UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

DESARROLLO DE UNA HERRAMIENTA PARA LA CALENDARIZACIÓN DE EVALUACIONES UNIVERSITARIAS

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN COMPUTACIÓN

ÁLVARO RODRIGO YÁÑEZ MENESES

PROFESOR GUÍA:
FEDERICO OLMEDO BERÓN

MIEMBROS DE LA COMISIÓN:
GONZALO NAVARRO BADINO
EDUARDO GODOY VEGA

SANTIAGO DE CHILE
2020

Resumen

La carga académica a la que están sometidos los estudiantes ha sido un tema bastante polémico en los últimos años, y la Facultad de Ciencias Físicas y Matemáticas de la Universidad de Chile no es la excepción. En el último tiempo ha habido paros estudiantiles por este mismo motivo, y la Facultad ha empezado a tomar medidas en pos de mejorar esta situación.

Gran parte de este problema que aqueja a los estudiantes es debido a una "mala organización" de las evaluaciones de los distintos ramos que forman un semestre académico. Esto ocurre porque los profesores de cada ramo fijan las fechas de sus evaluaciones de manera independiente –con una visión local–, sin considerar las fechas de las evaluaciones de los otros ramos que componen el semestre, generando así una distribución de carga académica muy desbalanceada –y muchas veces, inviable– para los estudiantes.

Para afrontar dicho problema, en este trabajo se desarrolló una herramienta que permite ordenar de una manera más razonable e informada la carga académica de los distintos semestres de la malla curricular de una carrera de nuestra Facultad. Concretamente, esta herramienta produce una calendarización óptima de las evaluaciones de cada semestre, donde con *calendarización* nos referimos a la asignación de una fecha concreta para cada evaluación.

Para obtener dicha calendarización óptima se modela el problema como uno de optimización matemática, formado por un conjunto de restricciones y una función objetivo que operan sobre el espacio de todas las calendarizaciones posibles. Las restricciones del modelo descartarán aquellas calendarizaciones que concentren mucha carga académica en un periodo corto de tiempo y la función objetivo primará aquellas calendarizaciones que fijan sus evaluaciones lo más pronto posible pasada la materia correspondiente.

La herramienta permite además reportar la cantidad de calendarizaciones (sub)óptimas que el usuario desee. De esta manera, se puede comparar las mejores calendarizaciones para seleccionar la más conveniente. Para realizar dicha selección de manera informada, la herramienta crea un ranking de las calendarizaciones reportadas en base a 3 criterios distintos: varianza de la carga académica de cada semana del semestre, varianza de la carga académica de cada día del semestre y cantidad de evaluaciones fijadas en días "no deseados".

Para evaluar la solución propuesta, se consideraron los semestres V, VI y VII de la malla de Ingeniería Civil en Computación de nuestra Facultad, se generaron calendarizaciones óptimas utilizando la herramienta y se compararon las mismas con respecto a calendarizaciones reales que tuvieron dichos semestres en 2016/1, 2016/2 y 2017/1, respectivamente. La comparación basada en un total de nueva variables cuantitativas arroja una mejora sustancial en la distribución de cargas obtenida.

Agradecimientos

Quiero comenzar agradeciendo al profesor Federico por apoyarme y ayudarme durante todo el proceso de mi memoria. A mis amigos que me acompañaron desde que comencé este viaje universitario y en los cuales muchas veces me apoyé para seguir adelante. A mi familia, ya que sin ellos no habría llegado a donde estoy hoy en día. Por último, agradecer a Bárbara que me ayudó mucho en la parte emocional y siempre me motivó a avanzar con mis estudios y finalmente con mi memoria.

Tabla de Contenido

Índice de Tablas

Índice de Ilustraciones

Capítulo 1

Introducción

1.1. Contexto

La calidad de vida en Chile es un asunto que ha estado muy presente en el país durante el último tiempo, tanto es así que para mejorarla se ha impulsado un proyecto de ley que busca reducir las jornada laboral de 45 horas semanales a 40 horas [4].

Una de las causas que contribuye directamente a esta baja en la calidad de vida es el estrés. Según un estudio realizado en 2016, el 42 % de los chilenos dice estar altamente estresado, en contraste con el mismo estudio realizado en el año 2012 donde solo el 22 % de la población consideraba que sufría el mismo escenario [11]. El estrés es una reacción fisiológica del organismo, un mecanismo de autodefensa, que el cuerpo y la mente disponen para sobrevivir, por lo que en sí el estrés no es malo. El problema viene cuando el estrés es excesivo y termina generando deterioros en la salud mental y física de las personas.

De la mano de esto, se desató la preocupación por el estrés que genera la carga académica de los estudiantes en diversas universidades, lo que quedó en evidencia el año 2019 con el reclamo iniciado por los estudiantes de la Facultad de Arquitectura y Urbanismo de la Universidad de Chile [7], que luego se extendió a muchas otras casas de educación.

En la Facultad de Ciencias Físicas y Matemáticas de la Universidad de Chile este problema de la carga académica no se queda atrás. Los estudiantes se quejan habitualmente de tener muchas evaluaciones en un corto plazo, lo que les deja poco tiempo de estudio, teniendo que postergar sus horas de sueño para optar por una mejor nota o para terminar una tarea. A veces deben incluso priorizar un ramo sobre otro, teniendo que escoger qué evaluación dar y qué evaluación saltarse.

Este estrés al que se encuentran sometidos los estudiantes afecta no sólo su desempeño académico, sino también su vida personal, teniendo poco tiempo para realizar las actividades de las que disfrutan y de esta manera bajar su nivel de estrés. Esto termina por generar un círculo vicioso del cual no se puede salir y que se acentúa cada vez más.

1.2. Problema

Gran parte de este problema se debe a una distribución deficiente de las evaluaciones de los distintos ramos que forman un semestre académico. A su vez, esto se debe principalmente a la ausencia de un mecanismo institucional que vele por que las evaluaciones sean distribuidas de una manera razonable y viable para los estudiantes. En la práctica, cada profesor calendariza los distintos hitos de evaluación de sus ramos de una manera ad hoc e independiente, considerando sólo las necesidades de su ramo, o dicho de otra manera, sin tener en cuenta cómo sus evaluaciones interactúan con las evaluaciones de los demás ramos del semestre.

Esta mirada local que tiene cada profesor al calendarizar –asignarle una fecha concreta a– sus evaluaciones resulta en una carga académica muy desbalanceada para los estudiantes, caracterizada por la presencia de periodos con una carga simplemente inviable para los estudiantes. Por ejemplo, la presencia de semanas con 4 evaluaciones es una situación bastante habitual.

Para abordar dicho problema se vuelve necesario diseñar un mecanismo *sistemático* y *bien informado* para la calendarización de evaluaciones semestrales. El desarrollo y adopción del mismo ayudará tanto a estudiantes, a balancear su carga académica, como a profesores, a fijar sus evaluaciones de una manera más informada y menos ad hoc.

1.3. Objetivos

En vista de lo anterior, los objetivos del presente trabajo de fin de carrera son:

Objetivo general: desarrollar una herramienta que permita distribuir las evaluaciones de un semestre, respetando los requisitos de cada ramo y generando, a la vez, una distribución de carga razonable y viable para los estudiantes.

Objetivos específicos:

1. Definir un modelo matemático que permita:
   (a) capturar convenientemente la noción de calendarización de evaluaciones y el conjunto de restricciones impuesto al respecto por los profesores;
   (b) cuantificar la calidad de una calendarización dada en términos de la carga académica que conlleva.
2. Desarrollar un algoritmo que, valiéndose de dicho modelo, compute las calendarizaciones óptimas de un semestre, para un conjunto dado de ramos y restricciones de sus hitos de evaluaciones.
3. Desarrollar una herramienta que implemente la solución propuesta.
4. Validar la solución propuesta a partir de la calendarización óptima generada para 3 semestres de la malla de Ingeniería Civil en Computación.

En particular, vamos a considerar solo los ramos obligatorios de estos 3 semestres, ya que los

ramos electivos se eligen de un *pool* de ramos bastante grande, lo que hace que el conjunto total de ramos que toma cada estudiante en un semestre varíe significativamente de estudiante a estudiante.

1.4. Solución propuesta

Para abordar este problema, desarrollamos una herramienta que dada la siguiente entrada, genera la siguiente salida:

Entrada:

- conjunto de ramos que se desea calendarizar. En nuestro casos van a ser los ramos obligatorios de un semestre de la malla de Ingeniería Civil en Computación;
- para cada ramo, el conjunto de sus evaluaciones. En particular, para cada evaluación ingresaremos su tipo (por ejemplo, control o tarea), posibles fechas en las que puede ser calendarizada y su tiempo de preparación (medido en horas);
- parámetros que determinan cuándo una calendarización es viable. Más concretamente, número máximo de horas (de estudio) permitido por día y por semana;
- datos generales sobre el semestre calendario, por ejemplo, 2020/2, donde se implementará la calendarización. Concretamente, días feriados y días donde se sugiere no fijar evaluaciones (por ejemplo, semana de las Jornadas Chilenas de Computación).

Salida: Fecha concreta dentro del semestre para cada evaluación ingresada.

Para devolver la calendarización óptima, la herramienta modela el problema como uno de programación lineal entera, formado por un conjunto de restricciones y una función objetivo que operan sobre el espacio de todas las calendarizaciones posibles. Las restricciones del modelo descartará aquellas calendarizaciones a) que no sean válidas, por ejemplo, porque no asignan fecha a alguna de las evaluaciones, o b) que no sean viables porque generan una carga (medida en horas totales) mayor a la permitida en una semana o día dado. Por su parte, dentro de todas las calendarizaciones válidas y viables, la función objetivo permitirá "seleccionar" aquella(s) calendarizacion(es) que fija(n) sus evaluaciones lo más pronto posible pasada la materia correspondiente. Dicho de otra manera, de todas las calendarizaciones válidas que generan una carga académica razonable, se preferirá aquella(s) que no posterga(n) o retrasa(n) las evaluaciones de manera innecesaria.

La herramienta permitirá reportar la cantidad de calendarizaciones (sub)óptimas que el usuario desee. De esta manera, se podrán comparar las mejores calendarizaciones para seleccionar la más conveniente. Para realizar dicha selección de manera informada, la herramienta permitirá ordenar las calendarizaciones reportadas en base a 3 criterios distintos: varianza de la carga académica de cada semana del semestre, varianza de la carga académica de cada día del semestre y cantidad de evaluaciones fijadas en días "no deseables".

Para que esta herramienta pueda realizar su tarea, primero se deberá tener una interfaz que reciba los datos de entrada. Esta interfaz, web, será programada en React.js. Una vez obtenidos los datos de entrada asociados al semestre que se desea calendarizar, la herramienta

generará el programa lineal entero asociado y lo resolverá utilizando el método *branch-and-cut*, provisto por la librería MIP de Python, lenguaje en el que se programará el *backend* herramienta.

Capítulo 2

Marco Teórico

En esta sección vamos a repasar las nociones y técnicas de programación lineal que usamos para desarrollar nuestra solución (Secciones 2.1 y 2.2) y el trabajo relacionado existente (Sección 2.3).

2.1. Programación lineal entera

Programación matemática. Informalmente, el objetivo de la *optimización* o *programación matemática* es encontrar la solución óptima a un problema dado. Formalmente, una problema de programación matemática tiene la forma

$$\begin{aligned} &\text{mín/máx } f(\bar{x}) \\ &\text{s.a.} \quad \bar{x} \in X \ , \end{aligned}$$

donde $X \subseteq \mathbb{R}^n$ representa el *espacio de soluciones* posibles del problema y $f : X \longrightarrow \mathbb{R}$ es la *función objetivo* que se desea optimizar. Normalmente se busca determinar no sólo el valor óptimo de f sino también el (los) punto(s) $x^* \in X$ donde este se alcanza.

Programación lineal. Un caso particular de la programación matemática es la *programación lineal* donde el espacio de soluciones (usualmente llamadas *factibles*) X está dado por la conjunción de un número de restricciones lineales sobre $\bar{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$ y la función objetivo f es una función lineal sobre $\bar{x}$. Formalmente, un problema de programación lineal tiene la forma

$$\begin{aligned} &\text{mín } z = \textstyle\sum_{j=1}^{n} c_j x_j \\ \text{s.a.} \quad &\textstyle\sum_{j=1}^{n} a_{\mathrm{i}j} x_j \leq b_{\mathrm{i}} && \forall\, \mathrm{i} = 1, \ldots, m \\ &x_j \geq 0 && \forall\, j = 1, \ldots, n \ , \end{aligned}$$

donde $a_{\mathrm{i}j}, b_{\mathrm{i}}, c_j, \in \mathbb{R}$ para todo $\mathrm{i} = 1, \ldots, m$ y $j = 1, \ldots, n$. Esta es la forma *canónica* de un programa lineal y cualquier otro programa lineal puede transformarse en uno equivalente en forma canónica. Por ejemplo, si una de las restricciones lineales tiene la forma de una igualdad en vez de una desigualdad, esta puede reescribirse como la conjunción de otras dos desigualdades. Si una variable no está restringida a valores positivos, puede reescribirse como

la diferencia de otras dos variables que sí lo están. Por fines didácticos, en el resto de la presentación nos permitiremos el uso de programas que no están en forma canónica.

Programación entera y binaria. De acuerdo a la naturaleza del problema que estamos modelando, a veces se requiere que las variables x_j tomen valores enteros en vez de reales. Esto es lo que se conoce como un problema de *programación entera*, y tiene la misma forma que un programa lineal, al que se le agrega la restricción

$$x_j \in \mathbb{Z} \qquad \forall\, j = 1, \ldots, n.$$

Para modelar nuestro problema de calendarización óptima vamos a usar un tipo particular de programación entera, donde

$$x_j \in \{0, 1\} \qquad \forall\, j = 1, \ldots, n.$$

Este tipo de problemas se conocen como de *programación binaria* y son particularmente útiles para modelar problemas de decisiones basados en respuestas binarias si/no.

2.2. Resolución de programas enteros

Ahora vamos a explorar los distintos algoritmos de resolución de los problemas de programación entera.

Dificultad de los programas enteros

En la actualidad se conocen diversos algoritmos que permiten resolver programas lineales de manera eficiente. El más tradicional es el *método Simplex*, pero existen otros como el *método del elipsoide* o el *algoritimo de Karmarkar* [10]. Sin embargo, ninguno de estos métodos pueden utilizarse (de manera directa) para resolver programas enteros: resolver un programa entero como si fuera uno lineal y posteriormente redondear las variables reales a enteros no garantiza el óptimo del programa entero.

Para ilustrar este fenómeno, consideremos el programa entero a la izquierda de la Figura 2.1. A su derecha puede observarse el conjunto de soluciones factibles del programa: las líneas azules delimitan el conjunto de soluciones factibles del programa lineal asociado, es decir, sin la restricción $x_1, x_2 \in \mathbb{Z}$, y los vértices de las cuadriculas en la sección azul representan las soluciones del programa entero.

La solución óptima del programa lineal se alcanza en el punto (1.8, 2.8), y tiene un valor de 2.8. Ahora bien, si redondeáramos las coordenadas de este punto para obtener el óptimo del programa entero, obtendríamos el punto $(2, 3)$, que como se puede apreciar en la figura, está directamente fuera del conjunto de soluciones factibles. A manera de observación, el óptimo del programa entero se alcanza en los puntos $(1, 2)$ y $(2, 2)$ y tiene un valor de 2.

En general, los programas enteros son más difíciles de resolver que los lineales. Es por esto que a lo largo de los años se han desarrollado otras técnicas para resolver esta clase de programas de manera satisfactoria. La más usada es el técnica *branch-and-cut*, que consite en la combinación de los métodos *branch-and-bound* y *cutting-plane*.

$$\begin{aligned} &\text{máx } x_2 \\ \text{s.a. } &\begin{cases} -x_1 + x_2 \leq 1 \\ 3x_1 + 2x_2 \leq 12 \\ 2x_1 + 3x_2 \leq 12 \end{cases} \\ &x_1, x_2 \geq 0 \\ &x_1, x_2 \in \mathbb{Z} \end{aligned}$$

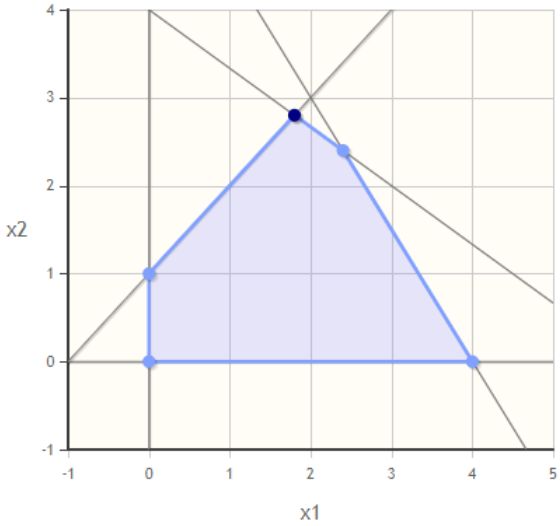

Figura 2.1: Programa entero (izquierda), junto al conjunto de soluciones factibles de su versión relajada (derecha).

Branch-and-bound

Para ilustrar el funcionamiento de esta técnica consideraremos el siguiente programa entero tomado de [16]:

$$\begin{aligned} &\text{máx } 4x_1 + 6x_2 \\ \text{s.a. } &\begin{cases} 2x_1 + 4x_2 \leq 12 \\ 4x_1 + 3x_2 \leq 16 \end{cases} \\ &x, y \geq 0 \\ &x, y \in \mathbb{Z} \end{aligned}$$

Comenzamos resolviendo la versión *relajada* del programa, donde x_1, x_2 toman valores en $\mathbb{R}$ en vez de en $\mathbb{Z}$; llamaremos P_0 a este programa. En la Figura 2.2 observamos que el óptimo de P_0 se alcanza en el punto de coordenadas $x_1 = 2.8$ y $x_2 = 1.6$, y tiene un valor de 20.8.

