UNIVERSIDAD DE BUENOS AIRES

Facultad de Ciencias Exactas y Naturales

Departamento de Matemática

Tesis de Licenciatura

Implementación de una heurística para la programación automática de horarios de una escuela secundaria

Nazareno Ángel Faillace Mullen

Director: Guillermo Durán

Julio de 2018

A la creatividad de mi padre
A la inteligencia de mi madre
A la perseverancia de mi hermano

Agradecimientos

En primer lugar quiero agradecer a mi familia por haberme acompañado y haberme alentado desde el momento que decidí emprender este viaje. Gracias a la sonrisa y al aliento de mi viejo por haber sido el motor que necesitaba en los momentos en los que dudé si esto era lo mío. Gracias a la guía y al impulso que me brindó mi vieja cuando necesitaba tomar decisiones difíciles y jugármela por las oportunidades que se me presentaron. Gracias a la tenacidad y perseverancia de mi hermano, quién no sólo me ayudó con su ejemplo sino que también colaboró con la realización de este trabajo. También agradezco a mis abuelos, a mis tíos y a mis primos por haber estado cuando los necesité.

Quiero agradecerle a Willy por mucho más que haber aceptado ser mi director de tesis. Es un excelente docente que presentó la Investigación Operativa de tal manera que casi inmediatamente decidí que ese era mi lugar dentro de la Matemática. También le agradezco por brindarme, entre otras, la enorme oportunidad de haber asistido a la ELAVIO este año y por la excelente predisposición a dar una mano cada vez que la necesité. También quiero agradecer a Agustín Gravano por la excelente cursada de Introducción a la Computación, que me acercó a esa disciplina de manera tan amena que me dejó con la motivación de querer aprender más. Gracias también a Javier Etcheverry por haberme mostrado un vistazo del mundo de las heurísticas y transmitirme esa inquietud de analizar minuciosamente los resultados y seguir experimentando. Agradezco a Patu y a Daniel por haber aceptado ser los jurados de esta tesis.

Gracias a Fede, Marian, Octi y Vicky, mis amigos de toda la vida, que los admiro, que me acompañan desde siempre, que me impulsaron a seguir adelante, que estuvieron ahí en los momentos más difíciles y de los que siempre tengo cosas que aprender. A Pol y a Juanma, mis amigos del CBC, con quienes empecé este viaje matándome de risa. A Nico y a Flor, con quienes compartí los primeros años de la carrera. A León, Agus, Yami, Nacho, Vale, Rocío y Andrés por haber compartido risas y las cursadas de tantas materias. Y le agradezco a Lucho, no sólo por haber cursado juntos tantas materias, sino también por haber sido un compañero de equipo muy laburante, haberme impulsado a terminar un par de materias optativas y sobre todo por ser un muy buen amigo.

Por último pero no menos importante, quiero agradecer a la educación pública. Me siento agradecido y privilegiado por la calidad de formación que recibí dentro de esta Facultad y por eso agradezco al Departamento de Matemática y al Instituto de Cálculo. Espero poder seguir formando parte del cuerpo docente de la Facultad para poder aportar lo propio, poder devolverle a la sociedad lo que ha invertido en mi formación y me comprometo a seguir luchando para defender la educación pública, tan asediada en estos tiempos.

¡Muchas gracias a todos los que me han acompañado!

Índice general

Agradecimientos III

1. Introducción 1

2. Descripción del problema 3

2.1. Contexto . . . 3

2.2. Caso de estudio . . . 5

2.3. Restricciones . . . 6

3. Modelización del problema 9

3.1. Modelo conceptual . . . 9

3.2. División del problema . . . 12

4. Etapa I: Implementación del Modelo 14

4.1. Estructuras de representación . . . 14

4.2. Restricciones duras suavizadas . . . 18

4.3. $\hat{\Psi}$, Función de penalidad del horario de cursada . . . 20

5. Generación de un horario de cursada inicial 28

5.1. Notación . . . 28

5.2. Modelo de Programación Lineal Entera . . . 30

6. Algoritmos de búsqueda local 36

6.1. Método de *hill-climbing* 38
6.2. Consideraciones previas 40
6.3. *Hill-climbing* simple 43
6.4. *Hill-climbing* con movimientos laterales y búsqueda tabú 48
6.5. *Hill-climbing* híbrido 50
6.6. *Hill-climbing* estocástico 55

7. Etapa II: Implementación del modelo y resolución **60**
7.1. Clases de apoyo: Modelo de Programación Lineal Entera 60
7.2. Proyectos: Modelo de Programación Lineal Entera 64

8. Resultados finales y conclusiones **67**
8.1. Aplicación al horario escolar de 2017 67
8.2. Aplicación al horario escolar de 2018 69
8.3. Conclusiones y trabajo futuro 71

Apéndices:

I. Modelos de PLE **73**

II. Cálculo de $|\mathcal{S}|$ y de $|N(G_{\hat{\eta}})|$ **77**

III.Pseudocódigos **79**

IV.Comparación de horarios de cursada **88**

Bibliografía **97**

Capítulo 1

Introducción

Los problemas de programación de horarios pueden ser descriptos como la tarea de asignar recursos a espacios de tiempo disponibles de manera tal que un conjunto de restricciones sea satisfecha. Además, un grupo de medidas de calidad permite establecer una relación de superioridad o inferioridad, permitiendo así una comparación entre dos horarios. Estos problemas suelen presentarse en diversos contextos. Por ejemplo, en la asignación de enfermeros [11], en la elaboración de fixtures deportivos [14] [22], en la confección de horarios de transporte [15] y en la planificación de horarios de instituciones educativas [8] [16], en el cual se encuentra enmarcado el trabajo de esta tesis.

