Programación de Horarios de Clases y Asignación de Salas para la Facultad de Ingeniería de la Universidad Diego Portales Mediante un Enfoque de Programación Entera

Rodrigo Hernández[*]
Jaime Miranda P.[**]
Pablo A. Rey[***]

Resumen

Un aspecto importante en la gestión académica de las universidades es la generación de horarios y la asignación de salas de clase para los distintos cursos que realizan. En este artículo se presenta un modelo de programación entera el cual decide simultáneamente los horarios de los cursos y la asignación de salas. Las variables utilizadas están asociadas a la definición del horario del curso para una semana por medio de un patrón horario. Una particularidad del modelo es que tanto las condiciones sobre capacidad y tipo de salas de clase, así como las combinaciones de bloques horarios para un curso, son manejadas implícitamente mediante las variables de decisión. Se reportan los resultados da la comparación de la programación obtenida con el modelo propuesto y la programación que efectivamente se utilizó. El modelo propuesto entrega de manera rápida y eficiente los horarios y asignaciones de sala de clase satisfaciendo todos los requerimientos obligatorios y condiciones deseables para la Facultad en un tiempo menor a los 5 minutos.

Palabras Clave: *Timetabling*, *class scheduling*, programación entera.

[*]Departamento de Ingeniería Industrial, Facultad de Ciencias Físicas y Matemáticas, Universidad de Chile, Santiago, Chile.

[**]Departamento de Control de Gestión y Sistemas de Información, Facultad de Economía y Negocios, Universidad de Chile, Santiago, Chile.

[***]Escuela de Ingeniería Industrial, Facultad de Ingeniería, Universidad Diego Portales, Santiago, Chile.

1. Introducción

Las instituciones educacionales enfrentan cada semestre el problema de la programación de horarios y asignación de salas de clase de los cursos que imparten. Desde la perspectiva de la Investigación de Operaciones, este tipo de problemas se enmarcan dentro del área conocida como *Timetabling* o programación horaria. Los problemas de esta área consisten en la asignación de ciertos eventos a distintos bloques horarios respetando una serie de requerimientos y condiciones. Dentro de estos problemas existe una rama específica, llamada *Class Scheduling*, que estudia problemas relacionados con la programación horaria para entidades educativas. Dentro de este contexto, existen tres tipos de problemas [45]:

- Programación de horarios de evaluaciones y exámenes (*Examination Timetabling*).
- Programación de horarios de clases para colegios (*School Course Timetabling*).
- Programación de horarios de clases para instituciones de educación superior o universidades (*University Course Timetabling*).

Como ya hemos señalado, la Facultad de Ingeniería debe resolver el problema de generar la programación horaria de sus cursos y la asignación de salas de clase, o sea, resuelve un problema del tipo *University Course Timetabling*.

Esta programación debe satisfacer una serie de requerimientos impuestos por políticas de la Facultad. Actualmente, el proceso de programación se realiza de forma manual demorando aproximadamente un mes en promedio. Cabe destacar que, la programación obtenida de esta manera no está libre de errores, detectándose en algunos casos ciertas ineficiencias e incumplimientos de los requerimientos básicos para el correcto funcionamiento de la institución educativa.

En consecuencia, una buena programación horaria genera una serie de beneficios para los principales actores que conviven en esta institución. Entre estos es posible mencionar por ejemplo: eliminar topes de horarios entre cursos del mismo semestre, respetar la disponibilidad de horarios de los profesores, respetar la capacidad de las salas de clase e incorporar condiciones deseables, como por ejemplo: favorecer las clases en bloque horarios específicos o minimizar la utilización de de salas especiales.

En este trabajo, se propone un modelo de programación entera para la generación de la programación horaria y asignación de salas. Este modelo

incorpora, por un lado, todos los requerimientos y condiciones deseables y, por otro, los objetivos perseguidos por la Facultad.

Siguiendo esta línea, la estructura de este artículo es la siguiente: la sección 2 presenta los antecedentes relevantes para esta investigación. Primero, se expone un análisis bibliográfico en relación al problema de programación horaria presentando los principales enfoques de solución. Luego, se realiza una descripción del problema presentando sus principales características y requerimientos. La sección 3 detalla el modelo de programación entera. La sección 4 analiza los resultados computacionales de la implementación del modelo y se compara, en términos de indicadores de desempeño, con la actual forma de operar para el semestre Otoño 2007. Finalmente, la sección 5 presenta las conclusiones de este estudio.

2. Antecedentes

2.1. Análisis Bibliográfico

Los problemas de generación de horarios y asignación de recursos en instituciones educativas han sido ampliamente estudiados en la literatura [5, 44, 45]. Estos problemas pueden ser clasificados de acuerdo al tipo de institución educativa (colegios o universidades) y por el tipo de eventos a programar, clases o evaluaciones. Calendarizar clases en colegios o universidades son problemas bien diferentes en la práctica. Usualmente, en los colegios los alumnos que pertenecen a un determinado curso toman en bloque las mismas asignaturas, pues se desean horarios compactos de clases[1]. En el caso de las universidades en cambio, debe existir cierta flexibilidad en los horarios y en la selección de los cursos que toma cada estudiante. La programación de clases y evaluaciones difieren, principalmente, en los aspectos siguientes [13]:

1. Las evaluaciones deben ser programadas después de la inscripción de los alumnos, mientras que los cursos son usualmente programados con anterioridad a la inscripción de los alumnos.

2. Las restricciones asociadas al uso de salas de clase pueden ser diferentes. Las clases de un curso deben ser programadas, por lo general, en una misma sala de clase. Para las evaluaciones los requerimientos pueden ser diferentes, ya que, en algunos casos, se programan evaluaciones de diferentes cursos compartiendo la misma sala de clase y, en otros casos, se requiere programar las evaluaciones de un mismo curso en varias salas de clase debido a la cantidad de alumnos inscritos en dicho curso.