Claramente esta solución no cumple con las restricciones del problema entero inicial, por lo que dividiremos el problema en dos subproblemas. Para ello, elegimos alguna de las variables que tiene valor fraccionario, digamos x_1, y particionamos su dominio en dos, a partir de su piso $\lfloor x_1 \rfloor = 2$. Esto genera dos nuevos subproblemas, P_1 y P_2, que se obtienen a partir de P_0 agregando respectivamente las restricciones $x_1 \leq 2$ y $x_1 \geq 3$. A este proceso se lo conoce con el nombre de *ramificación* y consiste básicamente en particionar el conjunto de soluciones factibles en dos, y analizar cada uno de los subconjuntos generados por separado.

En la Figura 2.3 podemos ver que el óptimo de P_1 se alcanza con $x_1 = 2$ y $x_2 = 2$, y tiene un valor de 20. Cabe destacar que el valor óptimo de P_0 (20.8) es mayor que el valor de P_1 (20), lo que es esperable ya que el espacio de soluciones factibles de P_1 es un subconjunto del espacio de soluciones de P_0.

$$\text{máx } 4x_1 + 6x_2$$
$$\text{s.a.} \quad \begin{cases} 2x_1 + 4x_2 \leq 12 \\ 4x_1 + 3x_2 \leq 16 \end{cases}$$
$$x, y \geq 0$$

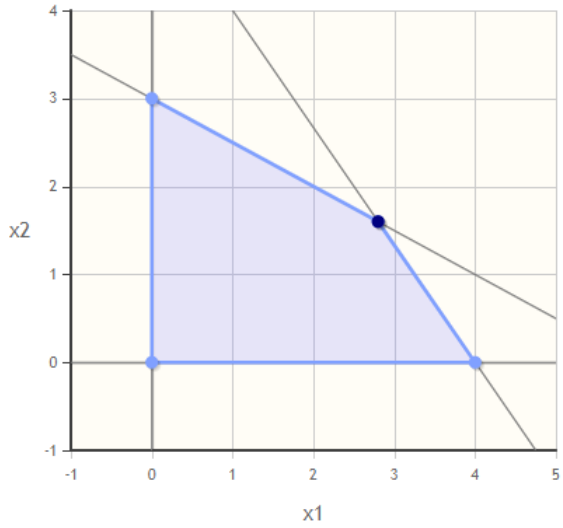

Figura 2.2: Programa relajado P_0 (izquierda) junto a su conjunto de soluciones factibles (derecha).

$$\text{máx } 4x_1 + 6x_2$$
$$\text{s.a.} \quad \begin{cases} 2x_1 + 4x_2 \leq 12 \\ 4x_1 + 3x_2 \leq 16 \\ x_1 \leq 2 \end{cases}$$
$$x, y \geq 0$$

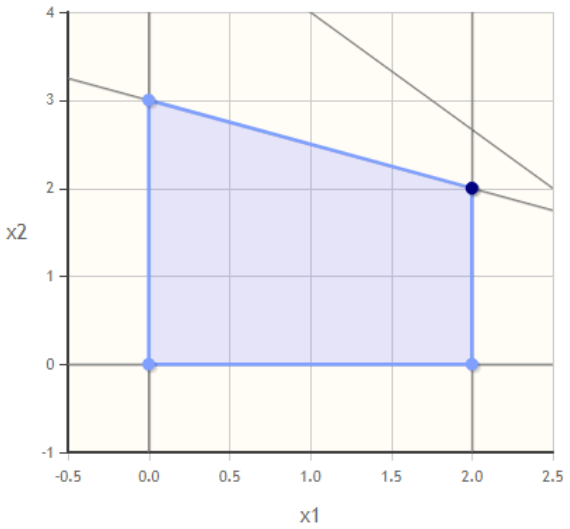

Figura 2.3: Programa P_1 (izquierda) junto a su conjunto de soluciones factibles (derecha).

Análogamente, en la Figura 2.4 observamos que el óptimo de P_2 se alcanza en el punto de coordenadas $x_1 = 3$ y $x_2 = \frac{4}{3}$, y tiene un valor de 20.

Observe que a través de P_1 se ha encontrado una solución óptima entera a un subproblema del problema relajado original. Esta solución óptima entera tiene un valor de 20. Por otro lado observe que no tiene sentido seguir ramificando P_1 ya que todos los subproblemas generados a partir de él tendrán un óptimo menor (o igual) a 20, ya que sus conjuntos de soluciones factibles estarán contenidos en el conjunto de soluciones factibles de P_1. Por el mismo motivo, cualquier subproblema generado a partir de P_2 tendrá un valor óptimo menor o igual al de P_2, que también es 20. Por tanto, el óptimo entero del problema original es 20 y se alcanza en el punto (óptimo de P_1) $x_1 = x_2 = 2$.

$$\text{máx } 4x_1 + 6x_2$$
$$\text{s.a.} \quad \begin{cases} 2x_1 + 4x_2 \leq 12 \\ 4x_1 + 3x_2 \leq 16 \\ x_1 \geq 3 \end{cases}$$
$$x, y \geq 0$$

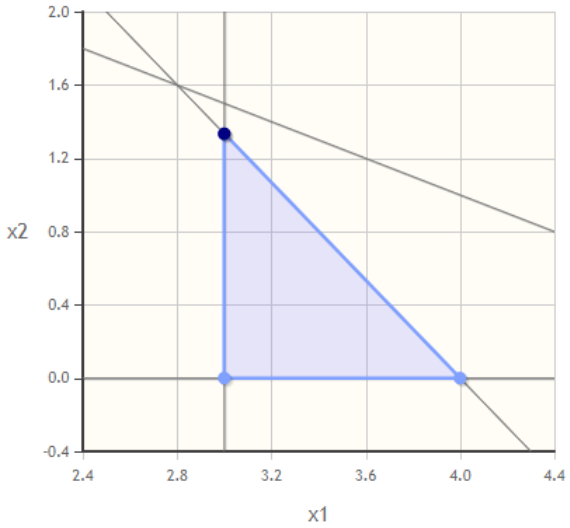

Figura 2.4: Programa P_2 (izquierda) junto a su conjunto de soluciones factibles (derecha).

Si por el contrario el óptimo de P_2 hubiese sido superior a 20, digamos 20.5, hubiese sido necesario seguir ramificando P_2 para ver si existía una solución entera con valor en el intervalo (20, 20.5].

En resumen, podemos entender el método *branch-and-bound* como que procede generando un árbol de búsqueda binario, donde en cada nodo se resuelve una versión relajada del programa original, y cada ramificación divide el espacio de soluciones factibles en dos. Para resolver cada versión relajada, usamos cualquiera de los métodos tradicionales de programación linear, por ejemplo, el algoritmo *simplex*.

Si bien ilustramos el método para el caso de programas enteros, el mismo se adapta muy fácilmente al caso de programas binarios. En cada ramificación, basta con particionar el dominio de la variable escogida, digamos x_k, a partir de las ecuaciones $x_k = 0$ y $x_k = 1$. Además, para relajar el programa entero original debemos reemplazar cada restricción $x_j \in \{0, 1\}$ por $0 \leq x_j \leq 1$.

Cutting-planes

Para resolver un programa entero, este método procede resolviendo una secuencia de relajaciones, de la siguiente manera:

1. Resuelve la versión relajada del problema entero original.
2. Si la solución encontrada $\bar{x}$ es entera, se detiene y devuelve $\bar{x}$.
3. Si no, genera un *corte* al problema, es decir, una restricción que es cumplida por todas las soluciones enteras, pero no por $\bar{x}$.
4. Añade esta nueva restricción al programa, lo resuelve, y vuelve al punto 2.

Para ilustrar esta técnica, consideremos el programa entero de la Figura 2.5, tomado de [13]. A la derecha de la figura se muestra el espacio de soluciones de la versión relajada del

programa. La solución óptima se alcanza en el punto de coordenadas $x_1 = 2.25$ y $x_2 = 3.75$, ambas fraccionarias.

$$\begin{aligned} &\text{máx } 5x_1 + 8x_2 \\ \text{s.a. } &\begin{cases} x_1 + x_2 \leq 6 \\ 5x_1 + 9x_2 \leq 45 \end{cases} \\ &x_1, x_2 \geq 0 \\ &x_1, x_2 \in \mathbb{Z} \end{aligned}$$

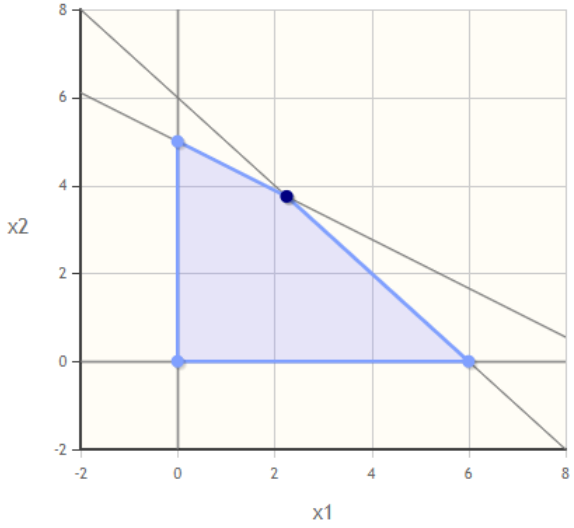

Figura 2.5: Programa entero (izquierda) y conjunto de soluciones factibles del programa relajado (derecha).

Como la solución óptima no es entera (Punto 3), se prosigue generando un corte, digamos $2x_1 + 3x_2 \leq 15$, y se añade a las restricciones del problema, creando un nuevo programa P_1 que se muestra en la Figura 2.6. Observe que el corte "contiene" a todas las soluciones enteras del problema original, y "deja afuera" al óptimo no entero $(x_1, x_2) = (2.25, 3.75)$.

$$\begin{aligned} &\text{máx } 5x_1 + 8x_2 \\ \text{s.a. } &\begin{cases} x_1 + x_2 \leq 6 \\ 5x_1 + 9x_2 \leq 45 \\ 2x_1 + 3x_2 \leq 15 \end{cases} \\ &x_1, x_2 \geq 0 \end{aligned}$$

Figura 2.6: Programa P_1 (izquierda) junto al conjunto de soluciones factibles (derecha).

Al resolver este nuevo programa (Punto 4), se obtiene una solución óptima $(x_1, x_2) = (3,3)$. Como es una solución entera el algoritmo termina, devolviendo esta como solución óptima del programa entero inicial (Punto 2).

En la Figura 2.7 se muestra en azul el subconjunto de soluciones que son eliminadas al agregar el corte. Notar que solo se eliminan soluciones no enteras.

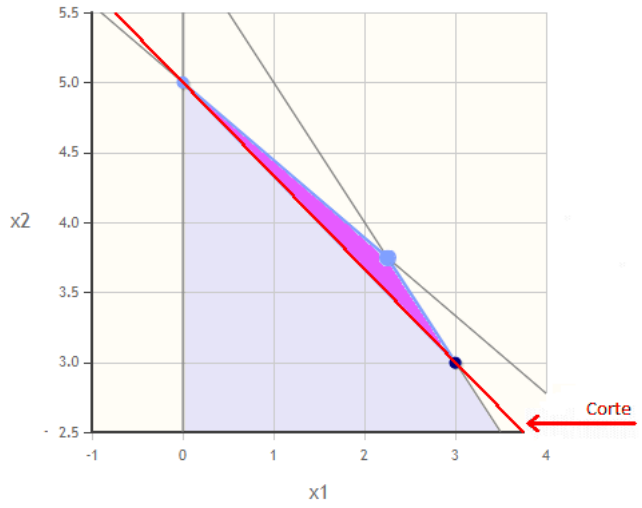

Figura 2.7: Subconjunto de soluciones factibles eliminadas (área violeta) al agregar el corte $2x_1 + 3x_2 \leq 15$ (linea roja) al problema P_0.

Como se explicó anteriormente, para que un corte sea significativo necesita que el actual óptimo del problema sea removido del espacio de soluciones, sin quitar ninguna de las posibles soluciones enteras.

Para la generación de estos cortes, la técnica más utilizada consiste en el uso de *cortes de Gomory* [2]. El resultado en el que se apoya esta técnica es que si

$$\sum_{j=1}^{n} a_{ij} x_j \ \leq \ b_i$$

es una restricción del programa original con b_i no entero, entonces la incorporación de la nueva restricción

$$\sum_{j=1}^{n} (a_{ij} - \lfloor a_{ij} \rfloor) x_j \ \geq \ b_i - \lfloor b_i \rfloor$$

mantiene las soluciones enteras del programa relajado original y elimina (al menos) un vértice del poliedro que representa su conjunto de soluciones. Como el óptimo se alcanza en uno de estos vértices, la restricción posiblemente lo elimina. En cuanto más restricciones de este tipo se agregan, mayor es el potencial de eliminarlo. En la práctica, este método se aplica junto al algoritmo Simplex (para resolver el programa relajado), que parte transformando el programa original en uno equivalente (llamado en forma *estándar*) donde la mayoría de los coeficientes resultan fraccionarios, lo que "prepara" al programa para el uso de cortes de Gomory.

Branch-and-cut

Este método es una combinación de los métodos *branch-and-bound* y *cutting-planes* recién presentados. Más concretamente, puede entenderse como una versión refinada del método

branch-and-bound donde al resolver cada (sub)problema relajado, si la solución óptima hallada no es entera, antes de proceder a realizar una ramificación se permite agregar uno o varios cortes y verificar si la solución de este nuevo programa es entera. Si ese es el caso, ya no es necesario generar la ramificación, dando lugar a una potencial mejora de eficiencia. Si por el contrario la solución de este nuevo programa sigue siendo fraccionaria, se procede a generar la ramificación.

2.3. Trabajo relacionado

Informalmente, el problema de organización de horarios, o más conocido por su nombre en inglés *scheduling*, consiste en, dado un conjunto de tareas, asignar a cada tarea un horario específico para su realización dentro de un rango definido. Más formalmente, es el proceso de mapear las tareas a un conjunto de intervalos de tiempo, donde este mapeo debe satisfacer ciertas restricciones, que varían dependiendo del problema que se desea modelar. El factor común de esta familia de problemas es que algunas asignaciones son mejor que otras, y lo que se desea es buscar aquella óptima, de acuerdo a algún criterio especificado.

El problema de *scheduling* es un problema bastante estudiado en la literatura ya que en muy diversos ámbitos de la vida se hace necesario organizar tareas. A manera de ejemplo, algunos problemas de la vida cotidiana que requieren naturalmente ser optimizados y pueden modelarse como problemas de *scheduling* son los siguientes:

- La organización de horarios y orden en que las enfermeras visitan a los pacientes en sus casas es una tarea que debe ser optimizada. De esta manera se puede atender mejor y en menos tiempo a los pacientes que lo necesitan [1].

- El transporte público es un servicio que siempre debe funcionar de una manera óptima, es por eso que muchos estudios se han realizado en relación a este tópico y en particular para organizar los viajes diarios que se realizan [17].

- Organizar el calendario de un campeonato deportivo no es tarea fácil, más cuando se tienen que cumplir variadas restricciones, y de distintos tipos. Es por esto que la Federación Ecuatoriana de Fútbol (FEF) también ha modelado este problema como uno de *scheduling* [15].

- En las universidades es necesaria la organización de diversos elementos. Por ejemplo, las salas que se asignan a cada curso, el profesor que dicta cada curso y el horario de cada curso [9].

Técnicas de resolución

Para resolver un problemas de *scheduling* –encontrar la asignación óptima de tareas– se han desarrollado diversas técnicas. Estas pueden clasificarse en dos grandes grupos: *exactas* y *aproximadas*.

Las técnicas exactas son las que encuentran la solución óptima al problema, en caso de que alguna exista. La desventaja de esta familia de técnicas es que suelen ser caras,

computacionalmente hablando. Ejemplos de estas técnicas son el *branch-and-bound* y *cutting-planes* estudiadas en la sección anterior. Otro ejemplo es el algoritmo de *branch-and-price*, que consiste básicamente en la aplicación del método de *branch-and-bound*, pero en vez de realizar el paso de ramificación, se utiliza la solución al llamado *problema de los precios*, el cuál se encargará de encontrar la restricción que será añadida al problema al problema para acotar las soluciones de este y poder seguir iterando para encontrar la solución entera óptima (en caso de existir). Esto implica encontrar una columna que tenga un costo reducido negativo, en caso de existir, es agregada alprolema relajado y se itera nuevamente. En caso de no existir y la solución al problema relajado no es entera, se produce el paso de ramificación del método *branch-and-bound*. El algoritmo de *branch-and-price* algoritmo es utilizado, por ejemplo en [3], para resolver el *scheduling* de las fechas de torneos.

Otro tipo de técnicas usadas para resolver problemas de *scheduling* son las técnicas aproximadas o heurísticas. Estas técnicas son en general más eficientes que las exactas, pero no garantizan que la solución devuelta sea óptima, sino solo subóptima.

La *búsqueda tabú* es un algoritmo aproximado, el cual usa la búsqueda de vecinos para, iterativamente, ir moviéndose de una solución potencial x a una versión mejorada x^* en su vecindario. Esta heurística se utiliza, por ejemplo, para abordar el problema de *scheduling* de horario de apertura de tiendas [12].

Otra técnica heurística es el algoritmo *hill climbing*. El algoritmo parte de una solución arbitraria del problema x para luego intentar encontrar una mejor solución variando incrementalmente una única componente de x. En caso de encontrar una mejor solución cambia su estado al encontrado, sin pensar en las posibles acciones futuras. Esta técnica es utilizada, por ejemplo, para poder organizar los horarios de trabajo de todos los mecánicos en un taller automovilístico junto con las tareas y lugares de trabajo asociados a cada trabajador [5].

Scheduling en el contexto de universidades

Como se mencionó anteriormente, un escenario de aplicación recurrente de los problemas de *scheduling* son las universidades, en donde se utilizan para determinar la de distribución de salas para cursos, así como también el horario en el que será dictado cada curso. Dado que cada universidad tiene sus propias reglas y requisitos, existen varios trabajos al respecto.

El artículo de Gunawan et al [8] aborda el problema de la asignación de profesores a los cursos de un semestre. El problema es modelado como uno de optimización entera y para resolverlo se utilizan algoritmos híbridos, que constan de una heurística avara y un algoritmo de *annealing* modificado.

Por otro lado, Dimopoulo y Miliotis [6] abordan el problema de calendarización de cursos y evaluaciones, modelándolo como un problema de programación binaria. Si bien este trabajo aborda el problema de la calendarización de evaluaciones –al igual que nosotros–, este no incluye nuestros requisitos. En particular, no se busca obtener una correcta distribución de la carga académica a lo largo del semestre.