El problema de la planificación de horarios de universidades y escuelas ha sido estudiado en detalle y la cantidad de trabajos publicados sobre ese tema ha ido aumentado a lo largo de los años. Entre las primeras investigaciones en este campo se encuentran enfoques desde teoría de grafos [3] y desde programación lineal entera (PLE) [4]. Sin embargo, en aquel entonces las técnicas utilizadas eran poco prácticas o demasiado simples como para resolver instancias más complejas o de mayor tamaño. De hecho, el problema de la planificación de horarios para escuelas secundarias fue modelado de manera abstracta como un problema de coloreo de aristas en un grafo bipartito [1] [2], el cual es resoluble en tiempo polinomial. Sin embargo, la adición de restricciones propias de la vida real hacen que el problema pertenezca a la clase $\mathcal{NP}$-completo [25]. Por esta razón, a lo largo del tiempo se ha encarado desde numerosos enfoques. Se han utilizado técnicas basadas en *constraint programming* [5] [6] [23] y más recientemente se han empleado también meta-heurísticas como búsqueda tabú [9], recocido simulado [7] y algoritmos evolutivos [18], entre otros.

En particular, este trabajo estudia una subclase de este tipo de problemas: la elaboración de horarios para escuelas secundarias (conocido en la literatura como *High School Scheduling Problem* o *High School Timetabling Problem*, abreviado como HSSP de aquí en adelante). La naturaleza de HSSP puede ser resumida de la siguiente manera: las clases de los cursos (conjunto de alumnos) deben ser programadas durante el horario laboral de los docentes teniendo en cuenta su disponibilidad y su especialización de manera tal que quede conformado un horario factible y balanceado. Un horario factible es aquél que cumple con una serie de restricciones que permiten que el horario sea aplicable, ya sean restricciones espacio-temporales o disposiciones de las autoridades educativas regionales, entre otros. Por otro lado, como suele ocurrir en los problemas de *scheduling*, la formación del horario afecta a varias partes, cada una con sus propios intereses y peticiones. Por ende, entendemos por balanceado a un horario que satisfaga, en la medida de lo posible, la mayor cantidad de peticiones de cada una de

las partes que, en el marco del HSSP, son los alumnos, los docentes y los directivos. De esta manera, en la confección del horario escolar se tienen en cuenta dos tipos de restricciones: las restricciones duras, que un horario debe cumplir para considerarse factible, y las restricciones blandas, que deben ser satisfechas tanto como sea posible. En general, debido a la complejidad del mundo real, algunas restricciones deben ser relajadas y es muy difícil encontrar una solución que no viole ninguna de ellas. Por esta razón, se utiliza una función de penalidad para evaluar cuán buena es una solución.

El trabajo de esta tesis utiliza como caso de estudio a una escuela pública de la Ciudad de Buenos Aires, que otorgó lo datos necesarios para correr el programa desarrollado así como también excelente predisposición a responder consultas y colaborar bajo la condición de confidencialidad, para proteger la información sensible de los docentes (cargos, horarios en los que trabajan, etc.). El objetivo es implementar un programa que permita confeccionar horarios para escuelas secundarias de la Ciudad de Buenos Aires, aunque podría fácilmente generalizarse a todas las escuelas secundarias de Argentina, teniendo en cuenta las diferencias que podrían existir entre los sistemas educativos de distintas provincias. Se pretende implementar un programa que no sólo otorgue soluciones de buena calidad, sino que lo haga también en poco tiempo. Al tratarse de un problema $\mathcal{NP}$-completo, no hay una manera eficiente de encontrar soluciones óptimas, por lo que las técnicas contemporáneas consisten en aplicar heurísticas a un punto inicial (en nuestro caso, un horario inicial) y desarrollar así una mejor solución. En esta tesis se trabajará principalmente con el algoritmo de búsqueda local *hill climbing* y sus variantes, algunas desarrolladas específicamente para este problema y otras ya existentes en la literatura.

En el Capítulo 2 se describe el problema a resolver, el contexto en el que encuentra y por qué resulta útil la herramienta que brinda este programa al momento de confeccionar un horario. También se presentan las restricciones que guiarán la optimización a lo largo de todo el trabajo.

En el Capítulo 3 se detalla formalmente el problema de confección de horarios y el modelo utilizado. Asimismo, se plantea la división del procedimiento es dos etapas: la elaboración del horario de cursada y la elaboración del horario extracurricular.

