias “no”, ya que el programa siempre hace la elección correcta.

Complejidad del problema

Problemas NP-completos

- Entre todos los problemas de los que se sabe que están en NP, hay un subconjunto, llamado problemas NP completos [De Werra, 1985].

- Un problema NP completo tiene la propiedad de que cualquier problema en NP puede ser reducido polinómicamente a él.

- El Timetabling Problem es NP-Completo [Even et al, 1976].

Técnicas de resolución

Heurísticas directas
Lógica de enunciados y Satisfacibilidad
Reducción a la coloración del grafo
Algoritmos genéticos
Simulated annealing
Tabú search

Técnicas de resolución

Heurísticas directas

- Las técnicas iniciales de school timetabling se basaban en la forma humana de resolver el problema.
- Un horario parcial se va extendiendo lección por lección, hasta que todas las lecciones se las ha asignado a una hora. Por ejem. El sistema Schola descripto en [Junginger, 1986]

Lógica de enunciados y Satisfacibilidad

- El problema de Satisfacibilidad Proposicional (SAT) es el problema de determinar la existencia de una asignación de valores para una fórmula proposicional o probar que ésta contiene una contradicción. [Kenneth-Chin Fat, 2004] presenta un método para construir horarios para una escuela secundaria de Rotterdam.

Técnicas de resolución

Reducción a la coloración del grafo

[Neufeld-Tartar,1974] propusieron una reducción del Timetabling Problem al problema de la coloración del grafo. Cada lección se asocia con un vértice del grafo, y hay una arista entre cada par de lecciones que no pueden asignarse a la misma hora a la vez.

Algoritmos genéticos

- El programa evolutivo es un algoritmo probabilístico que mantiene una población de “individuos”. A cada solución se le aplica una función de “fitness” que selecciona los individuos más aptos. Luego de un número de generaciones, el programa se acerca al valor óptimo de la solución.
- Un algoritmo genético ha sido aplicado al problema del school timetabling en [Colorni, 1992]

Técnicas de resolución

Simulated annealing (Recocido simulado)

- El algoritmo selecciona aleatoriamente un candidato de entre los que componen el entorno de la solución actual. Si el candidato es mejor que ella, entonces es aceptado como solución actual; en caso contrario, será aceptado con una probabilidad que decrece.
- Abramson [Abramson, 1991] aplicó Simulated Annealing al problema de school timetabling.

Tabú search

- La búsqueda Tabú enfatiza procedimientos determinísticos en lugar de aleatorios. Estos requieren de la exploración de un gran número de soluciones en poco tiempo.
- Una implementación para timetabling problem utilizando búsqueda tabú es la de [Costa, 1994].

Modelado e implementación

Unificación de Terminología
Restricciones
Coloración del grafo
Algoritmos genéticos
Satisfacibilidad

Unificación de Terminología

Clases
Curso
Materia
Programa académico
Bloque (*Timeslot*)
Horario (*Timetable*)

Unificación de Terminología

Clases

- Corresponde a reuniones que se realizan, donde se imparte el contenido de la materia.

- Una clase i representa a una tupla conformada por un número de lección X de un par materia-profesor Y correspondiente a un curso Z.