En [14] se trata el problema de la asignación de horarios a cursos. Se modela el problema como uno de programación entera y se resuelve utilizando la heurística de *búsqueda tabu*. Si

bien este trabajo aborda un problema similar al abordado en este trabajo de fin de carrera, los autores utilizan un algoritmo de resolución aproximado, mientras nuestro objetivo es generar soluciones óptimas.

Finalmente, en el artículo de Havås et al [9] aborda la asignación de profesores, cursos, salas y horarios en el contexto de la Universidad de Gotemburgo y la Universidad de Chalmers. El problema se modela a través de un programa entero binario, el que se resuelve utilizando el algoritmo *branch-and-cut*. Ahora bien, si bien este artículo ayuda a visualizar mejor nuestro problema, no lo resuelve. La formulación del modelo debe cambiar para adaptarse a nuestros requisitos. Se necesita modificar la función objetivo para que contemple la calendarización de evaluaciones, ya que lo que se busca optimizar en el artículo son las preferencias de horarios para los cursos y profesores en cuestión. Además, las restricciones del modelo también deben ser reformuladas para poder cumplir con los requisitos que se desea de una "buena calendarización". Por ejemplo, todas las restricciones que tratan la capacidad de las salas deben ser eliminadas, ya que no influyen para nada en una calendarización de evaluaciones. Mismo caso con las restricciones que relacionan los cursos con las salas. Por otro lado, hay restricciones que aseguran que no haya más de una cierta cantidad de evaluaciones por curso por día, la que puede ser utilizada para el mismo fin para asegurar una mejor distribución de la carga académica. También existe la restricción que asegura que todos los cursos tengan la cantidad correcta de evaluaciones y clases, la cuál puede adaptarse para garantizar que todos los ramos tengan todas sus evaluaciones calendarizadas.

Capítulo 3

Descripción del problema

En este capítulo se presentará el problema en detalle, se explicarán los términos necesarios para comprenderlo y por último, se explorarán las dificultades que este conlleva.

3.1. Elementos básicos de un semestre académico

Para describir el problema abordado en este trabajo de fin de carrera de manera precisa, comenzaremos presentando los diferentes elementos involucrados en la organización de un semestre académico de nuestra Facultad.

Semestre. El *semestre académico* en la Facultad de Ciencias Físicas y Matemáticas de la Universidad de Chile consta de 18 semanas. Las 15 primeras son de clases y evaluaciones, mientras que las últimas 3 están reservadas para la toma de exámenes finales. En la práctica, estas 18 semanas no se corresponden nunca con 18 semanas "calendario" consecutivas, ya que suelen haber semanas programadas de descanso e interrupciones imprevistas, generalmente debido a paros estudiantiles o por motivos de fuerza mayor (estallido social, contingencia sanitaria, etc.).

En términos de esta memoria, solo nos enfocaremos en la calendarización de las evaluaciones correspondientes a las primeras 15 semanas de clases, porque las últimas 3 semanas de clases corresponden a los exámenes finales, que son fijados por la Escuela y no por los profesores de cada curso.

Cursos. Cada semestre de la malla de una carrera está compuesto por un conjunto de *cursos* (o *ramos*) obligatorios y un conjunto de cursos electivos. Cada curso se imparte en varias clases semanales, en días y horarios preestablecidos.

Evaluaciones. Cada curso incorpora diversos tipos de *evaluaciones*, que pueden variar, entre otras cosas, en el tiempo que se da para realizarlas, la cantidad de materia que abordan o el método para rendirlas. Sin embargo, la mayoría de las evaluaciones caen dentro de alguna de las siguientes 4 categorías: *tareas*, *controles*, *ejercicios* y *presentaciones*.

Los controles, ejercicios y presentaciones son evaluaciones presenciales, que se fijan en un día de clases del curso. Por el contrario, las tareas se realizan de manera asíncrona, y generalmente se permite entregarlas cualquier día de la semana.

Un atributo importante de las evaluaciones es su *tiempo de preparación*. Con esto nos referimos al tiempo de estudio que requiere un control o ejercicio, el tiempo de preparación que requiera una presentación o el tiempo que lleva resolver una tarea.

Calendarización. Con *calendarización* de un semestre académico nos referiremos a la asignación de un día específico para la realización de cada evaluación de los cursos que conforman el semestre. Para los controles, ejercicios y presentaciones eso corresponde al día en que se van a llevar a cabo, y para las tareas eso corresponde al último día de plazo para su entrega.

Para los fines de esta memoria, se buscará calendarizar (solamente) las evaluaciones de los cursos obligatorios de cada semestre de la malla de una carrera dada. Esto debido a que los cursos electivos por cada semestre no son fijos, sino que se pueden elegir de un *pool* grande de cursos, y genera una noción de semestre académico muy variada para cada estudiante. Por ejemplo, al calendarizar el Semestre V de la malla de Ingeniería Civil en Computación consideraremos solo sus cursos obligatorios CC3101 (Matemáticas discretas para la computación), CC3001 (Algoritmos y estructuras de datos) y CC3501 (Modelación y computación gráfica para ingenieros). Más allá de esta decisión –de presentación–, cabe destacar que la solución desarrollada permite calendarizar cualquier conjunto de cursos, independientemente de su naturaleza.

3.2. El problema de las calendarizaciones

En la práctica, la manera en la que se define la calendarización de un semestre es completamente ad hoc. Cada profesor calendariza las evaluaciones de su curso siguiendo las pautas establecidas por su programa y también sus propios criterios, pero lo hace de una manera independiente, considerando sólo las necesidades de su ramo, o dicho de otra manera, sin tener en cuenta cómo sus evaluaciones interactúan con las evaluaciones de los demás cursos del semestre.

Esto generalmente resulta en una carga académica muy desbalanceada a lo largo del semestre, caracterizada por la presencia de periodos con una carga simplemente inviable. Por ejemplo, la presencia de semanas con 4 evaluaciones es una situación bastante habitual. De manera similar, no es extraña la presencia de días con 2 evaluaciones e incluso, hay veces en las que llegan a ser 3.

Muchas veces este problema suele abordarse de manera departamental, donde típicamente el jefe docente se encarga de "harmonizar" las evaluaciones para evitar estos periodos de sobrecarga. Sin embargo, este enfoque manual es poco satisfactorio por diversas razones. Primero, es muy tedioso. Segundo, no es fácil de resolver manualmente ya que requiere una visión global y simultánea de muchos elementos (cursos y evaluaciones). Tercero, no hay ninguna garantía que la solución derivada manualmente (generalmente consistente en la

postergación de algunas evaluaciones) sea óptima o cuasi-óptima. Y cuarto, es un problema recurrente que debe resolverse para cada semestre calendario (por ejemplo, 2020/2) y cada semestre académico (por ejemplo, Semestre V de la malla).

Se hace evidente la necesidad de abordar el problema de la calendarización de una manera más *sistemática*, que considere una *visión global* de todos los cursos (al menos obligatorios) del semestre, que provea alguna garantía formal sobre la *calidad* de la calendarización generada (en términos de la carga académica percibida por los estudiantes), y que sea automatizable.

Esto beneficiará tanto a estudiantes como a profesores. A los estudiantes les asegurará una carga académica razonable y viable, mejorando su desempeño académico y reduciendo su nivel de estrés, mejorando al final de cuenta su calidad de vida. A los profesores, les evitará la (tan frecuente) recalendarización de evaluaciones y les evitará posibles conflictos con los estudiantes.

3.3. Desafíos subyacentes

Desarrollar una herramienta que satisfaga los requisitos arriba mencionados no es nada fácil ya que requiere abordar un número de desafíos y requerimientos que muchas veces entran en conflicto.

Variabilidad de los requisitos de los profesores. La calendarización generada debe seguir –ser compatible con– los criterios que cada profesor utiliza para fijar sus evaluaciones. El inconveniente aquí es que estos criterios son muy variados. Por citar algunos ejemplos: "el Control 1 deber ser luego de 10 clases", "la fecha de entrega de la Tarea 2 debe ser previa al Control 2", "la Tarea 3 y el Control 3 no pueden caer en la misma semana", "los controles deben fijarse en días de clase auxiliar". Esta amplia gama de requisitos genera un dilema. Por un lado se podría adoptar un "lenguaje de especifición" altamente expresivo, que permita capturar todos estos requisitos. El problema de este enfoque es que posiblemente impacte negativamente en la usabilidad de la herramienta (especificar un requisito no sería tan simple) y también en su eficiencia (en cuanto más expresivo sea el lenguaje de especificación, más costoso será el problema de satisfactibilidad y optimización asociado). Por otro lado se puede adoptar un lenguaje de especificación de requisitos simple y menos expresivo. El inconveniente aquí es no dejar afuera demasiados requisitos, o requisitos muy usados.

Dificultad de cuantificar la calidad de una calendarización. Otro desafío sumamente relevante es cómo cuantificar la calidad de una calendarización. Una primera aproximación podría ser contar la cantidad de evaluaciones en un cierto periodo de tiempo, por ejemplo, por día o por semana. Sin embargo, considerar solo la cantidad no es suficiente ya que no todas las evaluaciones tienen el mismo nivel de dificultad: tener 3 evaluaciones "accesibles" en una semana es claramente menos demandante que tener 3 evaluaciones de dificultad elevada. Asimismo, considerar solo la dificultad de las evaluaciones no basta ya que tener muchas evaluaciones en un corto rango de tiempo, por muy poca dificultad que tengan, podría llegar a ser bastante estresante para los estudiantes. Esto debido a los sucesivos "cambios de contexto" que esto requeriría.

Factibilidad de una solución algorítmica eficiente. Ya que el problema a abordar consiste básicamente en un problema de optimización, debemos recordar que diferentes problemas de optimización tienen asociado diferente costo computacional, lo que en la práctica determina si pueden resolverse de manera exacta, o debe recurrirse a algoritmos aproximados. Idealmente deseamos un modelo de nuestro problema que permita resolverlo de manera exacta, en un tiempo razonable.

Imposibilidad de generar una calendarización única por semestre académico. Es imposible generar una calendarización para un semestre académico específico (por ejemplo, Semestre V de la malla) y repetirlo todos los semestres calendario (por ejemplo, 2021/1, 2022/1, etc.). Esto debido a que el profesor a cargo de un ramo puede variar de semestre a semestre, generando un esquema de evaluaciones distintos entre semestres. Además, aunque se lograra unificar las evaluaciones entre semestres, los días feriados o días en los que se sugiere no fijar evaluaciones (por ejemplo, días de la Jornada Chile de Computación) cambian indefectiblemente de año a año.

Capítulo 4

Solución

En este capítulo se expondrá la solución escogida y todo el proceso que se requiere para realizarla, desde su planteamiento teórico, hasta su posterior desarrollo práctico.

4.1. Diseño de la solución

Para abordar cada uno de los desafíos previamente mencionados se adoptaron las siguientes decisiones de diseño:

Variabilidad de los requisitos de los profesores. Se va a requerir que para cada evaluación, el profesor responsable provea un conjunto de días factibles en los que se puede calendarizar la evaluación. Si bien esta decisión deja afuera algunos criterios de calendarización (por ejemplo, que todas las tareas tengan como plazo el mismo día de la semana), es bastante robusta y permite capturar la mayoría de los criterios usados por los profesores para fijar sus evaluaciones. Por ejemplo, si se desea que la Tarea 2 se entregue antes que el Control 2, basta con que el último día factible declarado para la Tarea 2 sea previo al primer día factible declarado para el Control 2.

Además, para maximizar las posibilidades de encontrar una calendarización razonable (que genere una carga académica viable) se va a sugerir que el profesor provea más de una fecha factible para cada evaluación. En general, en cuanto más fechas factibles se declaren para cada evaluación, más altas serán las chances de encontrar mejores calendarizaciones.

Dificultad de cuantificar la calidad de una calendarización. La principal decisión de diseño adoptada para abordar este desafío fue considerar el tiempo de preparación de cada evaluación. Cabe destacar que la Facultad recientemente empezó a llevar cuenta de esta información en la plataforma de U-Cursos (pidiéndole tanto a profesores como a alumnos que la declaren), por lo que podemos asumir la presencia de estimaciones bastante precisas de esta variable. Partiendo del tiempo de preparación de cada evaluación podemos calcular la carga académica asociada a un periodo de tiempo dado (por ejemplo, un día o una semana) simplemente sumando el tiempo de preparación de las evaluaciones fijadas en ese periodo.

Vamos a suponer que una calendarización es *viable* si se cumplen las siguientes tres condiciones:

1. la carga académica de cada día está por debajo de un umbral;
2. la carga académica de cada semana está por debajo de otro umbral;
3. por curso, hay a lo sumo una evaluación diaria.

Factibilidad de una solución algorítmica eficiente. El punto clave acá es que modelamos el problema como uno de programación entera binaria, que admite soluciones bastante eficientes utilizando el algoritmo *branch-and-cut*. El modelo está construido en base a variables binarias de la forma $x_{\mathrm{d},w,r,t,\mathrm{i}}$ que valdrán 1 si la i-ésima evaluación de tipo t del ramo r es calendarizada el día d de la semana w y 0 si no.

Imposibilidad de generar una calendarización única por semestre académico. La calendarización generada va a incorporar información del semestre calendario en la cual se va a implantar. Además de la información respecto a las evaluaciones de cada ramo, se van a solicitar los días feriados y los días en los que se desaconseja fijar evaluaciones (también llamados "días no deseados").

4.2. Modelo matemático

Ahora describiremos en detalle el modelo matemático de nuestro problema, en términos de un programa entero binario. Primero comenzamos describiendo algunos conjuntos generales que son independientes del semestre que se desee calendarizar:

$D = \{1, 2, \ldots, 6, 7\}$	*Días de la semana*
$W = \{1, 2, \ldots, 14, 15\}$	*Semanas de un semestre*
$T = \{\text{tareas, ejercicios, controles, presentaciones}\}$	*Tipos de evaluaciones*

Luego introducimos los conjuntos que caracterizarán formalmente al semestre específico que se desea calendarizar.

$F \subseteq D \times W$	*Días del semestre donde no se permite fijar evaluaciones (típicamente, días feriados)*
$P \subseteq D \times W$	*Días del semestre donde se desaconseja fijar evaluaciones (por ejemplo, semana de la JCC)*
R	*Ramos cuyas evaluaciones serán calendarizadas*
$E \subseteq R \times T \times \mathbb{N}$	*Evaluaciones del semestre, representadas mediante tuplas, con el ramo al que pertenece la evaluación, el tipo y el índice. Por ejemplo (CC4101, control, 2) representa el Control 2 del ramo CC4101.*
$S : E \longrightarrow \mathcal{P}(D \times W)$	*Función que devuelve los días factibles en los que puede ser calendarizada cada evaluación*

$H : E \longrightarrow \mathbb{N}$	*Función que devuelve el tiempo (medido en horas) requerido para la preparación de cada evaluación*
$\bar{m}_{\mathrm{d}}$	*Carga académica máxima por día (medida en horas)*
$\bar{m}_w$	*Carga académica máxima por semana (medida en horas)*

Para mejorar la legibilidad, usaremos subíndices para denotar la aplicación a las funciones S y H. Por ejemplo, $S_{r,t,\mathrm{i}}$ denota el conjunto de días en los que puede ser calendarizada la i-ésima evaluación de tipo t del ramo r. Además, usaremos $E_r = \{(t, \mathrm{i}) \mid (r, t, \mathrm{i}) \in E\}$ para designar las evaluaciones pertenecientes al ramo r y $M_{r,t} = \text{máx}\,\{j \in \mathbb{N} \mid (t, j) \in E_r\}$ para designar el número de evaluaciones de tipo t del ramo r.

Ahora que ya tenemos todos los preliminares, presentaremos el programa entero binario que modelará nuestro problema de calendarización.

Variables. Las variables del programa, binarias, estarán indexadas por los conjuntos D, W, R, T y $\mathbb{N}$. Concretamente,

$$x_{\mathrm{d},w,r,t,\mathrm{i}} \in \{0, 1\} \qquad \forall\, \mathrm{d} \in D, w \in W, r \in R, t \in T, \mathrm{i} \in M_{r,t}\ ,$$

y $x_{\mathrm{d},w,r,t,\mathrm{i}}$ valdrá 1 si la i-ésima evaluación de tipo t del ramo r es calendarizada el día d de la semana w y 0 si no.

Restricciones. La primer restricción garantiza que cada evaluación se calendarice exactamente una vez:

$$\sum_{\mathrm{d} \in D} \sum_{w \in W} x_{\mathrm{d},w,r,t,\mathrm{i}} \;=\; 1 \qquad \forall\, (r, t, \mathrm{i}) \in E \tag{4.1}$$

La segunda restricción refina la primera, asegurando que cada evaluación se calendarice, además, en uno de los días factibles que fueron declarados:

$$\sum_{(\mathrm{d},w)\, \in\, S_{r,t,\mathrm{i}}} x_{\mathrm{d},w,r,t,\mathrm{i}} \;=\; 1 \qquad \forall\, r \in R,\ (t, \mathrm{i}) \in E_r \tag{4.2}$$

Observe que remover la primera restricción y dejar sólo la segunda no generaría soluciones correctas ya que permite que una evaluación sea calendarizada 2 veces, una en día factible y otra en un día no factible.

Las siguientes tres restricciones descartan aquellas calendarizaciones que no son viables exigiendo que la carga académica de cada día esté por debajo de $\bar{m}_{\mathrm{d}}$, la carga académica de cada semana esté por debajo de $\bar{m}_w$, y por ramo haya a lo más una evaluación por día:

$$\sum_{(r,t,\mathrm{i})\in E} x_{\mathrm{d},w,r,t,\mathrm{i}} \cdot H_{r,t,\mathrm{i}} \;\leq\; \bar{m}_{\mathrm{d}} \qquad \forall\, \mathrm{d} \in D, w \in W \tag{4.3}$$

$$\sum_{\mathrm{d}\in D} \sum_{(r,t,\mathrm{i})\in E} x_{\mathrm{d},w,r,t,\mathrm{i}} \cdot H_{r,t,\mathrm{i}} \;\leq\; \bar{m}_{w} \qquad \forall\, w \in W \tag{4.4}$$

$$\sum_{(t,\mathrm{i})\in E_r} x_{\mathrm{d},w,r,t,\mathrm{i}} \;\leq\; 1 \qquad \forall\, r \in R, \mathrm{d} \in D, w \in W \tag{4.5}$$

Aclaración: en las restricciones (4.3) y (4.4), $\bar{m}_{\mathrm{d}}$ y $\bar{m}_w$ no están ligadas a la cuantificación universal $\forall\, \mathrm{d} \in D$ y $\forall\, w \in W$. Se usarán las mismas cotas para todos los días y para todas las semanas.