[1] Horarios con clases seguidas o sin ventanas temporales entre cursos sucesivos.

A continuación se realiza una revisión bibliográfica de los trabajos publicados en esta área. La literatura presenta numerosas variaciones del problema de programación de horarios de acuerdo a los requerimientos específicos de cada institución. En particular, más allá de las restricciones de topes en la programación de horarios de profesores y uso de salas de clase, no hay otras condiciones que aparezcan en todos los estudios presentados en la literatura [33]. Sin embargo, existe una gran variedad de enfoques de solución para la generación de horarios y asignación de salas de clase. Mientras algunos trabajos se concentran en los aspectos prácticos y el desarrollo de sistemas, otros lo hacen en el modelamiento y en metodologías de solución.

Según las características de la institución, la programación de los horarios se realiza con antelación a la inscripción de los alumnos, o luego de ésta. Entre los trabajos que analizan el problema después que los alumnos han escogido sus cursos podemos mencionar el trabajo reciente de Boland *et al.* [9] en donde no se considera la capacidad de las salas de clase, sino que se definen un número predeterminado de secciones en cada curso y un número máximo de estudiantes por sección. Adicionalmente, en otros trabajos consideran la programación de los horarios de los cursos después de la inscripción de los alumnos [4, 18, 27].

En nuestro caso particular, la programación se debe realizar con anterioridad a la inscripción de los alumnos. En este caso, se deben imponer condiciones sobre los topes de horarios de los cursos que compartirán alumnos de acuerdo a estimaciones de demanda o al plan de estudios. Como veremos en la próxima sección, en nuestro problema, las condiciones impuestas corresponden a evitar topes horarios entre los cursos del mismo semestre, de acuerdo a los planes de estudio de las carreras que imparte la Facultad.

Las condiciones o requerimientos del problema pueden ser clasificados de acuerdo a su naturaleza en cinco grupos [21, 33]:

- *Restricciones unarias*, aquellas que involucran un sólo evento, como por ejemplo, las clases de un curso no pueden ser programadas un día lunes.
- *Restricciones binarias*, aquellas que involucran dos eventos. Un ejemplo típico son las restricciones de topes de horarios para un curso que requiere un mismo recurso: profesor, sala de clases, etc.
- *Restricciones de capacidad*, como por ejemplo, las que se imponen al asignar cursos a salas de clase con capacidad suficiente.
- *Restricciones de separación de eventos*, aquellas que requieren que las actividades estén separadas o siguiendo algún patrón en el tiempo. Algunos ejemplos son las impuestas por políticas de la institución de respetar asignaciones de horarios en patrones predefinidos o las condiciones de no existencia de horas intermedias vacías.

- *Restricciones asociadas a los agentes*, como son las limitaciones en los horarios asignados para cumplir con las preferencias de los profesores.

Cuando todas las condiciones no pueden ser satisfechas simultaneamente, lo común es divididirlas en *requerimientos fuertes* que deben cumplirse obligatoriamente y *requerimientos suaves* que no son obligatorios pero sí deseables. La *calidad* de la programación obtenida dependerá del grado de cumplimiento de estas condiciones. Cuando el problema es enfrentado mediante el uso de modelos de optimización, los requerimientos fuertes son utilizados como restricciones, mientras que los requerimientos blandos son incluídos como un termino en la función objetivo, las cuales al ser violadas se penalizan.

Entre las diversas técnicas de modelamiento y solución consideradas en la literatura, se destacan el uso de modelos de optimización, principalmente programación lineal y programación lineal entera, heurísticas y metaheurísticas, además de la programación por restricciones o *constraint programming*. Entre los trabajos que consideran este último enfoque podemos citar [1, 30, 43].

Numerosos son los trabajos que reportan el uso de heurísticas y metaheurísticas [26, 33, 39, 42]. Otros trabajos se concentran en el uso de metodologías particulares como por ejemplo: búsqueda tabú [4], *simulated annealing* [26], colonias de hormigas [46], algoritmos genéticos [50], búsqueda local [29], métodos multiagentes [37, 48], metaheurísticas [14, 15, 41]. Algunos trabajos, como [20, 51], analizan métodos híbridos que combinan una o más metodologías básicas enunciadas anteriormente.

La metodología presentada en este trabajo se basa en un modelo de programación entera. Hay una gran cantidad de trabajos que atacan problemas similares utilizando modelos de optimización.

Al-Yacoob y Sherali [2] proponen dos modelos para la asignación de profesores a las clases. Estos toman como información de entrada la asignación previa de clases a horarios y tratan de asignar los profesores de manera a satisfacer sus preferencias. Uno de los modelos realiza la asignación sin modificar los horarios preestablecidos, mientras que el otro considera la posibilidad de realizar modificaciones en los horarios manteniendo el uso eficiente de las salas de clase. En otro trabajo [3], estos mismos autores, proponen el uso de un modelo de programación entera mixta que tiene como objetivo mejorar las programaciones de clases e incorporar nuevas condiciones impuestas por la *Kuwait University*.

Avella y Vasil'Ev [6] proponen un algoritmo de tipo *branch-and-cut* para un problema de asignación de horarios. La formulación utilizada está basada en un problema de *set packing* y son utilizadas como cortes desigualdades del tipo cliques. El problema es estudiado poliedralmente y son derivadas otras familias de desigualdades, no válidas en general para el *set packing*, y sus problemas de separación.

Daskalaki *et al.* [23] proponen un modelo de programación entera para un problema de generación de horarios. El modelo considera como función objetivo una función lineal que representa las preferencias de los profesores en horarios y en salas asignadas para sus clases. Los autores también analizan cómo la definición apropiada de los coeficientes en esta función objetivo permite reducir el espacio de soluciones y vuelve tratable el problema. Daskalaki y Birbas [22] presentan un algoritmo en dos etapas para la solución de un modelo de programación entera para la programación de horarios. En una primera etapa, el modelo de programación entera es relajado, eliminándose restricciones que corresponden a la contigüidad de sesiones para algunos cursos que así lo requieren. Estas condiciones son recuperadas en la segunda etapa, donde los horarios diarios son optimizados, incluyendo las restricciones no consideradas en la primera etapa.