En el Capítulo 4 se introducen las estructuras de representación de diferentes elementos del horario de cursada así como también la noción de restricción *dura suavizada* y el concepto de aceptabilidad. En ese capítulo se presentará también una descripción detallada de la función de penalidad que se utiliza para comparar horarios de cursada y que deberá ser minimizada.

El Capítulo 5 consiste en el modelo de PLE que se emplea para generar horarios de cursada iniciales y la notación que se utiliza en el mismo.

En el Capítulo 6 se introducen los algoritmos de búsqueda local y se definen sus ingredientes en el marco del problema a resolver: espacio de búsqueda, noción de vecindad, etc. Luego, se exhiben la implementación, el rendimiento y los resultados de aplicar cuatro variantes de *hill-climbing*: básica, con movimientos laterales y búsqueda tabú, híbrida (desarrollada especialmente para este problema) y estocástica.

En el Capítulo 7 se muestran los modelos de PLE que se emplean para resolver la Etapa II del problema, es decir, la asignación de las horas extracurriculares.

Finalmente, en el Capítulo 8 se exhiben los resultados de haber aplicado el programa a la elaboración de horarios para los años 2017 y 2018. Se comparan con los elaborados manualmente y se resumen las conclusiones y el trabajo futuro.

Capítulo 2

Descripción del problema

2.1. Contexto

Como se ha mencionado anteriormente, la confección de un horario requiere tener en cuenta numerosas restricciones y variables, lo cual hace que la tarea no resulte simple. Actualmente, en la gran mayoría de las escuelas, los horarios son armados manualmente por los directivos o por algún docente que se ofrece como voluntario o voluntaria para la tarea. En algunos casos se utiliza software desarrollado para sistemas educativos de otros países que, si bien otorgan puntos de partida decentes, no terminan de solucionar el problema satisfactoriamente. El armado manual de horarios no es una costumbre exclusiva de nuestro país. Es interesante notar que, a pesar de la gran cantidad de investigación y de los avances en el área de HSSP, las instituciones educativas raramente utilizan herramientas automatizadas para la planificación de horarios. Una encuesta en universidades británicas [10] mostró que sólo el 21 % utiliza una computadora en la programación de horarios de exámenes, el 37 % utiliza una computadora como asistencia en el proceso, mientras que el 42 % no utiliza una computadora en ningún momento. La confección de horarios en Japón puede tomar hasta 100 horas hombre e incluso más en escuelas grandes.

Sin embargo, muchos docentes y directivos de distintas escuelas, tanto públicas como privadas, con los que se ha comentado la idea de esta tesis concuerdan con entusiasmo que una herramienta que permita programar el horario escolar automáticamente sería más que bienvenida. La principal razón es que la tarea de armar el horario suele resultar compleja, tediosa y, como en los casos antes mencionados, suele consumir mucho tiempo. Otra dificultad del armado manual se presenta al momento de modificar un horario ya establecido (por ejemplo, debido al cambio en la disponibilidad de un docente, la renuncia a un cargo, etc.) efectuando la menor cantidad de alteraciones posibles. Por último, como ocurre en otros problemas de *scheduling*, cuando el armado de un horario está en manos de una persona, se corre el riesgo de incurrir en favoritismo y priorizar el beneficio de aquellos que son más allegados.

Otro factor a tener en cuenta es la situación actual de cambio en las escuelas secundarias en Argentina y, en particular, en la Ciudad de Buenos Aires. Desde 2009 el Consejo Federal de Educación, presidido por el Ministerio de Educación de la Nación, está llevando adelante a nivel nacional el proyecto Nueva Escuela Secundaria (NES) [28] con el objetivo de "*revisar, actualizar y mejorar las estructuras y los procesos educativos que caracterizan a la escuela actual*". Más aún, con este proyecto se busca lograr

una secundaria común para todas las provincias, por lo que la utilidad del programa que se pretende desarrollar en esta tesis no estaría limitada a la Ciudad de Buenos Aires. Las reformas impulsadas por la NES aspiran a transformar a la escuela secundaria en la Escuela Secundaria Orientada. Los cinco años de la secundaria se dividen en dos ciclos: un ciclo de formación general común a las escuelas de todas las orientaciones (1^{ero} y 2^{do} año) y un ciclo de formación orientada (3^{ero} a 5^{to} año). En 2009 se establecieron 10 orientaciones a nivel nacional y en 2013, debido a la diversidad en la oferta educativa de la Ciudad de Buenos Aires, se aprobaron tres nuevas orientaciones en dicha jurisdicción. La NES comenzó a implementarse en escuelas pioneras en 2014 y en 2015 se sumaron 440 escuelas, tanto estatales como privadas. Las reformas se implementan año a año. Es decir, aquéllos alumnos que comenzaron el secundario antes del 2014/2015 cursarán el plan tradicional de sus respectivas instituciones hasta que egresen; recién los estudiantes que ingresaron en 2014/2015 experimentarán los cambios dispuestos por la NES.