Cabe destacar que hay dos requisitos adicionales que consideramos sobre las calendarizaciones, pero no los incluimos en el modelo porque son garantizados por la herramienta durante el proceso de ingreso de datos. El primero es que no se permiten calendarizar evaluaciones en días feriados. Al iniciar la herramienta, lo primero que se pide al usuario es que ingrese los días feriados; luego no le permite indicar un día feriado como factible de ninguna evaluación. El segundo es que para cada ramo, las evaluaciones del mismo tipo sean calendarizadas respetando su orden relativo. Dicho de otra manera, que el Control i-ésimo se calendarice antes que el Control $(\mathrm{i}+1)$-ésimo, y lo mismo para los otros tipos de evaluaciones. Esto se asegura verificando que el último día factible declarado para la evaluación i-ésima de cada tipo sea anterior al primer día factible declarado para la evaluación $(\mathrm{i}+1)$-ésima.

Función objetivo. Las restricciones (4.1)-(4.5) definen el espacio de calendarizaciones válidas y viables del semestre y garantizan una distribución de cargas "sana" para los estudiantes. Ahora nos resta elegir una de ellas de acuerdo a algún criterio razonable. Para ello vamos adoptar la siguiente suposición:

> *Dadas dos calendarizaciones válidas y viables para los estudiantes, (para los profesores) es preferible aquella que fija las evaluaciones lo más pronto posible, dentro de los días factibles declarados para cada una de ellas.*

Esto da origen a la siguiente función objetivo que se buscará minimizar:

$$f \;=\; \sum_{\mathrm{d}\in D} \sum_{w\in W} \sum_{(r,t,\mathrm{i})\in E} x_{\mathrm{d},w,r,t,\mathrm{i}} \cdot p_t \cdot \mathcal{K}^{S_{r,t,\mathrm{i}}}(\mathrm{d}, w)$$

Intuitivamente, f cuantifica cuán lejos se fija cada evaluación con respecto a su primer día factible. En el caso ideal donde todas las evaluaciones son calendarizadas el primer día factible, el valor de f sería 0.

En la definición de f, $\mathcal{K}^T(\mathrm{d}, w)$ devuelve la distancia del día (d, w) al primer día (cronológicamente hablando) en $T \subseteq D \times W$. Por ejemplo, si $T = \{(5,7),(5,8),(5,9)\}$, entonces $\mathcal{K}^T(5,9) = 2$, ya que si ordenamos cronológicamente los días en T, $(5,9)$ ocupa la segunda posición (empezando a contar desde cero). Observe que el valor de $\mathcal{K}^T(\mathrm{d}, w)$ es irrelevante cuando $(\mathrm{d}, w) \notin T$, ya que por la forma en la que se emplea $\mathcal{K}^T(\mathrm{d}, w)$ en f –con $T = S_{r,t,\mathrm{i}}$–, si $(\mathrm{d}, w) \notin T$ entonces el factor $x_{\mathrm{d},w,r,t,\mathrm{i}}$ que acompaña a $\mathcal{K}^T(\mathrm{d}, w)$ será 0. Finalmente en la

definición de f, el coeficiente $p_t \in \mathbb{R}_{\geq 0}$ permite ponderar de distinta manera la postergación de cada tipo de evaluación. Esto permite expresar, por ejemplo, que postergar un control es menos deseable que postergar una tarea (escogiendo $p_{\text{control}} > p_{\text{tarea}}$) o que postergar una tarea no conlleva ninguna penalización (escogiendo $p_{\text{tarea}} = 0$).

4.3. Optimización del modelo

El modelo recién descrito admite una optimización, que si bien simple, puede repercutir positivamente en el tiempo requerido de resolución, en nuestro caso usando el algoritmo *branch-and-cut*. Para aplicar esta optimización, partimos observando que en vista de la restricción (4.1), la restricción (4.2) resulta equivalente a:

$$\sum_{(\mathrm{d},w)\, \in\, D\times W\setminus S_{r,t,\mathrm{i}}} x_{\mathrm{d},w,r,t,\mathrm{i}} \;=\; 0 \qquad\qquad \forall\, r \in R, (t,\mathrm{i}) \in E_r \tag{4.2'}$$

Dicho de otra manera, ninguna evaluación se puede fijar en un día no indicado como factible para ello. Ahora, explotando que las variables involucradas $x_{\mathrm{d},w,r,t,\mathrm{i}}$ son binarias, resulta fácil ver la restricción de arriba es equivalente a:

$$x_{\mathrm{d},w,r,t,\mathrm{i}} \;=\; 0 \qquad\qquad \forall\, r \in R, (t,\mathrm{i}) \in E_r, (\mathrm{d},w) \in D \times W \setminus S_{r,t,\mathrm{i}} \tag{4.2*}$$

Observe que esta restricción es preferible a la restricción original (4.2) ya que determina unívocamente el valor de un conjunto de las variables involucradas. Reemplazando la restricción original (4.2) por esta última (4.2*) obtenemos el programa lineal final de nuestro problema:

$$\text{mín} \sum_{\mathrm{d}\in D}\sum_{w\in W}\sum_{(r,t,\mathrm{i})\in E} x_{\mathrm{d},w,r,t,\mathrm{i}} \cdot p_t \cdot \mathcal{K}^{S_{r,t,\mathrm{i}}}(\mathrm{d},w)$$

$$\text{s.a.} \begin{cases} \displaystyle\sum_{\mathrm{d}\in D}\sum_{w\in W} x_{\mathrm{d},w,r,t,\mathrm{i}} \;=\; 1 & \forall\, (r,t,\mathrm{i}) \in E \\ x_{\mathrm{d},w,r,t,\mathrm{i}} \;=\; 0 & \forall\, r \in R, (t,\mathrm{i}) \in E_r, (\mathrm{d},w) \in D \times W \setminus S_{r,t,\mathrm{i}} \\ \displaystyle\sum_{(r,t,\mathrm{i})\in E} x_{\mathrm{d},w,r,t,\mathrm{i}} \cdot H_{r,t,\mathrm{i}} \;\leq\; \bar{m}_{\mathrm{d}} & \forall\, \mathrm{d} \in D, w \in W \\ \displaystyle\sum_{\mathrm{d}\in D}\sum_{(r,t,\mathrm{i})\in E} x_{\mathrm{d},w,r,t,\mathrm{i}} \cdot H_{r,t,\mathrm{i}} \;\leq\; \bar{m}_w & \forall\, w \in W \\ \displaystyle\sum_{(t,\mathrm{i})\in E_r} x_{\mathrm{d},w,r,t,\mathrm{i}} \;\leq\; 1 & \forall\, r \in R, \mathrm{d} \in D, w \in W \end{cases}$$

$$x_{\mathrm{d},w,r,t,\mathrm{i}} \in \{0,1\}$$

4.4. Ranking de calendarizaciones

La herramienta que desarrollamos (ver Sección 4.5 para más detalles) permite reportar la cantidad de calendarizaciones (sub)óptimas que el usuario desee. De esta manera, se podrán comparar las mejores, digamos N, calendarizaciones para seleccionar la más conveniente.

Para realizar dicha selección de manera informada, la herramienta reportará para cada calendarización –además del valor de la función objetivo f– el valor de otras tres variables relevantes a la hora de ponderar la calidad de la calendarización. Estas variables son:

1. varianza de la carga académica diaria (σ^2_{d});
2. varianza de la carga académica semanal (σ^2_w);
3. cantidad de evaluaciones fijadas en días "no deseados" (ζ).

Las dos primeras variables permiten cuantificar qué tan homogéneamente está distribuida la carga académica a lo largo del semestre. Por ejemplo, una varianza de carga semanal baja indica que todas las semanas del semestre tienen asociada una carga similar, mientras que una varianza alta implica mucha variabilidad. Para definirlas formalmente, debemos introducir primero las siguientes expresiones que denotan, respectivamente, la carga asociada a un día (d, w), la carga asociada a una semana w y la carga asociada a todo el semestre, medidas como la suma del tiempo de preparación (en horas) de las evaluaciones en cada uno de estos periodos:

$$\begin{aligned}
\mathcal{L}(\mathrm{d}, w) &= \sum_{(r,t,\mathrm{i})\in E} x_{\mathrm{d},w,r,t,\mathrm{i}} \cdot H_{r,t,\mathrm{i}} \\
\mathcal{L}(w) &= \sum_{\mathrm{d}\in D} \mathcal{L}(\mathrm{d}, w) \\
\mathcal{M} &= \sum_{w\in W} \mathcal{L}(w)
\end{aligned}$$

La definición de la varianza de carga diaria y semanal es ahora inmediata:

$$\begin{aligned}
\sigma^2_{\mathrm{d}} &= \sum_{w\in W} \sum_{\mathrm{d}\in D} \left(\mathcal{L}(\mathrm{d}, w) - \frac{\mathcal{L}(w)}{|D_w|} \right)^2 \\
\sigma^2_w &= \sum_{w\in W} \left(\mathcal{L}(w) - \frac{\mathcal{M}}{|W|} \right)^2
\end{aligned}$$

En la definición de la varianza de carga diaria σ^2_{d}, $D_w = \{\mathrm{d} \in D \mid (\mathrm{d}, w) \notin F\}$ representa el conjunto de días no feriados de la semana w. Observe que la varianza de carga diaria σ^2_{d} se calculó no con respecto a la carga diaria promedio para todo el semestre, sino que para cada día, se consideró su distancia a la carga promedio diaria de la semana en la que se encuentra dicho día. Dicho de otra manera, esto corresponde a calcular la varianza de carga diaria para cada semana, y sumarlas.

Finalmente, la tercer variable es bastante natural; en general se van a preferir calendarizaciones que fijen la menor cantidad de evaluaciones posibles en días "no deseados". Formal-

mente, se calcula de la siguiente manera:

$$\zeta = \sum_{(\mathrm{d},w)\in P} \sum_{(r,t,\mathrm{i})\in E} x_{w,\mathrm{d},r,t,\mathrm{i}}$$

4.5. Implementación

Para automatizar y evaluar la solución propuesta, se desarrolló una herramienta que toma como entrada los datos del semestre a calendarizar, genera el programa lineal asociado, lo resuelve, y devuelve la(s) calendarización(es) óptima(s). Más precisamente,

Entrada:

- conjunto de ramos que se desea calendarizar (por ejemplo, CC3101, CC3001 y CC3501);
- para cada ramo, el conjunto de sus evaluaciones. Para cada evaluación se solicita su tipo e índice (por ejemplo, Control 1), posibles fechas en las que se puede calendarizar (por ejemplo, viernes de la Semana 7 o viernes de la Semana 8) y su tiempo de preparación (por ejemplo, 4 horas) (ver Figura 4.1);
- carga máxima permitida por día y por semana (por ejemplo, 10 y 25 horas);
- factor de ponderación de cada tipo de evaluación para la definición de la función objetivo (por ejemplo, $p_{\text{control}} = p_{\text{presentación}} = p_{\text{ejercicio}} = 1/3$, $p_{\text{tarea}} = 0$);
- días feriados y días en los que se sugiere no fijar evaluaciones (por ejemplo, lunes de la Semana 5 y lunes a viernes de la Semana 10) (ver Figura 4.2);
- número de calendarizaciones (sub)óptimas que se desea generar.

Salida: Para cada una de las calendarizaciones generadas se muestra:

- evaluaciones fijadas en cada día (ver Figura 4.3);
- valor de la función objetivo, de la varianza de carga diaria y semanal, y número de evaluaciones fijadas en días "no deseados" (ver Figura 4.4).

Para resolver el programa lineal se utiliza el algoritmo *branch-and-cut* ya que combina los beneficios de los algoritmos *branch-and-bound* y *cutting-planes*, y es el preferido en la práctica para la resolución de programas enteros. Además, está soportado de una manera muy conveniente por la librería MIP de Python, lenguaje que usaremos para implementar el *backend*.

La herramienta está accesible a través de la página web:

`https://memoria-68493.firebaseapp.com/`

Backend. Como anticipado arriba, para programar el *backend* se decidió utilizar el lenguaje de programación Python pues es de buen dominio del memorista, tiene una comunidad muy grande, y presenta un extenso soporte de librerías para las más diversas aplicaciones. En particular, la librería MIP[1] (Mixed-Integer Linear Programming) provee un conjunto de

[1] `https://www.python-mip.com/`

Figura 4.1: Conjunto de evaluaciones asociadas a un ramo.

Figura 4.2: Selección de días feriados correspondientes al semestre que se quiere calendarizar.

herramientas para la resolución de programas lineales mixtos (con variables tanto reales como enteras). Dentro de estas herramientas se encuentra el algoritmo de *branch-and-cut* que utilizamos para resolver nuestro programa entero binario. Dos características destacadas de la librería son su alto nivel de abstracción para describir los programas lineales y una implementación optimizada para minimizar el tiempo de resolución. De manera ilustrativa, el siguiente fragmento de código especifica la función objetivo del programa:

```
m.objective = minimize(xsum(x[w][d][r][t][i] * dist(S[r][t][i],d,w)
* weight[t] for (r,t,i) in E, for d in D, for w in W))
```

Por su parte, el siguiente fragmento agrega la restricción (4.1) (ver página 21) al modelo:

Figura 4.3: Evaluaciones fijadas para un día en específico del semestre.

Puntaje #	Cercanía a primera opción de calendarización	Varianza de carga académica semanal	Varianza de carga académica diaria	Evaluaciones en días no deseados
Schedule 1	0.3	21.8	21.3	2
Schedule 2	1.0	27.7	20.3	2
Schedule 3	2.1	20.7	21.4	2
Schedule 4	2.8	26.8	20.5	2

Figura 4.4: Tabla con información estadística sobre las calendarizaciones devueltas.

```
for (r,t,i) in E:
    m += xsum(x[w][d][r][t][i] for d in D, for w in W) == 1
```

Una limitación de la librería con la que nos encontramos es que devuelve una única solución óptima del programa. Por lo tanto, para obtener las *top-n* soluciones óptimas de nuestro programa tuvimos que resolver secuencialmente n programas distintos (donde el programa (i+1)-ésimo se obtiene a partir del i-ésimo agregando una restricción que "elimine" la solución óptima de este último).

Frontend. La interfaz se encarga de recopilar y validar los datos respecto al semestre que se quiere calendarizar para posteriormente mostrar las calendarizaciones calculadas. La validación de datos se encarga de garantizar dos requisitos adicionales a los capturados por el modelo matemático: asegura que no se calendarice ninguna evaluación un día feriado (bloqueando la selección de días feriados) y que se respete el orden relativo de las evaluaciones del mismo tipo (verificando que los conjuntos de días factibles para cada una de ellas no se solapen, y respeten su orden relativo). La interfaz se programó en Javascript, usando la

biblioteca de Reactjs. En las Figuras 4.1, 4.2, 4.3 y 4.4 se puede observar distintas partes de la misma.

Arquitectura. Ya que se trata de una herramienta sencilla, sin mucha interacción entre el *forntend* y el *backend*, se decidió adoptar una arquitectura de software MVC (Modelo Vista Controlador). De esta manera, en la interfaz, o vista, se obtendrán los datos de *input*. Luego el controlador los modificará para poder llamar al modelo de optimización. Una vez obtenidos los resultados de las calendarizaciones especificadas en el input, el controlador nuevamente adaptará estos datos para que sea representables en la interfaz.

Capítulo 5

Validación

Para validar la solución propuesta, se realizó una evaluación cuantitativa comparando las calendarizaciones devueltas por la herramienta con las obtenidas de manera manual, como ocurre hoy en día. La hipótesis que se busca validar es que el uso de la herramienta da lugar a calendarizaciones más viables para los estudiantes, con una distribución de cargas más efectiva y racional.

5.1. Diseño del experimento

Para ello, se consideraron tres semestres de la malla de Ingeniería Civil en Computación de nuestra Facultad (Semestres V, VI y VII), se generaron calendarizaciones óptimas (de sus ramos obligatorios) utilizando la herramienta y se compararon las mismas con respecto a calendarizaciones reales que tuvieron dichos semestres en 2016/1, 2016/2 y 2017/1, respectivamente.

Criterios de comparación. La comparación se llevó a cabo en base a un total de *nueve variables cuantitativas*, destinadas a "medir" de la manera más objetiva posible la calidad de las calendarizaciones. Dichas variables son:

- $\mathbf{V_1}$: cantidad de semanas con 3 o más evaluaciones.
- $\mathbf{V_2}$: cantidad de días con más de 1 evaluación.
- $\mathbf{V_3}$: cantidad de semanas con una carga académica mayor a 30 ($\mathbf{V_{31}}$), 35 ($\mathbf{V_{32}}$) y 40 ($\mathbf{V_{33}}$) horas.
- $\mathbf{V_4}$: cantidad de días con una carga académica mayor a 15 ($\mathbf{V_{41}}$), 17 ($\mathbf{V_{42}}$) y 20 ($\mathbf{V_{43}}$) horas.
- $\mathbf{V_5}$: carga académica semanal máxima, medida en horas.
- $\mathbf{V_6}$: carga académica diaria máxima, medida en horas.
- $\mathbf{V_7}$: cantidad de segmentos de 2 o más días consecutivos, todos con evaluaciones.
- $\mathbf{V_8}$: varianza de la carga académica semanal.
- $\mathbf{V_9}$: varianza de la carga académica diaria.

Las variables 1 y 2 ofrecen una buena comparación inicial de la viabilidad de dos calendarizaciones. Para la variable 1 se adoptó un umbral de 3 evaluaciones por semana ya que representa el doble de evaluaciones que habría por semana si estas estuvieran distribuidas uniformemente durante todo el semestre: los 3 semestres considerados para el experimento tienen un promedio de 20 evaluaciones cada uno (recordar que se consideraron solo los ramos obligatorios de cada semestre, que son tres) y si estas estuvieran distribuidas uniformemente habría $20/15 = 1.33$ evaluaciones por semana. La variable 2 considera un umbral de 1 evaluación diaria pues asume que, para rendir de manera óptima, un estudiante debería enfocarse en no más de 1 evaluación por día.