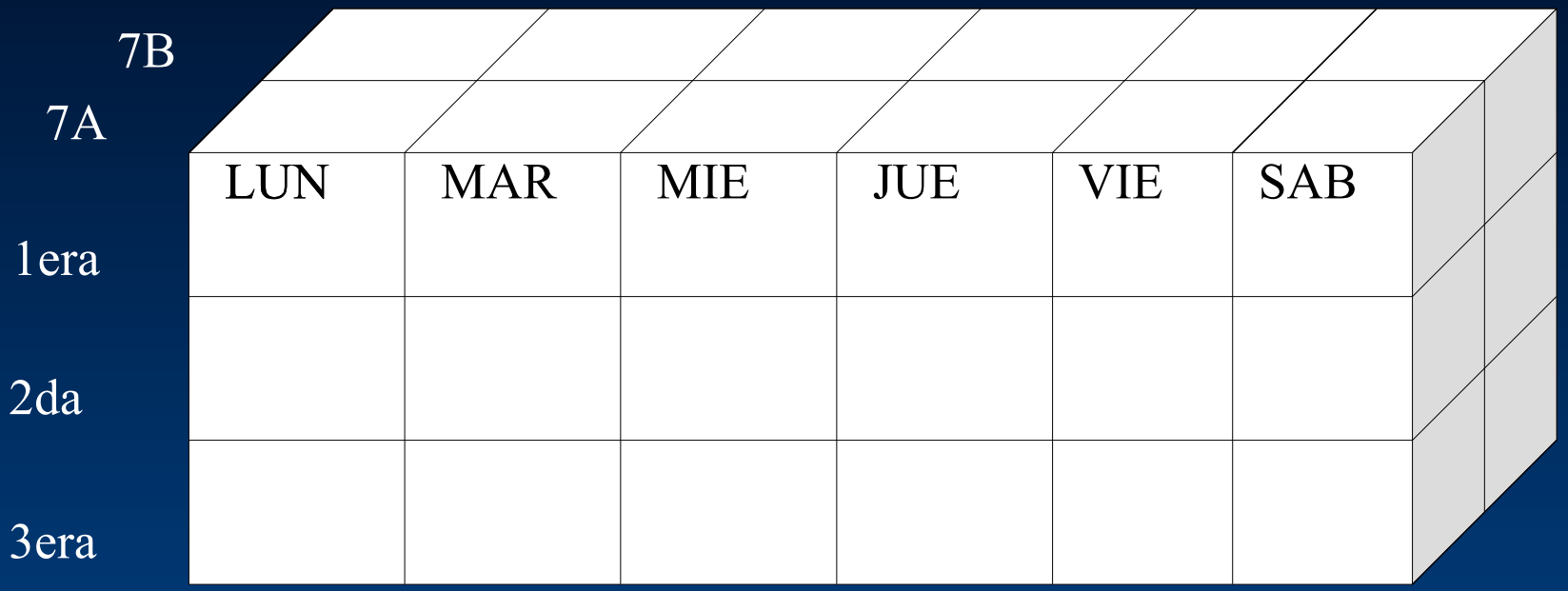
Horario (*Timetable*)

- Es el conjunto de todos los bloques que componen una semana asignados a diferentes clases para cada curso.

Unificación de Terminología

Timetable

- Es el conjunto de todos los bloques que componen una semana asignados a diferentes clases para cada curso.



Unificación de Terminología

Ejemplo de Timetable

Materia - Profesor	Lun	Mar	Mie	Jue	Vie
Sociales – Prof. 1	2 2 - - -	- - - - -	- - - - -	1 1 2 2 -	- - 1 1 -
Sociales – Prof. 2	5 4 4 - -	- - - - -	5 4 - - -	- - - - -	5 5 4 - -
Sociales – Prof. 3	- - 3 - -	- 6 - - -	- - - - -	- - 6 6 3	- - 6 3 3
Naturales – Prof. 4	3 1 2 2 -	3 3 2 - -	1 1 3 - -	4 4 1 - -	4 4 2 - -
Naturales – Prof. 5	- - - - -	- - - - -	- - - - -	6 6 5 5 -	6 6 5 5 -
Lengua – Prof. 6	4 3 1 6 5	- 1 5 3 4	3 5 6 6 -	- - 4 1 6	- 1 3 4 5
Lengua – Prof. 7	- - - - -	- - - - -	2 2 - - -	2 2 - - -	- - - - -
Artística – Prof. 8	6 - - 5 -	1 - - - -	- - 5 - 2	- - - - -	1 2 - 6 -
Artística – Prof. 9	- - - - -	- - 3 4 -	4 3 - - -	- - - - -	- - - - -
Inglés – Prof. 10	- 5 5 3 3	- - - 1 1	- - - - -	- - - - 1	- - - - -
Inglés – Prof. 11	- 6 6 4 4	- - - - -	- - - 4 6	- - - - -	- - - - -
Inglés – Prof. 12	- - - - -	- - - 2 2	- - - - -	- - - - 2	- - - - -
Matemática – Prof. 13	- - - 1 2	2 2 1 - -	- - 2 1 1	- - - - -	- - - - -
Matemática – Prof. 14	- - - - -	4 4 - - -	- - 4 - -	3 3 - 4 -	3 3 - - -
Matemática – Prof. 15	- - - - -	5 5 6 6 -	6 6 - - -	5 5 - - -	- - - - -
Computación–Prof 16	- - - - 6	- - - - -	- - - - -	- - - - -	2 - - - -
Computación–Prof 17	- - - - -	- - - - -	- - - - -	- - - 3 5	- - - - 4
Catequesis – Prof. 18	- - - - -	- - 4 5 6	- - - 5 3	- - 3 - 4	- - - - 6
Catequesis – Prof. 19	- - - - -	- - - - -	- - 1 2 -	- - - - -	- - - 2 1

Curso 7A → 1 . Curso 7B → 2 . Curso 8A → 3 . Curso 8B → 4 . Curso 9A → 5 . Curso 9B → 6 .