Más recientemente, en el marco de la implementación de los cambios dispuestos por la NES, se presenta una propuesta por parte del Gobierno de la Ciudad de Buenos Aires que plantea la profundización de dicho proyecto, denominada la Escuela que Queremos [27]. Esta propuesta busca favorecer la introducción de formas de agrupamiento de estudiantes en proyectos, debates, articulación de materias y actividades en las que participen alumnos de diferentes años. Al mismo tiempo, la propuesta aspira a implementar nuevos métodos como, por ejemplo, las parejas pedagógicas, que consiste en que dos profesores den clase de manera conjunta, articulando los conocimientos de ambas materias. También se le otorga mayor prioridad a los proyectos extraescolares para los alumnos y al apoyo escolar.

Naturalmente, la puesta en marcha de todos estos cambios presentan nuevos desafíos en la confección de los horarios de las escuelas secundarias. En primer lugar, el establecimiento de una orientación en una escuela implica que se deben impartir nuevas materias. Esto significa que los cargos para cubrir esas asignaturas deben ser presentados en concurso y que, si la carga horaria semanal de esa nueva materia es pequeña, se desea que esté dispuesta de la mejor manera posible (horario concentrado) de manera tal que resulte atractiva para el docente que concursa. Otra dificultad que se presenta es ubicar los proyectos y las clases de apoyo a contraturno de manera tal que resulten viables para los alumnos: se prefiere que los proyectos y clases de apoyo para el turno mañana estén ubicados durante las primeras horas de la tarde, puesto que no es realista pensar que un alumno que sale a las $12:50$ de la escuela esperará hasta las 17 para asistir a una clase de apoyo. Además, puesto que los alumnos tienen Educación Física también en contraturno, se desea que las horas de apoyo escolar estén distribuidas a lo largo de la semana uniformemente, de manera tal que todos tengan la oportunidad de asistir a al menos una de ellas.

La cuestión espacial y el uso de recursos es otra arista del desafío que supone la creación de las orientaciones, puesto que ciertas materias requieren equipamiento especial. Por ejemplo, si una escuela ofrece orientación en Informática, se deben administrar cuidadosamente las salas de computación para que no se superponga su uso, más aún teniendo en cuenta que no toda escuela dispone de una gran cantidad de recursos edilicios para satisfacer la demanda de la o las orientaciones que ofrece. Por otro lado, la implementación de las parejas pedagógicas complica la tarea de confección de horarios, puesto que no sólo hay que tener en cuenta la disponibilidad y la disposición del horario de un docente, sino de dos simultáneamente.

Es claro que la puesta en marcha de las medidas propuestas por la NES y la Escuela que Queremos tienen muchos retos más allá de la confección de los horarios, como desafíos pedagógicos, sociales, económicos, presupuestarios, etc. Pero lo que distingue al problema de elaboración de horarios del resto, además de ser el único sobre el cual el autor de esta tesis tiene conocimiento suficiente como para aspirar a encontrar una solución, es que podría resolverse con un programa adecuado.

2.2. Caso de estudio

Como se ha mencionado en la introducción, se ha tomado como caso de estudio a una escuela pública de la Ciudad de Buenos Aires. Se cuenta con los datos utilizados para elaborar los horarios que rigieron durante el ciclo lectivo de 2017 y de 2018; se trabajará con el primero a lo largo de la tesis, a modo de caso de prueba.

El turno mañana de esta escuela está compuesto por doce cursos y el turno tarde por diez, sumando un total de veintidós cursos. La escuela cuenta con la existencia de prehora para el turno tarde: el curso que tiene prehora un determinado día ingresa en el horario correspondiente a la séptima hora del turno mañana. Es decir, temporalmente, la séptima hora del turno mañana se desarrolla en el mismo momento que la prehora del turno tarde. Por esta razón, dado que hay docentes que trabajan en ambos turnos, se debe considerar que un docente no puede dar clase en la séptima hora del turno mañana y en la prehora del turno tarde un mismo día.

También, cada turno tiene actividades en contraturno, sean clases de Educación Física, apoyo escolar o proyectos. La asistencia a los dos últimos es opcional. En lo que respecta a Educación Física, sus horarios son confeccionados por sus profesores, y suelen presentarse en Abril, a comparación del horario escolar que, preferentemente, debe estar listo durante los primeros días de Febrero. Por esa razón, el programa no contempla la confección de los horarios de esa asignatura.

Debido a que los docentes y directivos ocasionalmente deben reunirse en un Taller Docente los Lunes a las $17:20$ horas, esos días no hay séptima hora para el turno tarde (por eso se distribuyen las horas de las materias que deberían ser dictadas en ese espacio de tiempo entre las prehoras).