El inconveniente de las variables 1 y 2 es que sólo consideran cantidad de evaluaciones y no ponderan el tiempo de preparación de cada una, que presenta de hecho una alta variabilidad. Por ejemplo, mientras algunas tareas tienen asociado un tiempo de preparación de 4 horas, algunos controles requieren una preparación de 15 horas. Es por eso que se incorporan las variables 3 y 4, que miden la carga semanal y diaria de una manera más precisa, considerando el tiempo de preparación de sus evaluaciones. Para la variable 3, se adoptó umbrales semanales de 30, 35 y 40 horas porque, como argumentaremos más adelante, creemos que 35 horas es la carga máxima que debería haber por semana para que una calendarización sea viable. Para los umbrales diarios de la variable 4, se consideró un escenario pesimista donde la mitad de la carga semanal máxima se da en un mismo día (observe que 15, 17 y 20 corresponde respectivamente a la mitad de 30, 35 y 40).

Las variables 5 y 6 miden la máxima concentración de carga en un periodo de una semana y de un día y creemos que no necesitan mayores explicaciones.

La presencia de varios días consecutivos con evaluaciones, sin descanso entre medio, da origen a estrés y puede repercutir negativamente en la preparación de las evaluaciones. La variable 7 se encarga precisamente de dar cuenta de cuántos veces ocurre este tipo de situaciones durante el semestre.

Finalmente, el objetivo de las variables 8 y 9 es cuantificar qué tan uniformemente está distribuida la carga a lo largo del semestre. Por ejemplo, una alta varianza de carga semanal implica una alta variabilidad de la carga, con semanas de mucha carga y semanas de poca carga. Por el contrario, una varianza de carga semanal baja implica que todas (o gran parte de) las semanas conllevan una carga similar.

5.2. Datos de prueba

El experimento se hizo a partir de los Semestres V, VI y VII de la malla de la Ingeniería Civil en Computación, considerando sus cursos obligatorios:

Semestre V

- Matemáticas discretas para la computación - CC3101
- Algoritmos y estructuras de datos - CC3001
- Modelación y computación gráfica para ingenieros - CC3501

Semestre VI

- Teoría de la computación - CC3102
- Base de datos - CC3201
- Metodologías de diseño y programación - CC3002

Semestre VII

- Lenguajes de programación - CC4101
- Ingeniería de software - CC4401
- Arquitectura de computadores - CC4301

Calendarizaciones óptimas. Para cada uno estos 3 semestres, se generaron las *top*-5 calendarizaciones usando la herramienta desarrollada. Para ello, se entregó el siguiente *input* a la herramienta:

Conjunto de días factibles y tiempo de preparación de cada evaluación ($S\colon E \to \mathcal{P}(D \times W), H\colon E \to \mathbb{N}$). Para determinar esta información se contactó por correo electrónico a los profesores a cargo de los respectivos cursos y se obtuvo la información correspondiente, resumida en el Apéndice A.1.

Carga académica máxima diaria y semanal ($\bar{m}_{\mathrm{d}}$ y $\bar{m}_w$). Como carga diaria máxima permitida se tomó $\bar{m}_{\mathrm{d}} = 20$ horas. La razón de esta elección es que, para los 3 semestres que considerados en nuestro experimento, el mayor tiempo de preparación de una evaluación es de 20 horas (ver Apéndice A.1). Elegir un $\bar{m}_{\mathrm{d}}$ menor haría que el problema sea insatisfactible. Por otro lado, se decidió no elegir un $\bar{m}_{\mathrm{d}}$ mayor ya que consideramos que esto generaría un carga excesiva sobre los estudiantes. Con respecto a la carga semanal máxima, se adoptó $\bar{m}_w = 37$ horas. Para derivar este valor se consideró que un estudiante debe dedicar en total 40 horas a la semana a la Facultad, de las cuales aproximadamente 13 horas corresponden a tiempo de cursado, dejándole un total de 37 horas para la preparación de las evaluaciones.

Pesos para ponderar la postergación de cada tipo de evaluación ($\{p_t\}_{t \in T}$). Se otorgó un peso $p = \frac{1}{3}$ para las tareas, controles y ejercicios, dado que son las evaluaciones más recurrentes en cada ramo. Por otro lado, el peso de las presentaciones se fijó en $p = 0$ ya que las presentaciones no están presentes en casi ninguno de los ramos de los semestres de prueba. De los 9 ramos utilizados en los 3 semestres, solamente Ingeniería de Software usa presentaciones dentro de su gama de evaluaciones.

Días feriados. Ya que como *baseline* de la comparación se va a utilizar la calendarización que tuvieron estos 3 semestres en 2016/1, 2016/2 y 2017/1 respectivamente, se va a generar calendarizaciones óptimas para estos mismos semestres. Por lo tanto, como feriados se utilizaron los días feriados que hubo respectivamente en 2016/1, 2016/2 y 2017/1.

Días no sugeridos para fijar evaluaciones. Se adoptaron i) los 3 días en que se desarrollan las Jornadas Chilenas de Computación, ya que siempre hay una sugerencia departamental de no fijar evaluaciones esos días, y ii) el viernes en que se realiza la Fonda DCC, ya que consideramos que es un evento social para de todos los estamentos del DCC.

Semestre	Calendarización	f	σ^2_{d}	σ^2_w	ζ
V	1	0.0	46.8	46.4	0
	2	0.3	46.8	46.4	0
	3	0.3	46.8	46.4	0
	4*	0.3	42.3	47.0	0
	5	0.3	46.8	46.4	0
VI	1*	0.7	33.1	39.9	1
	2	1.0	33.1	40.3	1
	3	1.0	33.1	39.9	1
	4	1.0	33.1	39.9	1
	5	1.0	33.8	39.7	1
VII	1*	1.3	52.7	39.3	0
	2	1.3	52.7	39.3	0
	3	1.3	52.7	39.3	0
	4	1.3	52.7	39.3	0
	5	1.3	52.7	39.3	0

Tabla 5.1: Información estadística presentada por la herramienta para cada una de las calendarizaciones generadas. La calendarización óptima seleccionada para cada semestre está marcada con un asterisco.

Calendarizaciones de *baseline*. Como *baseline* de la comparación, se emplearon las calendarizaciones reales –construidas manualmente– que estos tres semestres tuvieron en 2016/1, 2016/2 y 2017/1, respectivamente. Se eligieron estos tres semestres de *baseline* ya que fueron aquellos en los que el memorista cursó los Semestres V, VI y VII de la malla, y por lo tanto, se disponía de información completa y confiable al respecto. Estas calendarizaciones de *baseline* pueden consultarse en el Apéndice A.3.

5.3. Resultados

Para la generación de las 5 calendarizaciones óptimas por semestre, la herramienta requiere un tiempo de 13.2 segundos (Semestre V), 18.6 segundos (Semestre VI) y 11.5 segundos (Semestre VII). Las calendarizaciones generadas pueden consultarse en el Apéndice A.2. En la Tabla 5.1 reproducimos la información estadística devuelta por la herramienta para cada una de ellas. Recordemos que esta información consiste en:

- f : distancia ponderada (por el tipo de evaluación) entre la fecha donde se fijó cada evaluación y la primer fecha factible declarada para ello (función objetivo del modelo);
- σ^2_{d} : varianza de la carga académica diaria;
- σ^2_w : varianza de la carga académica semanal;
- ζ : cantidad de evaluaciones fijadas en días "no deseados".

Selección de calendarización óptima

Para elegir la calendarización óptima del Semestre V descartaremos del análisis la varianza de carga semanal (σ^2_w) y la cantidad de evaluaciones en días no deseados (ζ) ya que

Sem.	Calen.	$\mathbf{V_1}$	$\mathbf{V_2}$	$\mathbf{V_3}$			$\mathbf{V_4}$			$\mathbf{V_5}$	$\mathbf{V_6}$	$\mathbf{V_7}$	$\mathbf{V_8}$	$\mathbf{V_9}$
				$\mathbf{V_{31}}$	$\mathbf{V_{32}}$	$\mathbf{V_{33}}$	$\mathbf{V_{41}}$	$\mathbf{V_{42}}$	$\mathbf{V_{43}}$					
V	Óptima	1	1	1	1	0	3	3	0	36	20	1	42.3	47.0
	Real	4	2	2	2	2	4	4	1	42	42	2	57.3	56.4
VI	Óptima	2	3	0	0	0	3	3	0	25	20	2	33.1	39.9
	Real	5	4	2	1	1	3	3	1	41	32	4	46.3	46.1
VII	Óptima	4	2	3	2	0	0	0	0	37	15	5	52.7	39.3
	Real	2	3	2	2	1	3	3	2	51	24	3	56.7	46.4

Tabla 5.2: Tabla comparativa de calendarizaciones reales versus calendarizaciones óptimas, según la herramienta desarrollada.

las 5 calendarizaciones presentan valores (casi) idénticos para estas variables. Luego como posibles candidatas quedan las Calendarizaciones 1 y 4: la Calendarización 1 ofrece la menor postergación de evaluaciones (f) –0 contra 0.33–, y la Calendarización 4 ofrece la menor varianza de carga diaria (σ_{d}^2) –42.3 contra 46.8–. Elegiremos como óptima la Calendarización 4 ya que, dada la elección de nuestros parámetros ($p_{\mathrm{control}} = p_{\mathrm{tarea}} = p_{\mathrm{ejercicio}} = 1/3$, $p_{\mathrm{presentación}} = 0$), su peor desempeño en cuanto a postergación de evaluaciones se debe a la postergación de una evaluación a su segundo día factible, lo que consideramos para nada crítico. Además, la Calendarización 4 presenta una varianza de carga semanal un 10 % menor que la Calendarización 1.

Para el Semestre VI se escogerá la Calendarización 1 como la óptima ya que, presentando valores similares en varianza de carga (σ_{d}^2 y σ_w^2) y número de evaluaciones en días no deseados (ζ) con respecto a las demás calendarizaciones, presenta el mejor rendimiento en cuanto a postergación de evaluaciones (f) –0.66 contra 1–.

Finalmente, en el Semestre VII todas las calendarizaciones presentan valores idénticos en las 4 variables analizadas. Elegiremos –arbitrariamente– la Calendarización 1 como óptima.

Comparación entre calendarización óptima y real

Una vez seleccionada la calendarización óptima para cada semestre, se procede a comparar las mismas con las calendarizaciones reales que estos tuvieron respectivamente en 2016/1, 2016/2 y 2017/1, en base a las 9 variables cuantitativas descritas anteriormente (ver Sección 5.1). La Tabla 5.2 muestra el valor de dichas variables para la calendarización óptima y la real de cada uno de los tres semestres.

Para el Semestre V, la calendarización óptima mejora a la calendarización real en el total de las 9 variables. Situación muy parecida se da para el Semestre VI, con la salvación de que en las variables $\mathbf{V_{41}}$ y $\mathbf{V_{42}}$ los valores fueron iguales en ambas calendarizaciones.

En el caso del Semestre VII los resultados no son tan determinantes. Si bien la calendarización óptima mejora a la real en la mayoría de las variables, en las variables $\mathbf{V_1}$, $\mathbf{V_{31}}$ y $\mathbf{V_7}$ se observa lo contrario. Para explicar este comportamiento inesperado presentamos las calendarizaciones óptima y real en las Tablas 5.3 y 5.4, respectivamente.

	LU	MA	MI	JU	VI	SÁ	DO	Carga semanal
S-1								0 hrs
S-2			CC4301-E1[4h]					4 h
S-3		CC4301-E2[4h]						4 h
S-4		CC4301-E3[4h]						4 h
S-5			CC4101-T1[10h]					10 h
S-6	CC4301-T1[10h]			CC4401-P1[15h]	CC4101-C1[12h]			37 h
S-7	CC4301-C1[12h]	CC4301-E4[4h] CC4401-C1[3h]						19 hrs
S-8								0 h
S-9			CC4101-T2[14h]					14 h
S-10	CC4301-E5[4h]		CC4301-C2[12h]	CC4401-P2[15h]				31 h
S-11								0 h
S-12	CC4301-E6[4h]							4 h
S-13		CC4401-P3[15h]	CC4101-T3[14h]					29 h
S-14	CC4301-C3[12h]	CC4401-C2[3h] CC4301-T2[10h]			CC4101-C2[12h]			37 h
S-15								0 hrs

Tabla 5.3: Calendarización óptima para el Semestre VII.

La variable $\mathbf{V_1}$ representa el número de semanas con 3 o más evaluaciones. En la calendarización real, esto se da en las Semanas 6 y 14; en la calendarización óptima, además de en estas dos semanas, también en las 7 y 10. Analizando la Semana 10, se puede observar que en la calendarización óptima las tres evaluaciones fijadas para esa semana (Presentación 2 CC4401, Control 2 CC4301 y Ejercicio 5 CC4301) solo tienen como posibles fechas de calendarización días de la Semana 10 (ver Apéndice A.1), por lo que cualquier calendarización devuelta por la herramienta las agrupará en dicha semana. Por el contrario, eso no ocurre en la calendarización real ya que una de las tres evaluaciones (Presentación 2 CC4401) se fijó la semana siguiente.

En el caso de la Semana 7, la calendarización óptima contiene 2 evaluaciones (Ejercicio 4 CC4301 y Control 1 CC4401) que pueden ser fijadas únicamente esa semana y una tercer evaluación (Control 1 CC4301) que puede ser fijada tanto esa semana como la anterior. Sin embargo, esta última evaluación se fija esa semana porque la anterior (Semana 6) ya está ya al límite de su carga académica máxima. Esta colisión de 3 evaluaciones en la Semana 7 no ocurre en la calendarización real porque una de ellas (Control 1 CC4401) se postergó para la Semana 8. Dicho de otra manera, la calendarización real *no* respeta los días factibles de cada evaluación usados para generar la calendarización óptima, lo que explica porqué tiene 2 semanas menos con 3 o más evaluaciones.

La variable $\mathbf{V_{31}}$ representa el número de semanas de carga total mayor a 30 horas. La calendarización óptima tiene una semana más de este tipo que la calendarización real: Semanas 6, 10 y 14 (calendarización óptima) versus Semanas 6 y 14 (calendarización real). Observe que la semana extra es la Semana 10, y el origen de su carga excesiva en la calendarización óptima fue explicado dos párrafos arriba, al analizar la variable $\mathbf{V_1}$.

	LU	MA	MI	JU	VI	SÁ	DO	Carga semanal
S-1								0 h
S-2				CC4301-E1[4h]				4 h
S-3			CC4301-E2[4h]					4 h
S-4			CC4301-E3[4h]					4 h
S-5								0 h
S-6			CC4301-C1[12h]	CC4301-T1[10h] CC4101-T1[14h]				36 h
S-7			CC4301-E4[4h] CC4401-P1[15h]					19 h
S-8				CC4401-C1[3h]				3 h
S-9					CC4101-C1[12h]			12 h
S-10			CC4301-C2[12h]	CC4301-E5[4h]				16 h
S-11	CC4101-T2[14h]		CC4401-P2[15h]					29 h
S-12			CC4301-E6[4h]					4 h
S-13								0 h
S-14			CC4301-T2[10h]	CC4401-C2[3h]	CC4101-C2[12h] CC4301-C3[12h]		CC4101-T3[14h]	51 h
S-15		CC4401-P3[15h]						15 h

Tabla 5.4: Calendarización real del Semestre VII durante 2017/1.

Finalmente, la variable $\mathbf{V_7}$ representa el número de segmentos de 2 o más días consecutivos con evaluaciones. En la calendarización real estos segmentos se dan en las Semanas 6, 10 y 14, mientras que la calendarización óptima, además, en las Semanas 7 y 13. Al comparar la semana 7 de las dos calendarización, la mayor diferencia reside en que la óptima tiene una evaluación adicional, el Control 1 del ramo CC4301, que es precisamente el que genera los días consecutivos de evaluaciones. En la calendarización real, esta evaluación se fijó la semana anterior. Cabe destacar que la calendarización óptima tuvo que postergar dicha evaluación para la Semana 7 ya que de haberlo fijado en la Semana 6, se hubiese excedido su carga semanal máxima.

Caso similar ocurre en la Semana 13. En la calendarización óptima, la Tarea 3 del ramo CC4101 se fijó en la Semana 13, y no en la Semana 14 como ocurre en la calendarización real, porque de ser así se hubiera violado la restricción de carga máxima semanal. Por otro lado, dadas las fechas de calendarización declaradas para cada evaluación (ver Apéndice A.1), en la calendarización óptima la Presentación 3 del ramo CC4401 solo puede fijarse en la Semana 13. Ahora bien, ambas evaluaciones se fijaron en sus primeras opciones (dado que no incumplían ninguna de las restricciones del modelo), terminando calendarizadas en días consecutivos.

En resumen, la razón por la que la calendarización óptima se comporta peor que la real en término de días consecutivos con evaluaciones es porque los criterios que guían la selección de la calendarización óptima (fijar las evaluaciones lo antes posible, respetando la carga máxima diaria y semanal) no penaliza este tipo de comportamientos, haciendo que en algunos casos –como en este Semestre VII– se vuelva recurrente la presencia de días consecutivos con evaluaciones.

Capítulo 6

Discusión

A continuación se discutirá sobre los resultados obtenidos, así como también sobre algunas limitaciones que tiene la solución desarrollada y sus posibles soluciones.

6.1. Discusión de los resultados

Validación de la hipótesis de trabajo

El objetivo del experimento era validar la hipótesis de que *el uso de la herramienta da lugar a calendarizaciones más viables para los estudiantes, con una distribución de cargas más efectiva y racional*. Efectivamente creemos que los resultados observados confirman nuestra hipótesis de trabajo.