Restricciones

Obligatorias

- No asignar dos clases al mismo timeslot.
- Un profesor no puede dictar en dos cursos al mismo tiempo.

Opcionales

- Preferencias de horarios por parte de los profesores.
- Posibilidad o no de dejar timeslots sin asignación de clase, es decir, tener o no horas libres.

Otras

- Cantidad máxima de timeslots por dia que se puede dictar una clase a un curso.
- Una clase de la misma materia no se puede dictar en dos timeslots intercalados para el mismo curso y día.

Coloración del grafo

Grafo

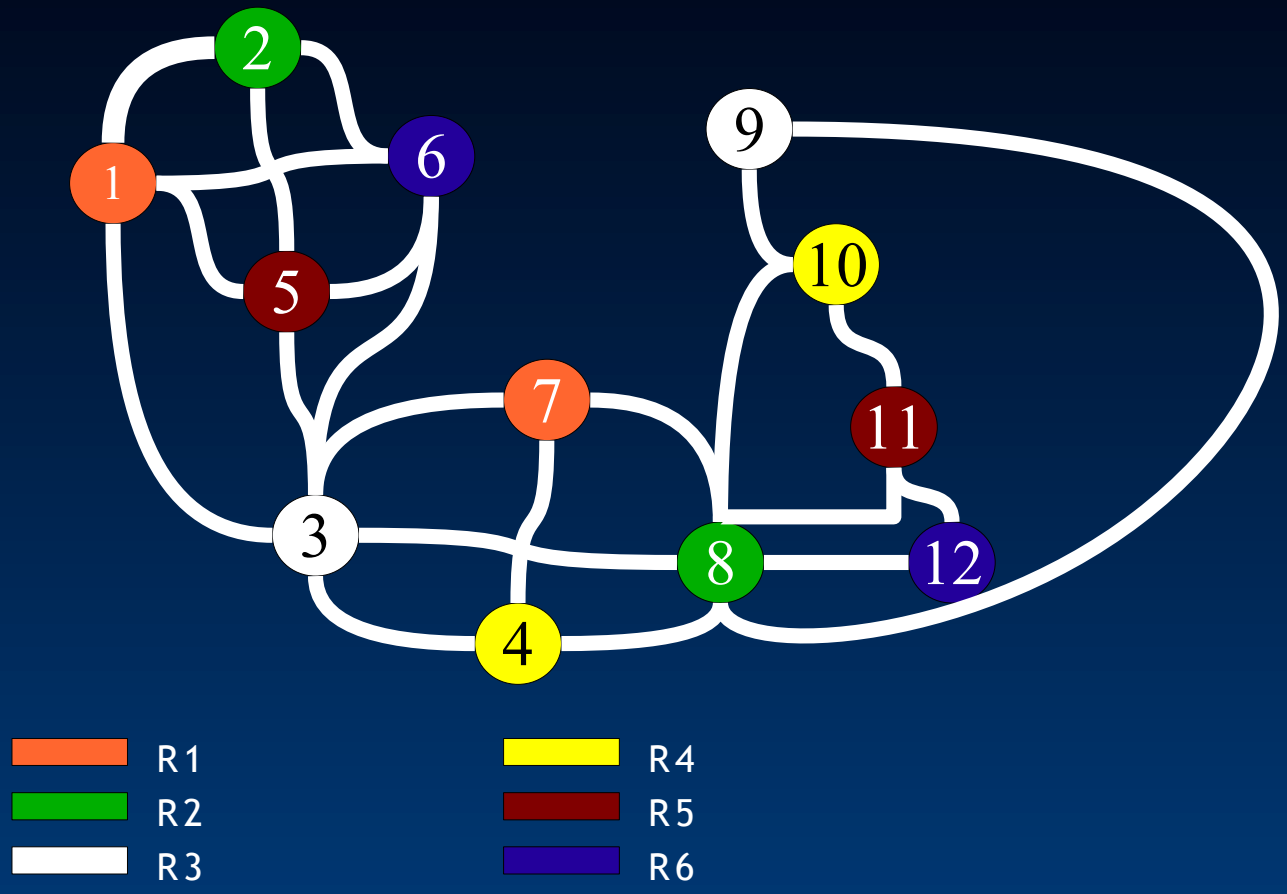
Un grafo G se define como un conjunto de vértices V(G) = $\{v_1, v_2, ..., v_n\}$ y un conjunto de aristas E(G)= $\{e_1, e_2, ..., e_m\}$. Decimos que dos vértices son adyacentes cuando existe al menos una arista entre ellos.