En ciertos cursos, a principio de año, cada alumno puede optar qué materia cursar entre dos opciones. Durante esas horas cátedra, el curso estará dividido en dos, cada parte asistiendo a la materia que eligió. Por otro lado, por disposición del Gobierno, ciertas materias que permiten elecciones se deben dar simultáneamente en dos cursos. Por ejemplo, si los cursos $C1$ y $C2$ pueden elegir entre las materias $M3$ y $M7$, si así es dispuesto por el Gobierno, se agrupa, durante esas horas cátedra, a los alumnos de $C1$ y de $C2$ que hayan elegido $M3$ y a los que eligieron $M7$ por el otro; cada grupo cursa las horas de esa materia y luego regresan a sus cursos respectivos. Se referirá a esta última situación como un caso de 'simultaneidades'.

En el caso de las escuelas secundarias argentinas, los alumnos son agrupados en cursos (en general no tienen libertad para elegir qué materias cursar) y a cada curso es asignada un aula, generalmente teniendo en cuenta la cantidad de estudiantes que lo conforman. Como la asignación de aulas se mantiene año a año y los alumnos permanecen en las aulas durante todas las horas de clase (salvo para ciertas materias), la asignación de aulas no forma parte del problema del armado de horarios. Dicho en otras palabras, los alumnos no se mueven, mientras que los docentes sí deben hacerlo. En cierta manera, el hecho de que se pueda tratar al conjunto total de alumnos como una unión disjunta de subconjuntos (cursos) simplifica la modelización del problema.

Suele ocurrir que la mayoría de los docentes trabajan en más de una escuela. Por esa razón, al momento de elaborar el horario escolar, se debe tener en cuenta la disponibilidad de cada uno. A fin de año cada docente debe presentar una declaración jurada donde indica qué horas tiene ocupadas en otras instituciones educativas, inhabilitándolo a dar clases en esos horarios.

Finalmente, en general, la distribución de docentes en los cursos no se modifica de año a año. Es decir, si a un docente enseña Matemática en primer año, el año que viene también enseñará Matemática

en primer año. Por lo tanto, la asignación de profesores a cursos tampoco forma parte del problema que hay que resolver.

2.3. Restricciones

Recordemos que en los problemas de *scheduling* se presentan dos tipos de restricciones: duras y blandas. Una solución del problema debe satisfacer todas las restricciones duras para poder considerarse factible, mientras que la cantidad de violaciones de las restricciones blandas determinan su calidad a través de una función de costo o función de penalidad. Un horario con poca penalidad es visto como un buen horario. Qué restricciones se consideran duras o blandas es una decisión que, por un lado, depende del modelado del problema y, por el otro, depende del contexto del mismo.

No sólo hace falta distinguir la naturaleza de las restricciones, sino también establecer un orden de prioridades para las restricciones blandas, dado que esto es esencial para construir una función de penalidad que refleje tan fielmente como sea posible lo que se considera un buen horario. Persiguiendo este objetivo, se llevaron a cabo reuniones con los directivos de la escuela y con la docente que había asumido la tarea de armar el horario escolar cada año. Denominaremos al conjunto de restricciones para el armado del horario escolar como $\mathcal{R}_1$ y se clasifican de la siguiente manera:

- Restricciones duras:
    1. *Espacio y tiempo*: un docente no puede estar en dos lugares al mismo tiempo. Es decir, dada una hora, el profesor puede estar en, a lo sumo, un curso, a excepción de las horas en las que dictan las materias que deben impartirse simultáneamente en varios cursos. Por otro lado, un curso no puede estar cursando dos materias al mismo tiempo, salvo en los casos de las simultaneidades.
    2. *Disponibilidad docente*: no se puede asignar una hora de clase a un docente que no esté disponible en ese momento.
    3. *Simultaneidades*: dado que es una disposición gubernamental, ciertas materias deben ser dictadas al mismo tiempo en dos cursos distintos.
    4. *Séptima hora del turno tarde del Lunes*: no pueden programarse clases para ese momento pues se lleva a cabo el Taller Docente, al cual deben asistir tanto los directivos como los profesores.
    5. *Cantidad de horas diarias de cursada*: el turno tarde puede tener entre 6 y 8 horas de clase diarias, mientras que el turno mañana debe tener exactamente 7 horas diarias (pues no hay prehoras en el turno mañana)
    6. *Horas diarias de una materia*: no pueden haber más de 3 horas diarias de una materia (preferentemente no deben haber más de 2, pero 3 es tolerable)
    7. *Carga horaria semanal*: se debe satisfacer la carga horaria semanal de cada materia en cada curso
    8. *Sin horas libres para los alumnos*: los alumnos deben tener asignada una materia en cada hora cátedra. Por ejemplo, no puede ocurrir que durante la tercera hora un curso no tenga asignada ninguna materia.
    9. *Cargo del docente*: cada docente debe tener tantas horas de clase y de extraclase como lo determine su cargo.