MirHassani [35] presenta un modelo de programación entera para un problema similar al considerado en nuestro trabajo. El modelo propuesto asigna cada clase individualmente a un horario. Junto con las restricciones de topes horarios y usos de salas, son impuestas condiciones y políticas de la institución, como por ejemplo, que las clases de los cursos que tienen más de una clase por semana no pueden ser programadas el mismo día o en días consecutivos o la existencia de combinaciones de horarios que permitan tomar a un alumno todos los cursos de un semestre de su carrera, de acuerdo al plan de estudios.

Todos estos trabajos presentan modelos que asignan por separado cada clase de un curso a un bloque horario y por medio de restricciones se imponen las condiciones de regularidad de las clases. A diferencia de estos trabajos, Qualizza y Serafini [40] proponen un modelo de programación entera basado en generación de columnas. En este caso, las columnas están asociadas al horario semanal de un curso, es decir, todas las clases de la semana son programadas simultáneamente. Las restricciones en el problema maestro corresponden a la ocupación de salas y a los topes horarios. Las restricciones particulares a cada curso están incluídas en el subproblema.

El modelo considerado en nuestro trabajo está basado en una idea similar. Ciertos patrones de horarios son predefinidos y las cursos son asignados a estos patrones. A diferencia de lo propuesto en [40], las variables son definidas explícitamente y no generadas dinámicamente. Otra diferencia importante es que en nuestra propuesta, las clases auxiliares y las clases de cátedra de un curso son programadas por separado.

Existen otros trabajos que basados en modelos de programación entera presentan particularidades, como el uso de múltiples criterios o enfoques en varias etapas. Badri [7] propone un modelo de programación entera multiobjetivo para la programación de clases. El autor analiza un procedimiento en dos etapas para resolver el modelo propuesto. En una primera etapa, se busca maximizar las preferencias de los profesores en relación a los cursos

que les son asignados y, en la segunda, se asignan los horarios de clases a los cursos buscándose maximizar las preferencias de los profesores en cuanto a sus horarios de clases. Badri *et al.* [8] proponen un modelo de programación entera multiobjetivo. Este nuevo modelo busca asignar, simultáneamente, profesores a cursos y diseñar el horario de estos cursos de manera de maximizar las preferencias de los profesores. Estas preferencias, tanto por horarios como por cursos, son incluídas por medio de una tabla que registra las tres opciones preferidas tanto de cursos como de horarios. Dimopoulou y Miliotis [25] analizan la implementación de un sistema computacional de generación de horarios en un ambiente distribuído utilizando un modelo de programación entera para diseñar los horarios para cada departamento. El sistema cuenta con una base de datos central y un procedimiento automático se encarga de generar los problemas de optimización para cada departamento y de resolver los conflictos que pudieran aparecer. Stallaert [47] propone una metodología que divide el problema en dos subproblemas: primero se programan los cursos principales mediante un modelo de programación entera y, luego, utilizando esta programación como información de entrada, se programa el resto de los cursos resolviendo una variante de un problema de asignación cuadrático. Tripathy [49] propone un modelo de programación entera para un problema de programación de clases. El autor incorpora una simplificación al problema, agrupando cursos que pueden ser programados en el mismo horario. El trabajo considera un algoritmo basado en relajación lagrangiana y un algoritmo de tipo *branch-and-bound* para resolver la problemática propuesta.

Finalmente, algunos trabajos se concentran en el desarrollo y construcción de sistemas de apoyo a las decisiones, como por ejemplo: [4, 18, 24, 28, 31, 32, 36, 37, 38]. En particular, McCollum [34] analiza cuestiones prácticas y problemas que surgieron a la hora de implementar un sistema centralizado en una universidad británica.

2.2. Descripción de Problema

La Facultad de Ingeniería imparte en total cuatro carreras en pregrado: Ingeniería Civil Industrial, Ingeniería Civil en Computación, Ingeniería Civil en Obras Civiles e Ingeniería en Construcción. Las primeras tres carreras tienen una duración de 12 semestres, en tanto que la última, tiene una duración de 10 semestres. Cabe destacar que existe una malla curricular, la cual determina el orden en que los alumnos deben tomar los distintos cursos para cada carrera. De este modo, es posible caracterizar a cada curso por el semestre en que se ubica dentro del plan de estudio de la carrera. Esta información es útil a la hora de definir qué cursos no pueden ser dictados en forma simultánea en un mismo bloque horario.

Cada semestre se dictan en promedio 150 cursos los que tienen un número variable de secciones. Para un curso el número de secciones puede variar desde una sección a diez secciones paralelas. Los cursos tienen dos tipos de clases: *cátedra* y *auxiliares*. Cada curso tiene al menos una una clase de *cátedra* semanalmente con un máximo de tres, en cambio para el caso de las clases auxiliares, existe la posibilidad de que un curso no tenga ninguna clase auxiliar o, en su defecto, tenga a lo más una clase por semana. Las clases se realizan de lunes a viernes en bloques de 1 hora y media de duración. Cada día se compone de 6 bloques horarios definidos por las letras A, B, C, D, E y F[2]. Las clases de cátedra deben seguir un patrón horario[3] definido por la Facultad. Respecto a las clases *auxiliares*, la Facultad tiene como condición deseable que estas se realicen el día miércoles, de no ser posible pueden dictarse en cualquier día y bloque horario. Cabe destacar que una de las condiciones impuestas es que para cada curso se debe respetar el mismo patrón horario para cada semana del semestre.