En los Semestres V y VI, las calendarizaciones construidas a partir del uso de la herramienta presentan mejores valores que la calendarizaciones reales en el total de las variables analizadas. En algunas de dichas variables la mejora fue sustancial. Por ejemplo, en el Semestre V, la calendarización real presenta 4 semanas con 3 o más evaluaciones (V_1); la calendarización óptima solo una. En ese mismo semestre, la calendarización real presenta un día con una carga total de 42 horas –valor que nos parece simplemente inviable–, mientras que en la calendarización óptima la carga diaria máxima es de 20 horas (V_6). En el Semestre VI, la calendarización óptima redujo la carga semanal máxima de 41 horas a 25 horas (V_7), y la varianza de carga semanal en un 30 % (V_8), lo que se traduce en una carga más balanceada a lo largo del semestre.

Por su parte, en el Semestre VII la calendarización óptima presenta mejores valores que la real en 10 de las 13 variables analizadas (asumiendo que las variables V_3 y V_4 se subdividen en tres variables, cada una). De las tres variables donde la calendarización real se comporta mejor que la óptima, dos –V_1 y V_{31}– se explican porque la calendarización real no sigue las restricciones usadas para generar la calendarización óptima. Concretamente, la calendarización real fija evaluaciones en fecha no declaradas como factible para ello. Esto vuelve la comparación entre ambas calendarizaciones simplemente injusta.

Por último, la tercer variable donde la calendarización real se comporta mejor que la ópti-

Semestre	Valor	f	σ^2_{d}	σ^2_w	ζ
V	Menor	0.00	42.3	46.4	0
	Mayor	0.66	50.4	48.5	0
VI	Menor	0.66	33.1	39.5	1
	Mayor	1.33	33.8	40.8	1
VII	Menor	1.33	52.7	39.3	0
	Mayor	1.66	54.2	40.1	0

Tabla 6.1: Valores extremos de las variables f, σ^2_{d}, σ^2_w y ζ al considerar 30 calendarizaciones óptimas por semestre.

ma es en el número de segmentos de dos o más días consecutivos con evaluaciones (V_7). Esto se debe a que nuestra solución no contempla esta variable al momento de generar la calendarización óptima. Efectivamente creemos que sería buena idea incorporar dicha variable, al menos dentro la información estadística presentada para cada una de las calendarizaciones generadas, así el usuario puede tenerla en cuenta al momento de seleccionar la calendarización óptima dentro de todas las candidatas reportadas.

Relevancia del conjunto de fechas factibles de las evaluaciones

La estrategia de generación de calendarizaciones óptimas es altamente sensible al conjunto de fechas factibles que se declare para las evaluaciones. Prueba de esto es lo ocurrido en el Semestre VII (ver discusión al final de la Sección 5.3): en la calendarización real se fijaron evaluaciones fuera de las fechas declaradas para ello, lo que dio lugar a una calendarización con menos semanas de 3 o más evaluaciones y menos semanas de carga académica mayor a 30 horas.

Recalcamos la importancia de que, al usar esta herramienta, los profesores declaren la mayor cantidad de fechas factibles para sus evaluaciones. En cuanto mayor es este número, mayor es el espacio de calendarizaciones válidas, y por lo tanto, mayor es el potencial de obtener mejores calendarizaciones.

Variabilidad de la calidad de las calendarizaciones óptimas

Como se puede ver en la Tabla 5.1, las *top* 5 calendarizaciones para cada semestre son muy similares entre sí, es más, en el Semestre VII las 4 variables consideradas (f, σ^2_{d}, σ^2_w y ζ) tienen exactamente los mismos valores en las 5 calendarizaciones. Para determinar a partir de qué calendarización empiezan a surgir cambios significativos, se decidió generar 30 calendarizaciones, y determinar el mayor y menor valor de cada una de las 4 variables.

Los resultados obtenidos pueden consultarse en la Tabla 6.1. Vemos que, al igual que al considerar sólo 5 calendarizaciones, al considerar 30 tampoco existen cambios significativos en las variables.

La razón por la que la varianza de carga semanal (σ^2_w) permanece aproximadamente constante es porque, en general, para cada evaluación se tiende a declarar como fechas factibles días de la misma semana. Por su parte, la varianza de carga diaria (σ^2_{d}) tampoco presenta

mucha variabilidad y creemos que es porque un semestre tiene $15 \cdot 7 = 105$ días y los 3 semestres que nosotros analizamos tienen en promedio 22 evaluaciones cada uno. Esto hace que al calcular la varianza de carga diaria, predominan los días en los que no hay evaluaciones –al menos $105 - 22 = 83$– que aquellos en los que sí hay evaluaciones –a lo más 22–.

La cantidad de evaluaciones en días no deseados (ζ) permanece (exactamente) constante, lo que puede deberse a que en dos (Semestre V y VII) de los tres semestres analizadas, no hay días no deseados. En el semestre restante (Semestre VI) hay solo una evaluación (Tarea 2 CC3102) que puede ser calendarizada en un día no deseado (viernes de la Semana 11). La evaluación es efectivamente fijada ese día en todas las calendarizaciones óptimas, ya que de fijarse en su otra fecha factible (lunes de Semana 11), violaría la carga académica diaria máxima (20 horas) de ese día.

6.2. Limitaciones y propuestas de mejora

Discutimos a continuación cuatro limitaciones de la solución desarrollada y para cada una de ellas proponemos posibles líneas de mejora.

Limitación 1. La manera en la que la herramienta fue usada hasta ahora, considerando solo los ramos obligatorios de un semestre de malla, restringe significativamente su utilidad. Esto último, debido a que la mayoría estudiantes, además de los cursos obligatorios, toman otros cursos electivos, los cuales varían mucho de estudiante a estudiante.

Para mejorar la aplicabilidad de la herramienta se propone estudiar posibles grupos o *clusters* de cursos (obligatorios y electivos) que sean tomados simultáneamente con mayor frecuencia por los estudiantes. De esta manera se ampliará el alcance y la utilidad de la herramienta al lograr calendarizaciones más cercanas a las que los estudiantes necesitan, considerando una mayor cantidad de los cursos que toman en un semestre.

Limitación 2. El enfoque que hemos adoptado hasta ahora para cuantificar la calidad de una calendarización presenta dos limitaciones:

i) Asume que un estudiante dedica el 100 % del tiempo de preparación de una evaluación durante el mismo día en que se da la evaluación. Esta suposición no refleja la realidad ya que los estudiantes se preparan con varios días de anticipación para cada evaluación.

ii) Solo considera el tiempo de preparación neto de todas las evaluaciones (diarias y semanales). Sin embargo, la carga "percibida" para un estudiante no es la misma al realizar tres evaluaciones de un tiempo de preparación de 4 horas cada una, que al realizar una única evaluación de un tiempo de preparación de 12 horas. Esto dado que en el primer caso hay que cambiar muchas veces "el enfoque", ya que cada evaluación contempla temas y estrategias de preparación muy diferente.

Para abordar i), al momento de calcular la carga académica de un período de tiempo dado, se propone distribuir el tiempo de preparación de cada evaluación en varios días, por ejemplo, en los 3 días previos de cada evaluación. Para abordar ii), se propone incorporar la

cantidad de evaluaciones por día y por semana al momento de cuantificar la calidad de una calendarización.

Limitación 3. La manera en la que obtenemos las n soluciones (calendarizaciones) óptimas del programa lineal que modela nuestro problema es bastante ineficiente. Esto debido a que la librería usada para implementar el algoritmo *branch-and-cut* devuelve una única solución óptima. Para obtener las restantes $n-1$ soluciones, resolvemos $n-1$ veces el programa lineal desde cero, agregándole en cada iteración una restricción extra que colectivamente se encargan de eliminar las soluciones previamente obtenidas. Esta estrategia genera la duplicidad de muchos cómputos, lo que repercute negativamente en el tiempo de resolución del problema.

Para abordar este inconveniente se propone usar una implementación propia del algoritmo *branch-and-cut*, en desmedro del provisto por la librería empleada. Esta variante del algoritmo exploraría (potencialmente todo) el espacio de soluciones enteras del programa lineal, y llevaría cuenta de las n mejores soluciones encontradas en cada "punto" de la exploración. Su mayor diferencia con respecto al algoritmo original es que podaría una rama solo si no hay posibilidades que genere soluciones mejores que las encontradas hasta ese punto.

Limitación 4. En su estado actual, la solución desarrollada no permite capturar todos los requisitos usados por los profesores a la hora de fijar sus evaluaciones. Por ejemplo, un criterio que no podemos capturar es cuando se desea que todas las tareas de un ramo (generalmente en forma de laboratorios) sean calendarizadas el mismo días de la semana, por ejemplo, o todas el jueves o todas el viernes (de sus respectivas semanas).

Como trabajo futuro se propone analizar con los profesores qué tipo de criterios no pueden ser capturados actualmente por la herramienta, y posteriormente qué tipo de modificaciones requeriría su incorporación al modelo. Un punto crucial acá es si dichos criterios pueden seguir siendo modeladas a través de un programa lineal, o se tiene que recurrir a un modelo de optimización más general.

Capítulo 7

Conclusión

Tener una buena calendarización de los diferentes hitos de evaluación de un semestre universitario es fundamental para que los estudiantes puedan rendir a un buen nivel sin descuidar su vida privada ni su salud. Sin embargo en la práctica, la calendarización de evaluaciones se hace de manera ad hoc, la mayoría de las veces resultando en una distribución de carga inviable para los estudiantes.

En torno a este contexto, el trabajo realizado en esta memoria consistió en modelar dicho problema de calendarización como un programa lineal entero, para hallar las mejores calendarizaciones siguiendo un conjunto de restricciones y función objetivo que se encargan de determinar la viabilidad y cuantificar la calidad de una calendarización.

Para resolver el programa lineal entero y generar las mejores calendarizaciones se utiliza la librería MIP de Python, en particular, su implementación del algoritmo de *branch-and-cut*. Se programó además una interfaz web que se encarga de recibir los datos de entrada del algoritmo y mostrar los resultados en forma amigable, de calendario, con la fecha asignada a cada evaluación.

Ya que no hay una acuerdo generalizado sobre lo que constituye una "buena" calendarización, la herramienta permite al usuario ingresar todos los parámetros usados para cuantificar la calidad y viabilidad de las calendarizaciones, por ejemplo, la carga máxima permitida por día y por semana, medida en cantidad de horas de estudio.

Un gran desafío que tuvimos que abordar fue cómo capturar de manera uniforme la gran diversidad de criterios que los profesores usan al momento de fijar sus evaluaciones. Se decidió que la manera más robusta y general de hacer esto es pedir que "enumeren" todos los días factibles en los que puede fijarse cada una de sus evaluaciones.

Otro problema que apareció desarrollando el modelo fue la elección de la función objetivo. En un principio se propuso una función "multi-objetivo" que consistía en la suma ponderada de la postergación de las evaluaciones respecto al primer día en que se pueden calendarizar, la varianza de carga semanal y diaria, y la cantidad de evaluaciones fijadas en días no deseados. Esto tenía dos problemas. Primero, no estaba claro cómo ponderar cada una de estas cuatro

variables, y segundo, el uso de la varianza de carga en la función objetivo generaría un programa no lineal, lo que es computacionalmente más costoso de resolver. Por tanto se optó por que el modelo optimizara la primera de las cuatro variables mencionadas, y la herramienta devolviera al usuario un *ranking* de las *top-n* calendarizaciones, ordenadas según el resto de las variables. De esta manera, el usuario puede elegir de manera informada la "mejor" calendarización de acuerdo a sus preferencias personales.

Según descrito a lo largo del documento, podemos concluir que se cumplió exitosamente el objetivo general de la memoria. Se desarrolló una herramienta que en base a un modelo matemático determina la calendarización que, respetando los requisitos de cada ramo, organiza la carga académica del semestre de la mejor manera posible.

Los objetivos específicos delineados inicialmente también se alcanzaron exitosamente. En particular, se validó la herramienta desarrollada a partir del análisis tres semestres de la malla de Ingeniería Civil en Computación. Se generaron calendarizaciones óptimas usando la herramienta y se compararon con las calendarizaciones reales que estos tres semestres tuvieron en 2016/1, 2016/2 y 2017/1, respectivamente. Las calendarizaciones óptimas resultaron ser sustancialmente mejores que las reales en dos de los tres semestres. En el semestre restante, la calendarización óptima mejoró a la real en la mayoría de las variables analizadas. En las variables que ocurrió lo inverso, podemos atribuirlo a que la calendarización real no sigue el conjunto de fechas factibles usado para la generación de calendarizaciones óptimas, volviendo la comparación injusta.

Finalmente se proponen líneas de trabajo futuro para abordar algunas limitaciones de la solución desarrollada. Por ejemplo, extender el conjunto de criterios para fijar evaluaciones que ahora permite capturar la herramienta (a través de su estrategia de "enumeración" de días factibles). Otro aspecto a mejorar es el método utilizado para cuantificar la calidad de una calendarización, reflejando que el tiempo que un estudiante invierte para preparar una evaluación se distribuye a lo largo de varios días previos a la evaluación. También se propone una mejora al algoritmo de resolución del programa lineal entero, para que compute las n soluciones óptimas de una manera más eficiente.

Durante el desarrollo de esta memoria se pudo entender el proceso completo de modelar un problema de la vida cotidiana como un programa matemático. También se comprendió la dificultad de definir de manera precisa la noción de "calidad" asociada a la solución de un problema, en este caso, la calidad de una calendarización. Si bien se intenta cuantificar de manera objetiva la calidad de una calendarización a partir de un conjunto de variables numéricas bien definidas, la elección y ponderación de dichas variables sigue siendo subjetiva.

Bibliografía

[1] An, Y., Kim, Y.-D., Jeong, B.-J., and Kim, S.-D. Scheduling healthcare services in a home healthcare system. *Journal of the Operational Research Society 63* (2012), 1589–1599.

[2] Balas, E., Ceria, S., Cornuéjols, G., and Natraj, N. Gomory cuts revisited. *Operations Research Letters 19*, 1 (1996), 1 – 9.

[3] Briskorn, D., and Drexl, A. A branch-and-price algorithm for scheduling sport leagues. *Journal of the Operational Research Society 60* (2009), 84–93.

[4] CNN Chile. Todo lo que necesitas saber para entender el proyecto que busca reducir la jornada laboral a 40 horas, Jul 2019.

[5] de Haan, H. J. Job shop scheduling with metaheuristics for car workshops.

[6] Dimopoulou, M., and Miliotis, P. Implementation of a university course and examination timetabling system. *European Journal of Operational Research 130*, 1 (2001), 202 – 213.

[7] Emol. El reclamo de los alumnos de arquitectura de la u. de chile que desató un intenso debate por la carga académica, Apr 2019.

[8] Gunawan, A., Ng, K. M., and Poh, K.-L. Solving the teacher assignment-course scheduling problem by a hybrid algorithm. *International Journal of Computer, Information, and Systems Science, and Engineering 1*, 2 (2007), 491–496.

[9] Havås, J., Olsson, A., Persson, J., and Schierscher, M. S. Modeling and optimization of university timetabling - A case study in integer programming.

[10] Karloff, H. *Linear Programming*. Birkhauser Boston Inc., 1991.

[11] La Tercera. El estrés es un problema, no un valor cultural, Feb 2020.

[12] Liaw, C.-F. A tabu search algorithm for the open shop scheduling problem. *Comput. Oper. Res. 26* (1999), 109–126.

[13] Mansouri, N., Kadri, I., Mazouz, S., Bouchenak, A., Abdelaziz, S., Benrabia, M., Atlaoui, S., and Attafi, I. Cutting plane method, 2018.

[14] Mushi, A. Tabu search heuristic for university course timetabling problem. *African Journal of Science and Technology 7* (2010).

[15] Recalde, D., Torres, R., and Vaca, P. Scheduling the professional ecuadorian football league by integer programming. *Computers & Operations Research 40*, 10 (2013), 2478–2484.

[16] Tutoriales, G. Ejemplo del algoritmo de branch and bound (ramificación y acotamiento), Jul 2016.

[17] Wren, A., and Wren, D. O. A genetic algorithm for public transport driver scheduling. *Computers & Operations Research 22*, 1 (1995), 101 – 110. Genetic Algorithms.

Apéndice A

A.1. Información para la generación de calendarizaciones óptimas

A continuación describimos la información necesaria para generar las calendarizaciones óptimas de los Semestres V, VI y VII de la malla Ingeniería Civil en Computación. Para cada evaluación de los ramos obligatorios de dichos semestres, se incluye: tipo y número, tiempo de preparación, y conjunto de días en los que puede ser calendarizada.

Los datos fueron recolectados contactando por correo a los profesores que dictan el ramo. Como no toda la información pudo ser obtenida de esta manera, el resto se completó de manera manual, considerando las evaluaciones de los semestres reales (Semestre V cursado en Otoño 2016, Semestre VI cursado en Primavera 2016 y Semestre VII cursado en Otoño 2017) y extrayendo datos de cursos similares a estos. Los datos que aparecen con color negro son datos provistos por el profesor. Los de color azul son datos que se obtuvieron de semestres reales antes mencionados y se utilizaron para completar los datos faltantes. Por último, los de color rojo son datos que no se entregaron por parte del profesor y que no pudieron ser obtenidos de los semestres reales anteriores y que se completaron considerando los modelos de los datos entregados por los profesores.