- Restricciones blandas (en orden descendente de prioridad):

    10. *Sin huecos en el dictado de materias*: dada una materia que se dicta en un día durante dos o más horas, todas las horas deben estar juntas. Por ejemplo, si el miércoles se dictan dos horas de Matemática en tercer año, se desea que ambas horas estén agrupadas; no que se dicte en la primera hora y luego en la quinta.
    11. *Cantidad de bloques*: se entiende por bloque a un conjunto de dos o más horas consecutivas en las que se imparte la misma materia. Por ejemplo, Matemática tiene un bloque en cierto curso un cierto día si se dicta durante la primera y la segunda hora. Se desea que las materias conformen la mayor cantidad de bloques de dos horas posible. Es decir, que si la carga horaria semanal de una materia M en el curso C es t_{MC}, con $t_{MC} \geq 2$, lo ideal es que se conformen $\left\lfloor \frac{t_{MC}}{2} \right\rfloor$ bloques durante la semana.
    12. *Triples*: mientras que es posible dictar tres horas de una materia en un mismo día, se prefiere que esto ocurra la menor cantidad de veces posible.
    13. *Horas inactivas de los docentes*: es preferible que los docentes tengan la menor cantidad de horas inactivas durante su horario laboral. Como el trabajo del docente es remunerado según el cargo, una hora inactiva es tiempo perdido para el docente.
    14. *Recursos*: la escuela cuenta con un conjunto de recursos y cada uno cuenta con una demanda distinta. Además, ciertos recursos resultan casi indispensables para dar clase (por ejemplo, la sala de computación) mientras que otros no (por ejemplo, el laboratorio).
    15. *Cantidad de prehoras*: se desea que, por una cuestión espacial, no hayan más de 6 prehoras del turno tarde por día. Según los directivos, hasta esa cantidad, se pueden organizar las clases en aulas vacías, SUM, biblioteca, etc.
    16. *Recreos*: es preferente que los recreos no interrumpan los bloques de las materias. En otras palabras, que antes y después de un recreo se dicten materias distintas.

Para la organización de los horas extraclase (los proyectos, las clases de apoyo y otras tareas de los docentes donde no participan los alumnos) también se tiene en cuenta una serie de restricciones. En este caso, ya no se tiene la noción de curso, pues cualquier alumno puede asistir a las clases de apoyo y a los proyectos, independientemente del año al que pertenezca. Recordar que ambos son optativos y se cursan a contraturno (el turno mañana los cursa a la tarde y viceversa). Se denomina $\mathcal{R}_2$ al conjunto de restricciones que guían la programación de las horas extraclase y se clasifican de las siguiente manera:

- Restricciones duras:

    1. *Espacio y tiempo*: en este caso esta restricción se limita a que un docente no puede llevar cabo dos actividades al mismo tiempo.
    2. *Disponibilidad docente*: ídem $\mathcal{R}_1$
    3. *Séptima hora del turno tarde del Lunes*: ídem $\mathcal{R}_1$
    4. *Contraturno*: las horas extraclase del turno mañana deben ser programadas durante las horas del turno tarde (exceptuando a la prehora, que coincide con la séptima del turno mañana) y lo análogo para el turno tarde.

- Restricciones blandas:

5. *Conveniencia para los alumnos*: se desea ubicar las horas extraclase tan cerca del horario de salida (entrada) del turno mañana (tarde) como sea posible. Vale aclarar que se le otorga mayor prioridad a la ubicación de las clases de apoyo que a los proyectos, debido a que las primeras son una herramienta a disposición del alumno para permitirle mejorar su rendimiento escolar.
6. *Distribución uniforme de las clases de apoyo*: recordar que las clases de Educación Física también se dictan a contraturno, dos veces por semana, y se divide a los alumnos según el sexo, el año y el deporte, por lo que distribuir equitativamente las horas de apoyo de cada materia a lo largo de la semana garantiza que la mayor cantidad posible de alumnos tengan acceso a ellas.

En el capítulo siguiente se presenta el modelo del problema, así como también definiciones formales de ciertos conceptos a los que se hará referencia durante el resto del trabajo.

Capítulo 3

Modelización del problema

3.1. Modelo conceptual

El HSSP se caracteriza por la sencillez con la que se puede entender y por la dificultad de definirlo en general. Cada país, e incluso cada jurisdicción dentro de un mismo país, tiene una manera propia de estructurar la escuela media y por esa razón muchos de los trabajos realizados en el área de HSSP emplean heurísticas y meta-heurísticas diseñadas a medida del sistema que se está estudiando [21]. Sin embargo también existen trabajos que se enfocan en formular modelos genéricos que podrían ser utilizados internacionalmente [26].

En este modelo se asume que la designación de aulas y la distribución de los profesores entre los cursos ya han sido efectuadas, por lo que el modelo se utilizará para determinar qué materia se asigna a cada hora de clase. Asimismo, se asume que los alumnos se encuentran disponibles durante todo el horario escolar y que el mismo abarca una semana (es decir, si una clase toma lugar durante la tercera hora del Lunes, siempre toma lugar durante la tercera hora del Lunes).

La siguiente definición formal del HSSP está basada en la presentada en [26] y adaptada al caso que se desea estudiar en este trabajo. Debajo se presentan las definiciones de multiconjunto, el cual será utilizado para definir formalmente el HSSP, y la definición de curso, que no es indispensable para formular el problema formalmente pero se considera oportuno introducirla ahora junto con su notación pues será utilizada reiteradas veces a lo largo del trabajo.