La Facultad de Ingeniería cuenta con 45 salas de clase, las cuales son compartidas por todos los cursos de las carreras impartidas. Las salas de clase se caracterizan por su capacidad, definida como el número máximo de alumnos que es posible asignar para un bloque horario. Estas salas se clasifican en 6 grupos: salas normales, laboratorios de física, laboratorios de computación, laboratorios de obras civiles, laboratorios de simulación de procesos y un Auditorio. El Auditorio es una sala de clase de mayor capacidad y tecnología utilizado para eventos importantes y de alta convocatoria. Cabe destacar que las salas de clase son el recurso escaso de la Facultad.

La Facultad dispone de un staff de 150 profesores para realizar las clases de *cátedra* y cerca de 100 profesores *auxiliares* para realizar las clases *auxiliares*. Cada profesor de cátedra se caracteriza por los cursos que dicta y por su disposición horaria. Para el caso de los profesores *auxiliares* no se consideran sus disposiciones, ya que son alumnos de la Facultad quienes, en estricto rigor, se acomodan a los horarios preestablecidos por la Facultad para los distintos cursos.

Actualmente, la programación horaria es generada por un equipo conformado por 3 profesionales, los cuales demoran en promedio un mes para obtener la programación final. La programación final para un semestre cualquiera se genera principalmente sobre la base de la programación horaria utilizada en el semestre anterior. Esta última es actualizada solamente al existir un nuevo requerimiento, al incorporar un curso nuevo en el plan de estudios o al haber cambios en las preferencias horarias de los profesores. Cabe destacar que esta programación no está exenta de errores, observándose una serie de ineficien-

[2]El bloque A es el primer bloque del día y el bloque F es el último.

[3]Se entiende como patrón horario a una combinación entre uno o más días con uno o más bloques horarios.

cias a la hora de realizar la asignación de salas de clase y un sinnúmero de conflictos entre los horarios de cursos de un mismo semestre.

Considerando los antecedentes planteados anteriormente, la generación de la programación de horarios y asignación de salas de clase se transforma en una tarea en extremo compleja y que consume una enorme cantidad de recursos. Por este motivo, el modelo propuesto busca que los requerimientos impuestos por la Facultad sean apropiadamente estructurados mediante la formulación de un modelo de programación entera. La resolución de este modelo entregará la programación de horarios y asignación de salas de clase de manera óptima respecto de alguna función objetivo.

2.2.1. Requerimientos Impuestos por la Facultad de Ingeniería

A continuación, se describen los requerimientos impuestos por la Facultad de Ingeniería para la programación horaria y asignación de salas de clase. Estos requerimientos fueron categorizados en dos grupos: *fuertes* y *suaves*. Los requerimientos *fuertes* deben ser cumplidos obligatoriamente y los requerimientos *suaves*, que si bien no son obligatorios, representan condiciones deseables para la Facultad. Los requerimientos *suaves* se incorporaron dentro de la función objetivo, la cual trata de minimizar el número de veces en que no se cumplen estos requerimientos. Cada vez que estos no se cumplan se incurre en un penalización.

Requerimientos Fuertes

1. Cada curso debe ser asignados a una sala de clase con capacidad suficiente para la demanda estimada de alumnos para dicho curso.

2. En una sala de clase, en un mismo día y bloque horario, se puede realizar a lo más una clase (*cátedra* o *auxiliar*).

3. Un profesor no puede dictar más de una clase a la vez.

4. Se deben respetar los horarios disponibles de los profesores.

5. No deben existir topes de horarios entre cursos de un mismo semestre.

6. Cada curso debe seguir alguno de los patrones horarios impuestos por la Facultad para la realización de sus clases. Existen cuatro tipos de patrones horarios:

   a) *Patrón 1*: Está compuesto por la combinación entre un día de la semana y un bloque horario. Por ejemplo, un patrón de esta clase es {LU-A} que corresponde al día lunes bloque horario A.

b) *Patrón 2*: Está compuesto por la combinación de un día de la semana y dos bloques horarios consecutivos. Por ejemplo un patrón de este grupo puede ser: {MA-C, MA-D} que corresponde al patrón que contiene el día martes y los bloques horarios C y D.

c) *Patrón 3*: En este caso los patrones deben contener dos días de la semana distintos y un bloque horario. Las combinaciones posibles son: lunes-jueves y martes-viernes. Un ejemplo de este tipo de patrón es {LU-E, JU-E}, o sea el patrón contiene los días lunes y jueves en el bloque horario E.

d) *Patrón 4*: Este grupo de patrones está compuesto por la combinación de tres días de la semana distintos y un bloque horario. En este caso, se permite cualquier combinación de días, es decir, los patrones de este tipo son aquellos de la forma {D1-X, D2-X, D3-X} donde $D1$, $D2$ y $D3$ son tres días distintos de la semana y X es un bloque horario cualquiera.

Requerimientos Suaves

1. Las clases auxiliares deben realizarse de preferencia los días miércoles en cualquier bloque horario. De no ser posible esta asignación, se pueden realizar en cualquier día y bloque horario.

2. Se debe evitar, en lo posible, asignar cursos al Auditorio.

En la próxima sección se detalla el modelo propuesto para enfrentar el problema descrito.

3. Enfoque de Solución

Se considera como unidad básica de modelamiento a lo que llamamos un (par) *curso-sección*. Esta unidad corresponde simplemente a una sección de un curso. La metodología desarrollada se basa en un modelo de programación entera que integra la definición de la programación de horarios y la asignación de salas de clase. El modelo, de manera similar al propuesto por Qualizza y Serafini [40], asigna de manera conjunta todas las clases de cátedra o auxiliares de un curso-sección correspondientes a una semana.