Semestre V

Matemáticas discretas para la computación - CC3101

- Control1 [10h] = {(5,5), (5,6)}
- Control2 [10h] = {(5,9), (5,10)}
- Control3 [10h] = {(4,13), (5,13)}
- Tarea1 [12h] = {(3,5), (4,5), (5,5), (1,6), (2,6), (3,6)}
- Tarea2 [12h] = {(3,8), (4,8), (5,8), (1,9), (2,9), (3,9)}
- Tarea3 [12h] = {(3,12), (4,12), (5,12)}

Algoritmos y estructuras de datos - CC3001

- Control1 [8h] = {(5, 6), (5, 7)}

- Control2 [8h] = {(5, 13), (5, 14)}
- Tarea1 [10h] = {(3, 4), (4, 4)}
- Tarea2 [10h] = {(3, 6), (4, 6)}
- Tarea3 [10h] = {(7, 8), (1, 9)}
- Tarea4 [10h] = {(7, 10), (1, 11)}
- Tarea5 [10h] = {(1, 13), (2, 13)}
- Tarea6 [10h] = {(7, 14), (1, 15)}

Modelación y computación gráfica para ingenieros - CC3501

- Control1 [8h] = {(5, 4), (5, 5)}
- Control2 [8h] = {(5, 13), (5, 14)}
- Tarea1 [20h] = {(1, 7), (2, 7), (3, 7), (4, 7), (5, 7), (1, 8), (2, 8), (3, 8), (4, 8), (5, 8)}
- Tarea2 [20h] = {(1, 10), (2, 10), (3, 10), (4, 10), (5, 10), (1, 11), (2, 11), (3, 11), (4, 11), (5, 11)}
- Tarea3 [20h] = {(1, 14), (2, 14), (3, 14), (4, 14), (5, 14), (1, 15), (2, 15), (3, 15), (4, 15), (5, 15)}

Semestre VI

Teoría de la computación - CC3102

- Control1 [8h] = {(3, 6), (4, 6)}
- Control2 [8h] = {(3, 9), (4, 9)}
- Control3 [8h] = {(3, 14), (4, 14)}
- Tarea1 [20h] = {(5, 7), (1, 8)}
- Tarea2 [20h] = {(5, 10), (1, 11)}
- Tarea3 [20h] = {(1, 15), (5, 15)}

Base de datos - CC3201

- Ejercicio1 [4h] = {(1, 3), (2, 3), (3, 3), (4, 3), (5, 3)}
- Ejercicio2 [4h] = {(1, 4), (2, 4), (3, 4),(4, 4), (5, 4)}
- Ejercicio3 [4h] = {(1, 5), (2, 5), (3, 5), (4, 4), (5, 5)}
- Ejercicio4 [4h] = {(1, 6), (2, 6), (3, 6), (4, 6), (5, 6)}
- Ejercicio5 [4h] = {(1, 7), (2, 7), (3, 7), (4, 7), (5, 7)}
- Ejercicio6 [4h] = {(1, 8), (2, 8), (3, 8), (4, 8), (5, 8)}
- Ejercicio7 [4h] = {(1, 9), (2, 9), (3, 9), (4, 9), (5, 9)}
- Ejercicio8 [4h] = {(1, 10), (2, 10), (3, 10), (4, 10), (5, 10)}
- Ejercicio9 [4h] = {(1, 11), (2,11), (3, 11), (4, 11), (5, 11)}
- Ejercicio10 [4h] = {(1, 12), (2, 12), (3, 12), (4, 12), (5, 12)}
- Ejercicio11 [4h] = {(1, 13), (2, 13), (3, 13), (4, 13), (5,13)}

- Tarea1 [12h] = {(1, 13), (2, 13), (3, 13), (4, 13), (5, 13), (1, 14), (2, 14), (3, 14), (4, 14), (5, 14)}

Metodologías de diseño y programación - CC3002

- Control1 [2h] = {(3, 6), (1, 7)}
- Control2 [2h] = {(1, 12), (3, 12)}
- Control3 [2h] = {(1, 14), (3, 14)}
- Tarea1 [9h] = {(3, 4), (1, 5)}
- Tarea2 [9h] = {(1, 12), (3, 12), (1, 13)}

Semestre VII

Lenguajes de programación - CC4101

- Control1 [12h] = {(5,6), (5,7)}
- Control2 [12h] = {(5,13), (5,14)}
- Tarea1 [10h] = {(3,5), (4,5), (5,5)}
- Tarea2 [14h] = {(3,9), (4,9), (5,9)}
- Tarea3 [14h] = {(3,13), (4,13), (5,13), (1,14), (2,14), (3,14), (4,14), (5,14)}

Ingeniería de software - CC4401

- Control1 [3h] = {(2, 7), (4, 7)}
- Control2 [3h] = {(2, 14), (4, 14)}
- Presentación1 [15h] = {(2, 6), (4, 6)}
- Presentación2 [15h] = {(2, 10), (4, 10)}
- Presentación3 [15h] = {(2, 13), (4, 13)}

Arquitectura de computadores - CC4301

- Tarea1 [10h] = {(1, 6), (2, 6), (3, 6), (4,6), (5, 6)}
- Tarea2 [10h] = {(1, 14), (2, 14), (3, 14), (4, 14), (5, 14)}
- Control1 [12h] = {(3, 6), (1, 7)}
- Control2 [12h] = {(1, 10), (3, 10)}
- Control3 [12h] = {(1, 14), (3, 14), (5, 14)}
- Ejercicio1 [4h] = {(3, 2), (4, 2), (5, 2), (1, 3)}
- Ejercicio2 [4h] = {(2, 3), (3, 3), (4, 3), (5, 3), (1, 4)}
- Ejercicio3 [4h] = {(2, 4), (3, 4), (4, 4), (5, 4), (1, 5)}
- Ejercicio4 [4h] = {(1, 7), (2, 7), (3, 7), (4, 7), (5, 7)}
- Ejercicio5 [4h] = {(1, 10), (2, 10), (3, 10), (4, 10), (5, 10)}
- Ejercicio6 [4h] = {(1, 12), (2, 12), (3, 12), (4, 12), (5, 12)}

A.2. Calendarizaciones óptimas

En esta sección se presentan las 5 mejores calendarizaciones generadas por la herramienta para los Semestre V, VI y VII de la malla de Ingeniería Civil en Computación. En la tablas, los días feriados se indican coloreando la celda respectiva de color rojo y los días no deseados de color amarillo.

Semestre V

	LU	MA	MI	JU	VI	SA	DO
S-1							
S-2							
S-3							
S-4			CC3001-T1[10h]		CC3501-C1[8h]		
S-5			CC3101-T1[12h]		CC3101-C1[10h]		
S-6			CC3001-T2[10h]		CC3001-C1[8h]		
S-7	CC3501-T1[20h]						
S-8			CC3101-T2[12h]				CC3001-T3[10h]
S-9					CC3101-C2[10h]		
S-10	CC3501-T2[20h]						CC3001-T4[10h]
S-11							
S-12			CC3101-T3[12h]				
S-13	CC3001-T5[10h]			CC3101-C3[10h]	CC3001-C2[8h] CC3501-C2[8h]		
S-14	CC3501-T3[20h]						CC3001-T6[10h]
S-15							

Tabla A.1: Calendarización 1 devuelta por la herramienta para el Semestre V de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO	Carga académica semanal
S-1								0 hrs
S-2								0 hrs
S-3								0 hrs
S-4			10 hrs		8 hrs			18 hrs
S-5			12 hrs		10 hrs			22 hrs
S-6			10 hrs		8 hrs			18 hrs
S-7	20 hrs							20 hrs
S-8			12 hrs				10 hrs	22 hrs
S-9					10 hrs			10 hrs
S-10	20 hrs						10 hrs	30 hrs
S-11								0 hrs
S-12			12 hrs					12 hrs
S-13	10 hrs			10 hrs	16 hrs			36 hrs
S-14	20 hrs						10 hrs	30 hrs
S-15								0 hrs

Tabla A.2: Carga académica asociada a la Calendarización 1 devuelta por la herramienta para el Semestre V de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO
S-1							
S-2							
S-3							
S-4			CC3001-T1[10h]		CC3501-C1[8h]		
S-5			CC3101-T1[12h]		CC3101-C1[10h]		
S-6			CC3001-T2[10h]		CC3001-C1[8h]		
S-7	CC3501-T1[20h]						
S-8			CC3101-T2[12h]				CC3001-T3[10h]
S-9					CC3101-C2[10h]		
S-10	CC3501-T2[20h]						CC3001-T4[10h]
S-11							
S-12				CC3101-T3[12h]			
S-13	CC3001-T5[10h]			CC3101-C3[10h]	CC3001-C2[8h] CC3501-C2[8h]		
S-14	CC3501-T3[20h]						CC3001-T6[10h]
S-15							

Tabla A.3: Calendarización 2 devuelta por la herramienta para el Semestre V de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO	Carga académica semanal
S-1								0 hrs
S-2								0 hrs
S-3								0 hrs
S-4			10 hrs		8 hrs			18 hrs
S-5			12 hrs		10 hrs			22 hrs
S-6			10 hrs		8 hrs			18 hrs
S-7	20 hrs							20 hrs
S-8			12 hrs				10 hrs	22 hrs
S-9					10 hrs			10 hrs
S-10	20 hrs						10 hrs	30 hrs
S-11								0 hrs
S-12				12 hrs				12 hrs
S-13	10 hrs			10 hrs	16 hrs			36 hrs
S-14	20 hrs						10 hrs	30 hrs
S-15								0 hrs

Tabla A.4: Carga académica asociada a la Calendarización 2 devuelta por la herramienta para el Semestre V de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO
S-1							
S-2							
S-3							
S-4			CC3001-T1[10h]		CC3501-C1[8h]		
S-5			CC3101-T1[12h]		CC3101-C1[10h]		
S-6			CC3001-T2[10h]		CC3001-C1[8h]		
S-7	CC3501-T1[20h]						
S-8			CC3101-T2[12h]				CC3001-T3[10h]
S-9					CC3101-C2[10h]		
S-10	CC3501-T2[20h]						CC3001-T4[10h]
S-11							
S-12			CC3101-T3[12h]				
S-13		CC3001-T5[10h]		CC3101-C3[10h]	CC3001-C2[8h] CC3501-C2[8h]		
S-14	CC3501-T3[20h]						CC3001-T6[10h]
S-15							

Tabla A.5: Calendarización 3 devuelta por la herramienta para el Semestre V de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO	Carga académica semanal
S-1								0 hrs
S-2								0 hrs
S-3								0 hrs
S-4			10 hrs		8 hrs			18 hrs
S-5			12 hrs		10 hrs			22 hrs
S-6			10 hrs		8 hrs			18 hrs
S-7	20 hrs							20 hrs
S-8			12 hrs				10 hrs	22 hrs
S-9					10 hrs			10 hrs
S-10	20 hrs						10 hrs	30 hrs
S-11								0 hrs
S-12			12 hrs					12 hrs
S-13		10 hrs		10 hrs	16 hrs			36 hrs
S-14	20 hrs						10 hrs	30 hrs
S-15								0 hrs

Tabla A.6: Carga académica asociada a la Calendarización 3 devuelta por la herramienta para el Semestre V de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO
S-1							
S-2							
S-3							
S-4			CC3001-T1[10h]		CC3501-C1[8h]		
S-5			CC3101-T1[12h]		CC3101-C1[10h]		
S-6			CC3001-T2[10h]		CC3001-C1[8h]		
S-7	CC3501-T1[20h]						
S-8			CC3101-T2[12h]				CC3001-T3[10h]
S-9					CC3101-C2[10h]		
S-10	CC3501-T2[20h]						
S-11	CC3001-T4[10h]						
S-12			CC3101-T3[12h]				
S-13	CC3001-T5[10h]			CC3101-C3[10h]	CC3001-C2[8h] CC3501-C2[8h]		
S-14	CC3501-T3[20h]						CC3001-T6[10h]
S-15							

Tabla A.7: Calendarización 4 devuelta por la herramienta para el Semestre V de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO	Carga académica semanal
S-1								0 hrs
S-2								0 hrs
S-3								0 hrs
S-4			10 hrs		8 hrs			18 hrs
S-5			12 hrs		10 hrs			22 hrs
S-6			10 hrs		8 hrs			18 hrs
S-7	20 hrs							20 hrs
S-8			12 hrs				10 hrs	22 hrs
S-9					10 hrs			10 hrs
S-10	20 hrs							20 hrs
S-11	10 hrs							10 hrs
S-12			12 hrs					12 hrs
S-13	10 hrs			10 hrs	16 hrs			36 hrs
S-14	20 hrs						10 hrs	30 hrs
S-15								0 hrs

Tabla A.8: Carga académica asociada a la Calendarización 4 devuelta por la herramienta para el Semestre V de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO
S-1							
S-2							
S-3							
S-4				CC3001-T1[10h]	CC3501-C1[8h]		
S-5			CC3101-T1[12h]		CC3101-C1[10h]		
S-6			CC3001-T2[10h]		CC3001-C1[8h]		
S-7	CC3501-T1[20h]						
S-8			CC3101-T2[12h]				CC3001-T3[10h]
S-9					CC3101-C2[10h]		
S-10	CC3501-T2[20h]						CC3001-T4[10h]
S-11							
S-12			CC3101-T3[12h]				
S-13	CC3001-T5[10h]			CC3101-C3[10h]	CC3001-C2[8h] CC3501-C2[8h]		
S-14	CC3501-T3[20h]						CC3001-T6[10h]
S-15							

Tabla A.9: Calendarización 5 devuelta por la herramienta para el Semestre V de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO	Carga académica semanal
S-1								0 hrs
S-2								0 hrs
S-3								0 hrs
S-4				10 hrs	8 hrs			18 hrs
S-5			12 hrs		10 hrs			22 hrs
S-6			10 hrs		8 hrs			18 hrs
S-7	20 hrs							20 hrs
S-8			12 hrs				10 hrs	22 hrs
S-9					10 hrs			10 hrs
S-10	20 hrs						10 hrs	30 hrs
S-11								0 hrs
S-12			12 hrs					12 hrs
S-13	10 hrs			10 hrs	16 hrs			36 hrs
S-14	20 hrs						10 hrs	30 hrs
S-15								0 hrs

Tabla A.10: Carga académica asociada a la Calendarización 5 devuelta por la herramienta para el Semestre V de la malla Ingeniería Civil en Computación.

Semestre VI

	LU	MA	MI	JU	VI	SA	DO
S-1							
S-2							
S-3	CC3201-E1[4h]						
S-4	CC3201-E2[4h]		CC3002-T1[9h]				
S-5	CC3201-E3[4h]						
S-6	CC3201-E4[4h]		CC3002-C1[2h] CC3102-C1[8h]				
S-7	CC3201-E4[4h]				CC3102-T1[20h]		
S-8	CC3201-E6[4h]						
S-9		CC3201-E7[4h]	CC3102-C2[8h]				
S-10	CC3201-E8[4h]				CC3102-T2[20h]		
S-11	CC3201-E9[4h]						
S-12			CC3201-E10[4h] CC3002-C2[2h]				
S-13	CC3201-E11[4h] CC3002-T2[9h]	CC3201-T1[12h]					
S-14	CC3002-C3[2h]		CC3102-C3[8h]				
S-15	CC3102-T3[20h]						

Tabla A.11: Calendarización 1 devuelta por la herramienta para el Semestre VI de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO	Carga académica semanal
S-1								0 hrs
S-2								0 hrs
S-3	4 hrs							4 hrs
S-4	4 hrs		9 hrs					13 hrs
S-5	4 hrs							4 hrs
S-6	4 hrs		10 hrs					14 hrs
S-7	4 hrs				20 hrs			24 hrs
S-8	4 hrs							4 hrs
S-9		4 hrs	8 hrs					12 hrs
S-10	4 hrs				20 hrs			24 hrs
S-11	4 hrs							4 hrs
S-12			6 hrs					6 hrs
S-13	13 hrs		12 hrs					25 hrs
S-14	2 hrs		8 hrs					10 hrs
S-15	20 hrs							20 hrs

Tabla A.12: Carga académica asociada a la Calendarización 1 devuelta por la herramienta para el Semestre VI de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO
S-1							
S-2							
S-3	CC3201-E1[4h]						
S-4	CC3201-E2[4h]		CC3002-T1[9h]				
S-5	CC3201-E3[4h]						
S-6	CC3201-E4[4h]		CC3002-C1[2h] CC3102-C1[8h]				
S-7	CC3201-E4[4h]				CC3102-T1[20h]		
S-8	CC3201-E6[4h]						
S-9		CC3201-E7[4h]	CC3102-C2[8h]				
S-10	CC3201-E8[4h]				CC3102-T2[20h]		
S-11	CC3201-E9[4h]						
S-12			CC3201-E10[4h] CC3002-C2[2h]				
S-13	CC3201-E11[4h] CC3002-T2[9h]	CC3201-T1[12h]					
S-14			CC3102-C3[8h] CC3002-C3[2h]				
S-15	CC3102-T3[20h]						

Tabla A.13: Calendarización 2 devuelta por la herramienta para el Semestre VI de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO	Carga académica semanal
S-1								0 hrs
S-2								0 hrs
S-3	4 hrs							4 hrs
S-4	4 hrs		9 hrs					13 hrs
S-5	4 hrs							4 hrs
S-6	4 hrs		10 hrs					14 hrs
S-7	4 hrs				20 hrs			24 hrs
S-8	4 hrs							4 hrs
S-9		4 hrs	8 hrs					12 hrs
S-10	4 hrs				20 hrs			24 hrs
S-11	4 hrs							4 hrs
S-12			6 hrs					6 hrs
S-13	13 hrs		12 hrs					25 hrs
S-14			10 hrs					10 hrs
S-15	20 hrs							20 hrs

Tabla A.14: Carga académica asociada a la Calendarización 2 devuelta por la herramienta para el Semestre VI de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO
S-1							
S-2							
S-3	CC3201-E1[4h]						
S-4	CC3201-E2[4h]		CC3002-T1[9h]				
S-5		CC3201-E3[4h]					
S-6	CC3201-E4[4h]		CC3002-C1[2h] CC3102-C1[8h]				
S-7	CC3201-E4[4h]				CC3102-T1[20h]		
S-8	CC3201-E6[4h]						
S-9		CC3201-E7[4h]	CC3102-C2[8h]				
S-10	CC3201-E8[4h]				CC3102-T2[20h]		
S-11	CC3201-E9[4h]						
S-12			CC3201-E10[4h] CC3002-C2[2h]				
S-13	CC3201-E11[4h] CC3002-T2[9h]	CC3201-T1[12h]					
S-14	CC3002-C3[2h]		CC3102-C3[8h]				
S-15	CC3102-T3[20h]						

Tabla A.15: Calendarización 3 devuelta por la herramienta para el Semestre VI de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO	Carga académica semanal
S-1								0 hrs
S-2								0 hrs
S-3	4 hrs							4 hrs
S-4	4 hrs		9 hrs					13 hrs
S-5		4 hrs						4 hrs
S-6	4 hrs		10 hrs					14 hrs
S-7	4 hrs				20 hrs			24 hrs
S-8	4 hrs							4 hrs
S-9		4 hrs	8 hrs					12 hrs
S-10	4 hrs				20 hrs			24 hrs
S-11	4 hrs							4 hrs
S-12			6 hrs					6 hrs
S-13	13 hrs		12 hrs					25 hrs
S-14	2 hrs		8 hrs					10 hrs
S-15	20 hrs							20 hrs