Definición 3.1. (Multiconjunto) Un multiconjunto es formalmente definido como una 2-tupla (A, m) donde A es el *conjunto subyacente* del multiconjunto, formado por sus distintos elementos, y $m : A \to \mathbb{N}_{\geq 1}$ es una función que determina la multiplicidad de cada elemento de A, es decir, $m(a)$ determina cuántas veces aparece a en el multiconjunto. El multiconjunto suele ser notado con [ ].

En otras palabras, a diferencia de un conjunto, en un multiconjunto se pueden hallar los mismos elementos repetidas veces. Lo que diferencia a un multiconjunto de una tupla es que el orden de los elementos no tiene importancia. Así, por ejemplo, $[a, a, b]$ y $[a, b, a]$ denotan el mismo multiconjunto. Al número de veces que aparece cierto elemento en un multiconjunto se lo denomina *multiplicidad del elemento en el multiconjunto.*

Observación. Un (sub)conjunto basado en un multiconjunto es otra vez un multiconjunto.

Definición 3.2. (Curso) Un curso es un conjunto de alumnos que pertenecen al mismo año escolar y comparten la misma programación de horarios a lo largo del año. Los alumnos de un curso no comparten clases con alumnos de otro curso, salvo por las materias que deben ser cursadas simultáneamente por disposición del gobierno.

Generalmente cada año escolar está, a su vez, separado en divisiones. Como los alumnos de diferentes divisiones tienen distintos horarios y tampoco cursan juntos, se considera a cada división como un curso. Por ejemplo, si cada año, de primero a quinto, está separado en dos divisiones, se tendrán 10 cursos en total. Observar entonces que si $\mathcal{A}$ es el conjunto de todos los alumnos del colegio, cada curso es un subconjunto de $\mathcal{A}$. Luego, si se tienen N cursos, $\mathcal{A} = \bigcup_{j}^{N} C_j$ con $C_j \cap C_i = \emptyset \quad \forall j \neq i$.

Definición 3.3. (Simultaneidad) Por disposición del gobierno, puede ocurrir que cierta materia m deba darse en una unión de algunos cursos $\mathcal{C}$ al mismo tiempo, puesto que los alumnos de los cursos involucrados se mezclan durante las horas cátedras asignadas a m y el grupo de profesores que imparten esa materia es común a todos los cursos de $\mathcal{C}$. Decimos entonces que m es una simultaneidad en $\mathcal{C}$.

A continuación se definen los conjuntos interrelacionados en los que se basa la modelización formal de nuestro problema y sus respectivas notaciones:

$\mathcal{A}$	Conjunto de todos los alumnos de la escuela
C_i	Curso i
$\mathcal{M}$	Conjunto de materias. Para la definición formal del modelo, cada materia está ligada al año en la que se imparte. Por ejemplo, en esta instancia, se considera que Matemática de primer año y Matemática de segundo año son dos materias diferentes.
$\mathcal{P}$	Conjunto de profesores
$\mathcal{H}$	Conjunto de horas cátedra
$\mathcal{S}$	Conjunto de simultaneidades
$\mathcal{G}$	Conjunto de grupos de asignatura, los cuales están conformados por una terna de una materia, un conjunto de alumnos y al menos un docente. De esta manera, $\mathcal{G} = \{(A, m, P) \,/\, A \subseteq \mathcal{A},\, m \in \mathcal{M},\, P \subseteq \mathcal{P}, P \neq \emptyset\}$ [1]
$\mathcal{L}$	Conjunto de clases, que es un multiconjunto de grupos de asignatura. La multiplicidad de un grupo de asignatura $g \in \mathcal{G}$ está dada por la carga horaria semanal.

Cuadro 3.1: Definición de los conjuntos empleados

Definición 3.4. (Horario) Se define como *horario* al multiconjunto de clases $\mathcal{L}$ mediante el cual a cada clase $l \in \mathcal{L}$ se le asigna una hora $h \in \mathcal{H}$. En otras palabras, el horario se crea tomando a las clases de $\mathcal{L}$ y asignándoles una hora cátedra.

Ejemplo 3.1. Sea el conjunto $\mathcal{G} = \{g_1, g_2, g_3\}$ donde:

$$g_1 = (C_1, M_3, P_1) \quad g_2 = (C_1, M_4, P_5) \quad g_3 = (C_1, M_7, P_3)$$

[1] Notar que no se requiere que $A \neq \emptyset$, abarcando las tareas de los docentes que no involucran alumnos.

con C_1 un curso, $M_3, M_4, M_7 \in \mathcal{M}$, $P_1 \subseteq \mathcal{P}$, $P_5 \subseteq \mathcal{P}$ y $P_3 \subseteq \mathcal{P}$. Es decir, el grupo de asignatura g_1 representa el siguiente escenario: la materia $M3$ es dictada por los profesores pertenecientes a P_1 al curso 1. Se podría tener que:

$$\mathcal{L} = [g_1, g_1, g_2, g_2, g_3, g_3, g_3, g_3]$$

Recordar que la multiplicidad cada elemento en $\mathcal{L}$ representa la carga horaria semanal del grupo de asignatura. En este caso, como g_1 tiene multiplicidad 2 en $\mathcal{L}$, esto significa que la materia $M3$ es dictada por los profesores de P_1 en el curso 1 dos horas a la semana. A partir de $\mathcal{L}$, un posible horario sería:

$$\eta = [(g_1, a), (g_1, d), (g_2, a), (g_2, c), (g_3, a), (g_3, b), (g_3, c), (g_3, d)]$$