Para esto, se definen combinaciones de bloques horarios-salas factibles. Estas combinaciones de horarios y salas corresponden a agrupaciones de pares (bloque horario, sala) que llamamos *patrones horarios-salas* o simplemente, *patrones HS*. La idea es que el modelo decida si programar o no las clases de cátedra o auxiliares de un curso-sección directamente en un patrón HS sin

identificar cada una de las clases de un curso-sección en una variable distinta, sino que asignándolas todas como conjunto al patrón. Por ejemplo, si un curso posee dos cátedras que deben ser asignadas a bloques horarios consecutivos del mismo día y a la misma sala, entonces sus dos cátedras se asignarán a algún patrón HS de la forma: $\{(t, s), (t + 1, s)\}$ con t y $t + 1$ bloques horarios consecutivos del mismo día y s una sala de clase adecuada para el curso-sección en cuestión.

De esta manera, las condiciones que todas las clases de cátedra de un curso-sección deberán ser realizadas en la misma sala y que los bloques horarios asignados correspondan a una de las combinaciones definidas por la Facultad se manejan de manera implícita al definir las variables.

Antes de describir el modelo, definimos la notación utilizada. Sea I el conjunto de pares curso-sección, S el conjunto de salas disponibles, P, el conjunto de profesores y H el conjunto de semestres de las mallas académicas de las carreras impartidas por la Facultad. El conjunto de bloques horarios de la semana es denotado por T. Los elementos de este conjunto son pares de la forma "día-bloque", por ejemplo LU-A es el primer bloque de la semana y MI-F, es el último bloque del día miércoles. El conjunto de los patrones HS es denotado por B. Por último, existen ciertos grupos de cursos cuyas clases no deben topar por diferentes motivos (por ejemplo, porque corresponden al mismo semestre del plan de estudios). A la familia de grupos de curso-sección cuyas clases no deben topar en horario lo denotamos por L.

Adicionalmente, son necesarios los siguientes subconjuntos de los conjuntos que acabamos de definir:

- $CPR_p \subseteq I$: Conjunto de cursos-sección asignados profesor p.
- $SMC_{hl} \subseteq I$: l-ésimo grupo de cursos-sección del semestre h que no deben tener topes horarios.
- $HR_p \subseteq T$: Conjunto de disponibilidad horaria del profesor p.
- $MIE \subseteq B$: Conjunto de patrones HS que poseen bloques horarios del día miércoles.
- $PC_i \subseteq B$: Conjunto de patrones HS que pueden ser utilizados por las cátedras del curso-sección i.
- $PA_i \subseteq B$: Conjunto de patrones HS que pueden ser utilizados por las clases auxiliares del curso-sección i.
- $PSC_{si} \subseteq B$: Conjunto de patrones HS que ocupan la sala s y pueden ser utilizados por las cátedras del curso-sección i.
- $PSA_{si} \subseteq B$: Conjunto de patrones HS que ocupan la sala s y pueden ser utilizados por las clases auxiliares del curso-sección i.

- $PTC_{ti} \subseteq B$: Conjunto de patrones HS que contienen al bloque horario t y pueden ser utilizados por las cátedras del curso-sección i.
- $PTA_{ti} \subseteq B$: Conjunto de patrones HS que contienen al bloque horario t y pueden ser utilizados por las clases auxiliares del curso-sección i.
- $PSTC_{sti} \subseteq B$: Conjunto de patrones HS que ocupan la sala s y el bloque horario t y pueden ser utilizados por las cátedras del curso-sección i.
- $PSTA_{sti} \subseteq B$: Conjunto de patrones HS que ocupan la sala s y el bloque horario t y pueden ser utilizados por las clases auxiliares del curso-sección i.
- $AX \subseteq I$: Conjunto de cursos-sección que poseen clases auxiliares.

Se definen dos conjuntos de variables binarias asociadas a la asignación de clases de cátedra y clases auxiliares a patrones HS, respectivamente. Estas variables son:

$$x_{ib} = \begin{cases} 1 & \text{si las clases de cátedra del curso-sección } i \text{ se realizan de acuerdo} \\ & \text{al patrón } b \in PC_i, \\ 0 & \text{en caso contrario;} \end{cases}$$

y

$$y_{ib} = \begin{cases} 1 & \text{si las clases auxiliares del curso-sección } i \text{ se realizan de acuerdo} \\ & \text{al patrón } b \in PA_i, \\ 0 & \text{en caso contrario;} \end{cases}$$

Con la notación definida, el modelo de programación entera es el siguiente:

$$\text{minimizar } z = \sum_{i \in I} \Big(\sum_{b \in L \setminus MIE} y_{ib} + \sum_{b \in PSA_{AUi}} y_{ib} \Big) + \sum_{i \in I} \sum_{b \in PSC_{AUi}} x_{ib}$$

sujeto a

$$\sum_{b \in PC_i} x_{ib} = 1 \qquad \text{para todo } i \in I\,, \tag{1}$$

$$\sum_{b \in PA_i} y_{ib} = 1 \qquad \text{para todo } i \in AX\,, \tag{2}$$

$$\sum_{i \in I} \sum_{b \in PSTC_{sti}} x_{ib} + \sum_{i \in I} \sum_{b \in PSTA_{sti}} y_{ib} \leq 1 \qquad \text{para todo } s \in S\,,\, t \in T\,, \tag{3}$$

$$\sum_{b \in PTC_{ti}} x_{ib} + \sum_{b \in PTA_{ti}} y_{ib} \leq 1 \qquad \text{para todo } i \in I\,,\, t \in T\,, \tag{4}$$

$$\sum_{i \in CPR_p} \sum_{b \in PTC_{ti}} x_{ib} \leq 1 \qquad \text{para todo } p \in P\,,\, t \in HR_p\,, \tag{5}$$

$$\sum_{i \in SMC_{hl}} \sum_{b \in PTC_{ti}} x_{ib} + \sum_{i \in SMC_{hl}} \sum_{b \in PTA_{ti}} y_{ib} \leq 1 \qquad \text{para todo } h \in H\,,\, l \in L\,,\, t \in T\,, \tag{6}$$

$$x_{ib} \in \{0,1\} \qquad \text{para todo } i \in I\,,\, b \in PC_i\,, \tag{7}$$

$$y_{ib} \in \{0,1\} \qquad \text{para todo } i \in AX\,,\, b \in PA_i\,. \tag{8}$$

La función objetivo representa la minimización de clases auxiliares asignadas a bloques horarios que no pertenecen al día miércoles más la cantidad de clases asignadas al Auditorio de la Facultad (AU).