K-coloración

Una k-coloración de G es una función que usa exactamente k colores y satisface la propiedad de que f(x) /= f(y) para x e y vértices adyacentes en G. [Chartrand, 1993]

Coloración del grafo

Ejemplo Grafo



Coloración del grafo

Algoritmo Dsatur [Brelaz, 1979]

1. Ordena los vértices por orden decreciente de grados de saturación
2. Colorea el vértice de máximo *degree* con el color 1
3. Elige un vértice con el máximo Dsat
4. Colorea este vértice con el color más chico posible
5. Si todos los vértices están coloreados termina, sino vuelve a el paso 3.

Coloración del grafo

Timetabling

A) Representamos un *Timetable* como un grafo. Representación utilizada por [Redl, 2004]

B) Utilización de Dsatur

C) Interpretación del *Timetable* resultado.

Algoritmos genéticos

Sistemas basados en evolución

- Y en herencia.. mantienen una “población” de soluciones posibles

Población

- Se conforma por un número de individuos, o cromosomas. Cada individuo representa una solución potencial al problema.

Proceso de selección

- Basado en la función *fitness*, que da una medida del grado de optimización de un cromosoma.

Algoritmos genéticos

Algoritmo genético FET

1) Genera una población aleatoria de n cromosomas
2) Evaluación del *fitness* f(x) de cada cromosoma x en la población
3) Creación de una nueva población repitiendo los pasos:

3a) Selecciona dos cromosomas padre de la población, de acuerdo a su *fitness*
3b) Se cruzan los padres para formar un nuevo hijo. Si no se realiza la cruza, el hijo es un clon de los padres.
3c) Mutación del cromosoma hijo.

4) Usar la población generada para una nueva ejecución del algoritmo
5) Si se satisface la condición de terminación, termina y retorna la población actual como mejor solución.
6) Volver al paso 2)

Algoritmos genéticos

Ejemplo de cruza

1	2	3	4	5	6	7	8	9	10
1	**4**	**2**	**0**	**1**	**0**	**3**	**5**	**0**	**1**

1	2	3	4	5	6	7	8	9	10
0	**3**	**5**	**0**	**4**	**4**	**0**	**1**	**0**	**2**

Punto de cruza

1	2	3	4	5	6	7	8	9	10
1	**4**	**2**	**0**	**4**	**4**	**0**	**1**	**0**	**2**

Algoritmos genéticos

Timetabling

A) Representamos un *Timetable* con un cromosoma. Como hace [Lalescu,2003]

B) Utilizamos el algoritmo FET.

C) Interpretamos la población de *Timetables* que halla el algoritmo.

Satisfacibilidad

Fórmula en lógica proposicional [Hamilton,1981]

- Es una cadena bien formada que puede contener:
  - Variables proposicionales $x_1, x_2, .. , x_n$
    - Valores de verdad: V (Verdadero), F (Falso)
    - Operadores: $\neg$ (no), $\vee$ (ó), $\wedge$ (y)
    - Paréntesis (para anidar subfórmulas)

Modelo

- de una fórmula S, es una asignación de valores de verdad a las variables en S que hacen que S sea Verdadera

Fórmula satisfactible

- La fórmula S es satisfactible si y sólo si existe al menos un Modelo de S. es insatisfactible en caso contrario.

Satisfacibilidad

Método Davis-Putnam [DPLL,1962]

Sea S una fórmula FNC y ∞ una asignación vacía (o parcial)
El método consiste en:

- Chequear las dos condiciones siguientes:
    1. Si S está vacía, retornar V y ∞
    2. Si S contiene la cláusula vacía, retornar F y {}

- Aplicar una de las dos reglas :
    1. Regla de cláusula Unitaria
    2. Regla de Splitting

El método retorna V y ∞ en cuyo caso S es satisfactible y ∞ es un Modelo; ó F y ∞ = {} en cuyo caso S es insatisfactible.

Satisfacibilidad

Timetabling

- Representamos un Timetable como una FNC (Fórmula Normal Conjuntiva), como hacen en el trabajo de [Kenneth-Chin Fat,2004]

- Testeo de satisfacibilidad mediante Davis-Putnam

- Interpretación del Modelo obtenido.