Tabla A.16: Carga académica asociada a la Calendarización 3 devuelta por la herramienta para el Semestre VI de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO
S-1							
S-2							
S-3	CC3201-E1[4h]						
S-4	CC3201-E2[4h]		CC3002-T1[9h]				
S-5	CC3201-E3[4h]						
S-6		CC3201-E4[4h]	CC3002-C1[2h] CC3102-C1[8h]				
S-7	CC3201-E4[4h]				CC3102-T1[20h]		
S-8	CC3201-E6[4h]						
S-9		CC3201-E7[4h]	CC3102-C2[8h]				
S-10	CC3201-E8[4h]				CC3102-T2[20h]		
S-11	CC3201-E9[4h]						
S-12			CC3201-E10[4h] CC3002-C2[2h]				
S-13	CC3201-E11[4h] CC3002-T2[9h]	CC3201-T1[12h]					
S-14	CC3002-C3[2h]		CC3102-C3[8h]				
S-15	CC3102-T3[20h]						

Tabla A.17: Calendarización 4 devuelta por la herramienta para el Semestre VI de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO	Carga académica semanal
S-1								0 hrs
S-2								0 hrs
S-3	4 hrs							4 hrs
S-4	4 hrs		9 hrs					13 hrs
S-5	4 hrs							4 hrs
S-6		4 hrs	10 hrs					14 hrs
S-7	4 hrs				20 hrs			24 hrs
S-8	4 hrs							4 hrs
S-9		4 hrs	8 hrs					12 hrs
S-10	4 hrs				20 hrs			24 hrs
S-11	4 hrs							4 hrs
S-12			6 hrs					6 hrs
S-13	13 hrs		12 hrs					25 hrs
S-14	2 hrs		8 hrs					10 hrs
S-15	20 hrs							20 hrs

Tabla A.18: Carga académica asociada a la Calendarización 4 devuelta por la herramienta para el Semestre VI de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO
S-1							
S-2							
S-3	CC3201-E1[4h]						
S-4	CC3201-E2[4h]		CC3002-T1[9h]				
S-5	CC3201-E3[4h]						
S-6	CC3201-E4[4h]		CC3102-C1[8h]				
S-7	CC3201-E4[4h] CC3002-C1[2h]				CC3102-T1[20h]		
S-8	CC3201-E6[4h]						
S-9		CC3201-E7[4h]	CC3102-C2[8h]				
S-10	CC3201-E8[4h]				CC3102-T2[20h]		
S-11	CC3201-E9[4h]						
S-12			CC3201-E10[4h] CC3002-C2[2h]				
S-13	CC3201-E11[4h] CC3002-T2[9h]	CC3201-T1[12h]					
S-14	CC3002-C3[2h]		CC3102-C3[8h]				
S-15	CC3102-T3[20h]						

Tabla A.19: Calendarización 5 devuelta por la herramienta para el Semestre VI de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO	Carga académica semanal
S-1								0 hrs
S-2								0 hrs
S-3	4 hrs							4 hrs
S-4	4 hrs		9 hrs					13 hrs
S-5	4 hrs							4 hrs
S-6	4 hrs		8 hrs					12 hrs
S-7	6 hrs				20 hrs			26 hrs
S-8	4 hrs							4 hrs
S-9		4 hrs	8 hrs					12 hrs
S-10	4 hrs				20 hrs			24 hrs
S-11	4 hrs							4 hrs
S-12			6 hrs					6 hrs
S-13	13 hrs		12 hrs					25 hrs
S-14	2 hrs		8 hrs					10 hrs
S-15	20 hrs							20 hrs

Tabla A.20: Carga académica asociada a la Calendarización 5 devuelta por la herramienta para el Semestre VI de la malla Ingeniería Civil en Computación.

Semestre VII

	LU	MA	MI	JU	VI	SA	DO
S-1							
S-2			CC4301-E1[4h]				
S-3		CC4301-E2[4h]					
S-4		CC4301-E3[4h]					
S-5			CC4101-T1[10h]				
S-6	CC4301-T1[10h]			CC4401-P1[15h]	CC4101-C1[12h]		
S-7	CC4301-C1[12h]	CC4301-E4[4h] CC4401-C1[3h]					
S-8							
S-9			CC4101-T2[14h]				
S-10	CC4301-E5[4h]		CC4301-C2[12h]	CC4401-P2[15h]			
S-11							
S-12	CC4301-E6[4h]						
S-13		CC4401-P3[15h]	CC4101-T3[14h]				
S-14	CC4301-C3[12h]	CC4401-C2[3h] CC4301-T2[10h]			CC4101-C2[12h]		
S-15							

Tabla A.21: Calendarización 1 devuelta por la herramienta para el Semestre VII de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO	Carga académica semanal
S-1								0 hrs
S-2			4 hrs					4 hrs
S-3		4 hrs						4 hrs
S-4		4 hrs						4 hrs
S-5			10 hrs					10 hrs
S-6	10 hrs			15 hrs	12 hrs			37 hrs
S-7	12 hrs	7 hrs						19 hrs
S-8								0 hrs
S-9			14 hrs					14 hrs
S-10	4 hrs		12 hrs	15 hrs				31 hrs
S-11								0 hrs
S-12	4 hrs							4 hrs
S-13		15 hrs	14 hrs					29 hrs
S-14	12 hrs	13 hrs			12 hrs			37 hrs
S-15								0 hrs

Tabla A.22: Carga académica asociada a la Calendarización 1 devuelta por la herramienta para el Semestre VII de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO
S-1							
S-2			CC4301-E1[4h]				
S-3		CC4301-E2[4h]					
S-4		CC4301-E3[4h]					
S-5			CC4101-T1[10h]				
S-6	CC4301-T1[10h]			CC4401-P1[15h]	CC4101-C1[12h]		
S-7	CC4301-C1[12h]	CC4301-E4[4h] CC4401-C1[3h]					
S-8							
S-9			CC4101-T2[14h]				
S-10	CC4301-E5[4h]	CC4401-P2[15h]	CC4301-C2[12h]				
S-11							
S-12	CC4301-E6[4h]						
S-13		CC4401-P3[15h]	CC4101-T3[14h]				
S-14	CC4301-C3[12h]	CC4401-C2[3h] CC4301-T2[10h]			CC4101-C2[12h]		
S-15							

Tabla A.23: Calendarización 2 devuelta por la herramienta para el Semestre VII de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO	Carga académica semanal
S-1								0 hrs
S-2			4 hrs					4 hrs
S-3		4 hrs						4 hrs
S-4		4 hrs						4 hrs
S-5			10 hrs					10 hrs
S-6	10 hrs			15 hrs	12 hrs			37 hrs
S-7	12 hrs	7 hrs						19 hrs
S-8								0 hrs
S-9			14 hrs					14 hrs
S-10	4 hrs	15 hrs	12 hrs					31 hrs
S-11								0 hrs
S-12	4 hrs							4 hrs
S-13		15 hrs	14 hrs					29 hrs
S-14	12 hrs	13 hrs			12 hrs			37 hrs
S-15								0 hrs

Tabla A.24: Carga académica asociada a la Calendarización 2 devuelta por la herramienta para el Semestre VII de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO
S-1							
S-2			CC4301-E1[4h]				
S-3		CC4301-E2[4h]					
S-4		CC4301-E3[4h]					
S-5			CC4101-T1[10h]				
S-6	CC4301-T1[10h]			CC4401-P1[15h]	CC4101-C1[12h]		
S-7	CC4301-C1[12h]	CC4301-E4[4h] CC4401-C1[3h]					
S-8							
S-9			CC4101-T2[14h]				
S-10	CC4301-C2[12h]	CC4301-E5[4h]		CC4401-P2[15h]			
S-11							
S-12	CC4301-E6[4h]						
S-13		CC4401-P3[15h]	CC4101-T3[14h]				
S-14	CC4301-C3[12h]	CC4401-C2[3h] CC4301-T2[10h]			CC4101-C2[12h]		
S-15							

Tabla A.25: Calendarización 3 devuelta por la herramienta para el Semestre VII de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO	Carga académica semanal
S-1								0 hrs
S-2			4 hrs					4 hrs
S-3		4 hrs						4 hrs
S-4		4 hrs						4 hrs
S-5			10 hrs					10 hrs
S-6	10 hrs			15 hrs	12 hrs			37 hrs
S-7	12 hrs	7 hrs						19 hrs
S-8								0 hrs
S-9			14 hrs					14 hrs
S-10	12 hrs	4 hrs		15 hrs				31 hrs
S-11								0 hrs
S-12	4 hrs							4 hrs
S-13		15 hrs	14 hrs					29 hrs
S-14	12 hrs	13 hrs			12 hrs			37 hrs
S-15								0 hrs

Tabla A.26: Carga académica asociada a la Calendarización 3 devuelta por la herramienta para el Semestre VII de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO
S-1							
S-2			CC4301-E1[4h]				
S-3		CC4301-E2[4h]					
S-4		CC4301-E3[4h]					
S-5			CC4101-T1[10h]				
S-6	CC4301-T1[10h]	CC4401-P1[15h]			CC4101-C1[12h]		
S-7	CC4301-C1[12h]	CC4301-E4[4h] CC4401-C1[3h]					
S-8							
S-9			CC4101-T2[14h]				
S-10	CC4301-C2[12h]	CC4301-E5[4h]		CC4401-P2[15h]			
S-11							
S-12	CC4301-E6[4h]						
S-13		CC4401-P3[15h]	CC4101-T3[14h]				
S-14	CC4301-C3[12h]	CC4401-C2[3h] CC4301-T2[10h]			CC4101-C2[12h]		
S-15							

Tabla A.27: Calendarización 4 devuelta por la herramienta para el Semestre VII de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO	Carga académica semanal
S-1								0 hrs
S-2			4 hrs					4 hrs
S-3		4 hrs						4 hrs
S-4		4 hrs						4 hrs
S-5			10 hrs					10 hrs
S-6	10 hrs	15 hrs			12 hrs			37 hrs
S-7	12 hrs	7 hrs						19 hrs
S-8								0 hrs
S-9			14 hrs					14 hrs
S-10	12 hrs	4 hrs		15 hrs				31 hrs
S-11								0 hrs
S-12	4 hrs							4 hrs
S-13		15 hrs	14 hrs					29 hrs
S-14	12 hrs	13 hrs			12 hrs			37 hrs
S-15								0 hrs

Tabla A.28: Carga académica asociada a la Calendarización 4 devuelta por la herramienta para el Semestre VII de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO
S-1							
S-2			CC4301-E1[4h]				
S-3		CC4301-E2[4h]					
S-4		CC4301-E3[4h]					
S-5			CC4101-T1[10h]				
S-6	CC4301-T1[10h]	CC4401-P1[15h]			CC4101-C1[12h]		
S-7	CC4301-C1[12h]	CC4301-E4[4h] CC4401-C1[3h]					
S-8							
S-9			CC4101-T2[14h]				
S-10	CC4301-C2[12h]	CC4301-E5[4h]		CC4401-P2[15h]			
S-11							
S-12	CC4301-E6[4h]						
S-13			CC4101-T3[14h]	CC4401-P3[15h]			
S-14	CC4301-C3[12h]	CC4401-C2[3h] CC4301-T2[10h]			CC4101-C2[12h]		
S-15							

Tabla A.29: Calendarización 5 devuelta por la herramienta para el Semestre VII de la malla Ingeniería Civil en Computación.

	LU	MA	MI	JU	VI	SA	DO	Carga académica semanal
S-1								0 hrs
S-2			4 hrs					4 hrs
S-3		4 hrs						4 hrs
S-4		4 hrs						4 hrs
S-5			10 hrs					10 hrs
S-6	10 hrs	15 hrs			12 hrs			37 hrs
S-7	12 hrs	7 hrs						19 hrs
S-8								0 hrs
S-9			14 hrs					14 hrs
S-10	12 hrs	4 hrs		15 hrs				31 hrs
S-11								0 hrs
S-12	4 hrs							4 hrs
S-13			14 hrs	15 hrs				29 hrs
S-14	12 hrs	13 hrs			12 hrs			37 hrs
S-15								0 hrs

Tabla A.30: Carga académica asociada a la Calendarización 5 devuelta por la herramienta para el Semestre VII de la malla Ingeniería Civil en Computación.

A.3. Calendarizaciones reales

En esta sección se presentan las calendarizaciones reales que tuvieron los Semestre V, VI y VII de la malla de Ingeniería Civil en Computación durante 2016/1, 2016/2 y 2017/1, respectivamente. En la tablas, los días feriados se indican coloreando la celda respectiva de color rojo y los días no deseados de color amarillo. Para el tiempo de preparación de las evaluaciones se usarán los datos recogidos en la Sección A.1.

Semestre V

	LU	MA	MI	JU	VI	SA	DO
S-1							
S-2							
S-3							
S-4				CC3001-T1[10h]			CC3501-T1[20h]
S-5			CC3101-C1[10h]				
S-6		CC3101-T1[12h]		CC3501-C1[8h] CC3001-T2[10h]			
S-7					CC3001-C1[8h]		
S-8							
S-9	CC3001-T3[10h]			CC3501-T2[20h]			CC3101-T2[12h]
S-10			CC3101-C2[10h]	CC3501-C2[8h]			
S-11	CC3001-T4[10h]						
S-12							
S-13		CC3001-T5[10h]					
S-14			CC3101-C2[10h]	CC5001-C3[8h]	CC3001-C2[8h]		
S-15	CC3001-T6[10h] CC3101-T3[12h] CC3501-T3[20h]						

Tabla A.31: Calendarización del Semestre V de la malla de Ingeniería Civil en Computación durante 2016/1.

	LU	MA	MI	JU	VI	SA	DO	Carga académica semanal
S-1								0 hrs
S-2								0 hrs
S-3								0 hrs
S-4				10 hrs			20 hrs	30 hrs
S-5			10 hrs					10 hrs
S-6		12 hrs		18 hrs				30 hrs
S-7					8 hrs			8 hrs
S-8								0 hrs
S-9	10 hrs			20 hrs			12 hrs	42 hrs
S-10			10 hrs	8 hrs				18 hrs
S-11	10 hrs							10 hrs
S-12								0 hrs
S-13		10 hrs						10 hrs
S-14			10 hrs	8 hrs	8 hrs			26 hrs
S-15	42 hrs							42 hrs

Tabla A.32: Carga académica del Semestre V de la malla de Ingeniería Civil en Computación durante 2016/1.

Semestre VI

	LU	MA	MI	JU	VI	SA	DO
S-1							
S-2		CC3201-E1[4h]					
S-3		CC3201-E2[4h]					
S-4			CC3002-C1[2h]				CC3002-T1[9h]
S-5		CC3201-E3[4h] CC3201-E4[4h]					
S-6		CC3102-C1[8h]	CC3201-E5[4h]			CC3102-T1[20h]	
S-7	CC3201-C1[8h]		CC3201-E6[4h]				
S-8		CC3201-E7[4h]	CC3102-C2[8h]				
S-9		CC3201-E8[4h]	CC3002-C2[2h]				CC3002-T2[9h]
S-10			CC3201-E9[4h]				
S-11							
S-12				CC3201-E10[4h] CC3201-E11[4h]	CC3102-T2[20h]		
S-13	CC3201-C2[8h] CC3002-C3[2h]		CC3102-C3[8h]		CC3201-E12[4h]		
S-14		CC3002-T3[9h]			CC3201-T1[12h] CC3102-T3[20h]		
S-15		CC3201-E13[4h]					

Tabla A.33: Calendarización del Semestre VI de la malla de Ingeniería Civil en Computación durante 2016/2.

	LU	MA	MI	JU	VI	SA	DO	Carga académica semanal
S-1								0 hrs
S-2		4 hrs						4 hrs
S-3		4 hrs						4 hrs
S-4			2 hrs				9 hrs	11 hrs
S-5		8 hrs						8 hrs
S-6		8 hrs	4 hrs			20 hrs		32 hrs
S-7	8 hrs		4 hrs					12 hrs
S-8		4 hrs	8 hrs					12 hrs
S-9		4 hrs	2 hrs				9 hrs	15 hrs
S-10			4 hrs					4 hrs
S-11								0 hrs
S-12				8 hrs	20 hrs			28 hrs
S-13	10 hrs		8 hrs		4 hrs			22 hrs
S-14		9 hrs			32 hrs			41 hrs
S-15		4 hrs						4 hrs

Tabla A.34: Carga académica del Semestre VI de la malla de Ingeniería Civil en Computación durante 2016/2.

Semestre VII

	LU	MA	MI	JU	VI	SA	DO
S-1							
S-2				CC4301-E1[4h]			
S-3			CC4301-E2[4h]				
S-4			CC4301-E3[4h]				
S-5							
S-6			CC4301-C1[12h]	CC4301-T1[10h] CC4101-T1[14h]			
S-7			CC4301-E4[4h] CC4401-P1[15h]				
S-8				CC4401-C1[3h]			
S-9					CC4101-C1[12h]		
S-10			CC4301-C2[12h]	CC4301-E5[4h]			
S-11	CC4101-T2[14h]		CC4401-P2[15h]				
S-12			CC4301-E6[4h]				
S-13							
S-14			CC4301-T2[10h]	CC4401-C2[3h]	CC4101-C2[12h] CC4301-C3[12h]		CC4101-T3[14h]
S-15		CC4401-P3[15h]					

Tabla A.35: Calendarización del Semestre VII de la malla de Ingeniería Civil en Computación durante 2017/1.

	LU	MA	MI	JU	VI	SA	DO	Carga académica semanal
S-1								0 hrs
S-2				4 hrs				4 hrs
S-3			4 hrs					4 hrs
S-4			4 hrs					4 hrs
S-5								0 hrs
S-6			12 hrs	24 hrs				36 hrs
S-7			19 hrs					19 hrs
S-8				3 hrs				3 hrs
S-9					12 hrs			12 hrs
S-10			12 hrs	4 hrs				16 hrs
S-11	14 hrs		15 hrs					29 hrs
S-12			4 hrs					4 hrs
S-13								0 hrs
S-14			10 hrs	3 hrs	24 hrs		14 hrs	51 hrs
S-15		15 hrs						15 hrs

Tabla A.36: Carga académica del Semestre VII de la malla de Ingeniería Civil en Computación durante 2017/1.