Las restricciones (1) y (2) garantizan que para todos los cursos se asignen patrones que programen todas las clases de cátedra y auxiliares. La restricción (3) impide la asignación de más de un curso-sección a la misma sala en el mismo horario. La restricción (4) imposibilita que las clases de cátedra y auxiliares de un mismo curso sean programadas en el mismo horario. La restricción (5) evita que se programen en el mismo horario clases dictadas por el mismo profesor. Esta restricción no garantiza que las clases sean asignadas a bloques en que el profesor está disponible. Esto es controlado al definir el subconjunto PC_i de patrones HS que pueden ser utilizados para dictar las clases de cátedra del curso-sección i. Finalmente, la restricción (6) controla que no se produzcan topes de horarios entre clases de cursos diferentes que así lo requieran. En el caso particular de la aplicación considerada, se evitaron los topes entre cursos del mismo semestre en el plan de estudios de alguna de las carreras.

4. Resultados Experimentales

En esta sección se realiza una comparación de los resultados obtenidos entre el modelo propuesto y el sistema manual que actualmente utiliza la Facultad, para la generación de horarios y asignación de salas de clase. El objetivo central de esta sección es mostrar las ventajas de la utilización del modelo

matemático. Para esto se comparan distintos indicadores de desempeño. Estos indicadores miden el cumplimiento de las condiciones deseables y los objetivos de la Facultad.

Los indicadores utilizados fueron los siguientes:

I_1 : Porcentaje de clases auxiliares el día miércoles.

I_2 : Número de cursos asignados al Auditorio.

I_3 : Número cursos con horarios de cátedra sin patrón

I_4 : Número cursos asignados a salas de clase distintas para la realización de sus actividades.

I_5 : Número cursos asignados a salas con capacidad insuficiente.

I_6 : Número de pares de cursos asignados a una misma sala de clases.

I_7 : Número de semestres sin horarios factibles.

I_8 : Tiempo de generación de la programación.

El indicador I_3 se refiere a la existencia de cursos que no son dictados en los patrones definidos por la Facultad. El indicador I_7 cuenta los semestres que no tienen un horario factible. Esto quiere decir que, al considerar todos los cursos que componen a dicho semestre de acuerdo al plan de estudios, no existe una combinación de horarios que permita inscribirlos sin topes de horario.

4.1. Descripción de la Instancia

La instancia utilizada para la comparación de resultados fue suministrada por la Facultad de Ingeniería. Esta instancia está compuesta por 200 cursos, 150 profesores y 45 salas de clase.

El modelo de programación entera tiene 284.766 variables de decisión y 13.185 restricciones. Este modelo fue modelado utilizando la herramienta GAMS 22.5 y resuelto mediante el solver CPLEX 10.0 en un ordenador con procesador Intel Centrino Duo de 1.83 GHZ con 2 GB de memoria RAM.

4.2. Comparación de Resultados

La solución entregada por el sistema actual no cumple con las condiciones deseables para la operatibilidad normal de la institución. La Tabla 4.2 muestra los resultados obtenidos para la instancia descrita anteriormente.

Los dos primeros indicadores (I_1 y I_2) están relacionados directamente con la función objetivo del modelo. Existe una mejora significativa en la asignación de clases auxiliares para los días miércoles, pasando de un 43 % en el sistema

Indicadores	Sistema Actual	Modelo Propuesto
I_1	43 %	92 %
I_2	0	1
I_3	2	0
I_4	60	0
I_5	105	0
I_6	10	0
I_7	2	0
I_8	1 mes	2 min.

Cuadro 1: Comparación de resultados para las programaciones generadas por el modelo propuesto y el sistema actual.

actual a un 92 % obtenido por el modelo propuesto considerando el total de clases. Cabe destacar que este requerimiento nunca se podrá alcanzar en un 100 % debido a que la cantidad de salas de clase de la Facultad no puede albergar a todos los cursos que tienen clases auxiliares en un día. Respecto del segundo indicador, a pesar de que el sistema manual no genera asignaciones de cursos en el Auditorio, podemos señalar que trajo consigo que algunos cursos fueran asignados a salas de clase con capacidad insuficiente, ya que este Auditorio es la sala de clase de mayor capacidad.

Respecto a la condiciones deseables para la operatibilidad del sistema, el modelo propuesto asigna todos los cursos a un patrón establecido, no permite que un curso sea asignado a salas de clase diferentes, no permite la asignación de dos o más cursos a una misma sala de clase en un bloque horario y genera siempre al menos una combinación factible para los cursos de un semestre particular.

El sistema actual viola estas condiciones, ya que se obtienen 60 cursos asignados en salas de clase diferentes y 105 cursos asignados a una sala con capacidad insuficiente y asigna más de un curso a una misma sala de clases. Este tipo de ineficiencias causa problemas dentro del alumnado y, en especial, cuando no se respeta la capacidad de las salas de clase. Esto último en la práctica ocasiona que un número significativo de alumnos no pueda asistir a esos cursos.

Otro punto a destacar es que en la instancia estudiada para dos semestres de los planes de estudio no existe un horario factible para los distintos cursos que los componen. Esto significa que un alumno que curse dichos semestres, no podrá seleccionar todos sus cursos o, en su defecto, siempre tendrá algún tope de horario. El modelo propuesto proporciona siempre un horario factible permitiendo al alumnado inscribir todos los cursos correspondientes a un semestre especifico.