Satisfacibilidad

Ejemplo FNC

$$(x_{1,1,1,1,1} \vee x_{1,1,1,1,2} \vee x_{1,1,1,2,1} \vee x_{1,1,1,2,2}) \wedge \quad \text{cláusula 1.-}$$

$$(x_{1,2,1,1,1} \vee x_{1,2,1,1,2} \vee x_{1,2,1,2,1} \vee x_{1,2,1,2,2}) \wedge \quad \text{cláusula 2.-}$$

$$(x_{1,3,1,1,1} \vee x_{1,3,1,1,2} \vee x_{1,2,1,2,1} \vee x_{1,3,1,2,2}) \wedge \quad \text{cláusula 3.-}$$

$$(x_{1,4,1,1,1} \vee x_{1,4,1,1,2} \vee x_{1,2,1,2,1} \vee x_{1,4,1,2,2}) \wedge \quad \text{cláusula 4.-}$$

Resultados

Requerimientos del caso de estudio

- Tamaño del problema: 143 clases, 25 *timeslots*, 6 cursos, 19 profesores y 8 materias por curso.

- Duración de la clase: 1 *timeslot*. Cada *timeslot* representa un período del día (hay 5 *slots* por día)

- Restricciones obligatorias: las 2 restricciones antes nombradas

- Restricciones opcionales:

a) Preferencias de horarios por parte de los profesores
b) Posibilidad o no de dejar *timeslots* libres.

2 Variantes a testear: con preferencias y sin ellas.

Resultados

Función de puntuación

- Clase asignada en un *timeslot* NO preferido: 1 punto
- Clase NO asignada (hueco en el *timetable*): 5 puntos
- Materia con más de 2 hrs en un día y un curso: 10 puntos

Los valores son una medida arbitraria que intentan penalizar con más peso las restricciones opcionales consideradas más importantes.

Resultados

Plan de pruebas

1) Representar los datos de entrada
2) Generar los archivos de entrada para los diferentes algoritmos
3) Ejecutar los programas Dsatur, Sato y Fet
4) Ingresar los resultados para imprimir el *timetable* de forma legible por el usuario
5) A los *timetables* obtenidos aplicar la función de puntuación para medir su calidad.

Resultados

Calidad de los timetables – sin preferencias

Penalización	DSATUR	SATO	FET	Resolución Manual
Clase asignada en un timeslot NO preferido	NO	NO	NO	NO
Clase No asignada	35	35	35	35
Materia con más de 2 hrs en un día y un curso	100	130	40	0
Total de penalización	135	165	75	35

Resultados

Calidad de los timetables – con preferencias

Penalización	DSATUR	SATO	FET	Resolución Manual
Clase asignada en un timeslot NO preferido	4	0	NO	NO
Clase No asignada	35	35	35	35
Materia con más de 2 hrs en un día y un curso	100	40	40	0
Total de penalización	139	75	75	35

Resultados

Resumen

Area testeada	Grafo	SAT	AG
Representatividad del problema	Difícil	Difícil	Fácil
Técnica independiente del problema	Si	Si	No
Cantidad de soluciones por ejecución	0-1	0-1	0-N, N= tamaño de la población
Eficiencia en tiempo	milisegundos	milisegundos	segundos
Calidad de la solución	Mala	Buena	Muy Buena

Conclusiones

Problemas

- No pudimos comparar los algoritmos utilizando una medida o estándar internacional.

- No pudimos comparar búsqueda tabú y *simulated annealing* pues no hallamos implementaciones de código abierto.

- Coloración de grafos: cumplió con una asignación básica

- Sat: se hace difícil la representación mediante una sola fórmula proposicional

- Fet: tiempo de ejecución

Conclusiones

Aciertos

- Utilizamos datos reales de la Escuela Secundaria

- Conseguimos implementaciones de código abierto de las técnicas estudiadas

- Realizamos una unificación de terminología para realizar cuadros comparativos de las técnicas (por ej. definición de Clase y *Timetable*)

- Utilizamos formatos de archivos estándares (Dimacs y Xml)

- Diseñamos e implementamos una aplicación Pascal que genera las entradas a los algoritmos, toma las salidas de los mismo, y finalmente imprime los *timetables*.

Conclusiones

Obtuvimos resultados coherentes con nuestras expectativas

Encontramos una respuesta afirmativa al uso de técnicas de la Inteligencia Artificial

Confirmamos lo que indican las tendencias de la comunidad científica con respecto al uso de éstas técnicas.

Conclusiones

Trabajos Futuros

- Utilización del Cálculo Proposicional como complemento a la lógica proposicional

- Realizar una combinación de la técnica de coloración del grafo con otras heurísticas, por ej. Backtracking.

- Agregar en la comparativa Búsqueda tabú y simulated annealing

Referencias

[Brelaz, 1979] New Methods to color vertices of a graph.
[DPLL,1962] A machine program for theorem proving.
[Lalescu, 2003] Timetabling experiments using genetic algorithms.
[Schaerf, 1995] A survey of Automated Timetabling.
[Swee-Chuan,2003] An object oriented timetabling framework with applications.

FIN