Finalmente un indicador muy importante a la hora de definir la programación de cursos y la asignación de salas de clase es el tiempo que demora dicho proceso. Las mejoras en este sentido son sustanciales, pasando desde un mes para el procedimiento manual a dos minutos con el modelo propuesto. Esta mejora permite no solo obtener una solución óptima en un corto tiempo, sino que es posible explorar múltiples escenarios, obtener soluciones alternativas y poder reaccionar ante eventos inesperados, como lo son la incorporación de cursos nuevos sobre la marcha.

En resumen, los beneficios reportados por el modelo propuesto son evidentes. La utilización de este modelo permitirá a la Facultad de Ingeniería eliminar todos los conflictos y, satisfacer todas las condiciones de operación, y además, ser eficiente en la asignación de un recurso escaso como lo son las salas de clase.

5. Conclusiones

Este trabajo presenta un modelo de programación entera para la programación de horarios y asignación de salas de clase para la Facultad de Ingeniería de la Universidad Diego Portales. Este modelo asigna simultáneamente todas las clases de cátedra o auxiliares de un curso a algún patrón horario-sala. Este enfoque es semejante al propuesto por Qualizza y Serafini [40] y se diferencia de otros modelos presentados en la literatura que asignan cada clase a bloques horarios por separado y fuerzan la formación de patrones a través de restricciones.

Considerando el tamaño del problema, la tarea de generar la programación de cursos manualmente se transforma en una labor en extremo compleja y sujeta a múltiples errores. La utilización del modelo propuesto garantiza la satisfacción de todos los requerimientos obligatorios y mejora en forma significativa el cumplimiento de las condiciones deseables. En particular, mientras la programación actual sólo consigue programar alrededor de un 40 % de las clases auxiliares el día miércoles, la solución obtenida por el modelo permite hacerlo con más del 90 %. Adicionalmente, se reduce sustancialmente el tiempo requerido para la obtención de la programación de horarios y asignación de salas de clase.

Actualmente, se está desarrollando un sistema computacional que utiliza como optimizador el modelo de programación entera propuesto e incorporará diferentes opciones mediante una interfaz amigable. Con la implementación del sistema computacional será posible automatizar el proceso de generación de la programación de horarios y asignación de salas aumentando la calidad de las programaciones, la flexibilidad ante cambios en los requerimientos y condiciones deseables y la posibilidad de explorar múltiples escenarios.

Una línea interesante para profundizar en el trabajo de investigación es la adaptación del modelo propuesto para su resolución mediante generación de columnas y la implementación de un algoritmo tipo *branch-and-price*.

Agradecimientos: Al Instituto Científico Milenio "Sistemas Complejos de Ingeniería" P04-066-F por el apoyo económico para la concreción de este proyecto. Adicionalmente, los autores agradecen la buena disposición del decano de la Facultad de Ingeniería, José Manuel Robles, al secretario de estudios de la Escuela de Ingeniería Industrial, Claudio Gutiérrez y, de manera especial, a Francia Lara I. por sus valiosos comentarios para la edición final de este documento.

Referencias

[1] S. Abdennadher and M. Marte. University course timetabling using constraint handling rules. *Applied Artificial Intelligence*, 14:311–325, 2000.

[2] S. Al-Yakoob and S. Sherali. Mathematical programming models and algorithms for a class-faculty assignment problem. *European Journal of Operational Research*, 173:488–507, 2006.

[3] S. Al-Yakoob and S. Sherali. A mixed-integer programming approach to a class timetabling problem: A case study with gender policies and traffic considerations. *European Journal of Operational Research*, 180:1028–1044, 2007.

[4] R. Alvarez-Valdes, E. Crespo, and J. Tamarit. Design and implementation of a course scheduling system using tabu search. *European Journal of Operational Research*, 137:512–523, 2002.

[5] A. Asratian and D. de Werra. A generalized class-teacher model for some timetabling problems. *European Journal of Operational Research*, 143:531–542, 2002.

[6] P. Avella and L. Vasil'ev. A computational study of a cutting plane algorithm for university course timetabling. *Journal of Scheduling*, 8(6):497–514, 2004.

[7] M. Badri. A two-stage multiobjective scheduling model for [faculty-course-time] assignments. *European Journal of Operational Research*, 94:16–28, 1996.

[8] M. Badri, D.L. Davis, D.F. Davis, and J. Hollingsworth. A multi-objective course scheduling model: Combining faculty preferences for courses and times. *Computers and Operations Research*, 25:303–316, 1998.

[9] N. Boland, B. Hughes, L. Merlot, and P. Stuckey. New integer linear programming approaches for course timetabling. *Computers and Operations Research*, 35:2209–2233, 2008.

[10] E. Burke and M. Carter, editors. *The Practice and Theory of Automated Timetabling II: Selected Papers from the 2nd International Conference on the Practice and Theory of Automated Timetabling*. LNCS 1408, Springer-Verlag, 1998.

[11] E. Burke and P. De Causmaecker, editors. *The Practice and Theory of Automated Timetabling IV:Selected papers from the 4th International Conference (PATAT IV)*. LNCS 2740, Springer-Verlag, 2003.

[12] E. Burke and W. Erben, editors. *Practice and Theory of Automated Timetabling:Selected papers from the 3rd International Conference*. LNCS 2079, Springer-Verlag, 2001.

[13] E Burke, K Jackson, JH Kingston, and R Weare. Automated university timetabling: The state of the art. *Computer Journal*, 40(9):565–571, 1997.

[14] E. Burke, B. MacCarthy, S. Petrovic, and R. Qu. Knowledge discovery in a hyper-heuristic for course timetabling using case-based reasoning. In Burke and Causmaecker [11], pages 276–287.

[15] E. Burke, B. McCollum, A. Meisels, S. Petrovic, and R. Qu. A graph-based hyper-heuristic for educational timetabling problems. *European Journal of Operational Research*, 176:177–192, 2007.

[16] E. Burke and H. Rudová, editors. *PATAT 2006: Proceedings of The 6th International Conference on the Practice and Theory of Automated Timetabling*, Masaryk University, Brno, República Checa, 2006. Disponible en la página `http://patat06.muni.cz/doc/PATAT_2006_Proceedings.pdf`.

[17] E. Burke and M. Trick, editors. *The Practice and Theory of Automated Timetabling V: Selected Revised Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling*. LNCS 3616, springer-Verlag, 2005.

[18] M. Carter. A comprehensive course timetabling and student scheduling system at the university of waterloo. In Burke and Erben [12], pages 64–82.

[19] L. C. Chambers, editor. *The Practical Handbook of Genetic Algorithms*, volume 1. CRC Press, 1995.

[20] M. Chiarandini, M. Birattari, K. Socha, and O. Rossi-Doria. An effective hybrid algorithm for university course scheduling. *Journal of Scheduling*, 9:403–432, 2006.

[21] D. Corne, P. Ross, and H. Fang. *Evolving timetables*, pages 219–276. Volume 1 of Chambers [19], 1995.

[22] S. Daskalaki and T. Birbas. Efficient solutions for a university timetabling problem through integer programming. *European Journal of Operational Research*, 160:106–120, 2005.

[23] S. Daskalaki, T. Birbas, and E. Housos. An integer programming formulation for a case study in university timetabling. *European Journal of Operational Research*, 153:117–135, 2004.

[24] M. Dimopoulou and P. Miliotis. Implementation of a university course and examination timetabling system. *European Journal of Operational Research*, 130:202–213, 2001.

[25] M. Dimopoulou and P. Miliotis. An automated university course timetabling system developed in a distributed environment: A case study. *European Journal of Operational Research*, 153:136–147, 2004.

[26] M. Elmohamed, P. Coddington, and G. Fox. A comparison of annealing techniques for academic course scheduling. In Burke and Carter [10], pages 92–112.

[27] J. Ferland and C. Fleurent. Saphir - a decision-support system for course scheduling. *Interfaces*, 24:105–115, 1994.

[28] L. Foulds and D. Jonhson. Slotmanager: a microcomputer-based decision support system for university timetabling. *Decision Support Systems*, 27:367–381, 2000.

[29] L. Di Gaspero and A. Schaerf. Multi-neighbourhood local search with application to course timetabling. In Burke and Causmaecker [11], pages 262–275.

[30] H. Goltz and D. Matzke. University timetabling usin constraint logic programming. In *Practical Aspects of Declarative Languages*, volume 1551 of *LNCS*, pages 320–334. Springer-Verlag, 1999.

[31] K. Haase, H. Scheel, and D. Sebastian. Management of lecture-rooms. model, method and intenet-based application for efficient course scheduling. *Wirtschaftsinformatik*, 46:87–95, 2004.

[32] T. Hinkin and G. Thompson. Schedulexpert: Scheduling courses at cornell university school of hotel administration. *Interfaces*, 32:45–57, 2002.

[33] R. Lewis. A survey of metaheuristics-based techniques for university timetabling problems. *OR Spectrum*, 30:167–190, 2008.

[34] B. McCollum. The implementation of a central timetabling system in a large british civic university. In Burke and Carter [10], pages 237–253.

[35] S. MirHassani. A computational approach to enhancing course timetabling with integer programming. *Applied Mathematics and Computation*, 175:814–822, 2006.

[36] E. Mooney, R. Rardin, and W. Parmenter. Large-scale classroom scheduling. *IIE Transactions*, 28:369–378, 1996.

[37] M. Oprea. Mas_up-uct: A multi-agent system for university course timetabling scheduling. *International Journal of Computer Communications and Control*, 2:94–102, 2007.

[38] B. Paetcher, R. Rankin, and A. Cumming. Improving a lecture timetabling system for university wide-use. In Burke and Carter [10], pages 156–165.

[39] P. Pongcharoen, W. Promtet, P. Yenradee, and C. Hicks. Stochastic optimisation timetabling tool for university course scheduling. *International Journal of Production Economics*, 112:903–918, 2008.

[40] A. Qualizza and P. Serafini. A column generation scheme for faculty timetabling. In Burke and Trick [17], pages 161–173.

[41] P. Rattadilok, A. Gaw, and R. Kwan. Distributed choice function hyperheuristic for timetabling and scheduling. In Burke and Trick [17], pages 51–67.

[42] O. Rossi-Doria, M. Sampels, M. Birattari, M. Chiarandini, M. Dorigo, L. Gambardella, J. Knowles, M. Manfrin, M. Mastrolilli, B. Paetcher, L. Paquete, and T. Stutzle. A comparison of the performance of different metaheuristics on the timetabling problem. In Burke and Causmaecker [11], pages 329–351.

[43] H. Rudova and K. Murray. University course timetabling with soft constraints. In Burke and Causmaecker [11], pages 310–328.

[44] K. Sandhu. *Automating class schedule generation in the context of a university timetabling information system*. PhD thesis, School of Managementm, Griffith University, 2001.

[45] A. Schaerf. A survey of automated timetabling. *Artificial Intelligence Review*, 13:87–127, 1999.

[46] K. Socha, M. Sampels, and M. Manfrin. Ant algorithm for the university course timetabling problem with regard to the state-of-the-art. In *Applications of evolutionary computing*, volume 2611 of *LNCS*, pages 334–345. Springer-Verlag, 2003.

[47] J. Stallaert. Automated timetabling improves course scheduling at UCLA. *Interfaces*, 27(4):67–81, 1997.

[48] D. Strnad and N. Guid. A multi-agent system for university timetabling. *Applied Artificial Intelligence*, 21:137–153, 2007.

[49] A. Tripathy. School timetabling – A case in large binary integer linear programming. *Management Science*, 30(12):1473–1489, 1984.

[50] Y. Wang. Usin genetic algorithm methods to solve course scheduling problems. *Expert Systems with Applications*, 25:39–50, 2003.

[51] G. White and J. Zhang. Generating complete university timetables by combining tabu search with constraint logic. In Burke and Carter [10], pages 187–198.