Notar que la Definición 3.4 no impone ninguna restricción sobre la asignación de horas cátedra a las clases. Teóricamente podría ocurrir que un mismo curso tenga asignada dos materias distintas en la misma hora, aunque las restricciones que se impondrán luego no permitirán que esto suceda.

Generalmente no se trabaja con el horario η completo, sino que en ciertas ocasiones resulta más útil analizar una porción del mismo. Eso inspira la definición de *horario reducido*.

Definición 3.5. (Horario reducido) Un horario reducido es un subconjunto del horario η

Ejemplo 3.2. Un horario reducido podría utilizarse para conocer el horario de sujetos específicos, como profesores, cursos o materias. De esta manera, el horario de la profesora p_1 sería el horario reducido definido como $[((A, m, P), h) \in \eta \ / \ p_1 \in P]$. Analizar ese horario reducido nos permitiría, por ejemplo, contar cuántas horas inactivas tiene la profesora p_1.

Asimismo, se puede encontrar el horario específico para cada una de las otras entidades que conforman al horario.

Definición 3.6. Sea $b \in B$, donde $B \in \{\mathcal{A}, \mathcal{M}, \mathcal{P}, \mathcal{G}, \mathcal{H}\}$, se define como $\eta_B(b)$ al horario restringido de η donde las clases contienen al elemento $b \in B$

Ejemplo 3.3. Para $p \in \mathcal{P}$, $\eta_{\mathcal{P}}(p)$ es el horario reducido de η que contiene a las clases impartidas por el docente p.

Como se ha visto anteriormente, todo horario está sujeto a un conjunto de restricciones; entre ellas se encuentran las restricciones duras. Si un horario viola alguna de estas últimas, pierde la validez. De esta manera, surge la necesidad de definir una función que chequee la validez de un horario (restringido).

Definición 3.7. (Función de validez) Sea $\texttt{VALIDO} : \mathbb{P}(\mathcal{L} \times \mathcal{H}) \to \{0, 1\}$. $\texttt{VALIDO}$ es la función de validez de un horario (restringido) $\eta \in \mathbb{P}(\mathcal{L} \times \mathcal{H})$. Si η no viola ninguna restricción dura, $\texttt{VALIDO}(\eta) = 1$ (y por ende η resulta un horario (restringido) válido); de lo contrario, $\texttt{VALIDO}(\eta) = 0$.

Si bien es cierto que las restricciones blandas no cambian la validez de un horario, el grado de cumplimiento de las mismas determinan si uno es mejor o peor que otro. Al mismo tiempo, las preferencias representadas por las restricciones blandas tienen distintos niveles de prioridad. Por estas razones, es necesario definir una función de penalidad que permita cuantificar la calidad de un horario válido basándose en el conjunto $\mathcal{R}_S$ de las restricciones blandas.

Definición 3.8. (Función de penalidad) $\Psi : \mathbb{P}(\mathcal{L} \times \mathcal{H}) \times \mathbb{P}(\mathcal{R}_S) \to \mathbb{R}_{\geq 0}$ es la función de penalidad de un horario $\eta \in \mathbb{P}(\mathcal{L} \times \mathcal{H})$ sujeto a un conjunto de restricciones blandas $R \in \mathcal{R}_S$.

La implementación particular de esta función depende de las restricciones blandas contempladas, sus prioridades y los algoritmos utilizados. Generalmente, se define la penalidad para cada restricción violada y se toma la suma de cada una de esas penalidades. Luego, el horario con la menor penalidad global es elegido como el mejor horario. La descripción detallada de la función de penalidad utilizada en este trabajo se encuentra en la sección 4.3.

Con las definiciones y notaciones presentadas en esta sección, estamos en condiciones de definir el problema que queremos resolver como un problema de minimización:

$$\begin{array}{ll} \text{Dados} & \mathcal{L}, \mathcal{R}_S, \mathcal{H} \\ \text{minimizar} & \Psi(\eta, \mathcal{R}_S) \\ \text{sujeto a} & \texttt{VALIDO}(\eta) = 1, \\ & (l, h_l) \in \eta \qquad l \in \mathcal{L}, h \in \mathcal{H} \end{array}$$

Según esta definición, el cumplimiento de las restricciones blandas está contemplado en la minimización de $\Psi(\eta, \mathcal{R}_S)$, mientras que el de las restricciones duras viene dado por el requerimiento de que $\texttt{VALIDO}(\eta) = 1$. La segunda condición del problema de minimización está relacionada con la Definición 3.4 y establece el requerimiento de que toda clase de $\mathcal{L}$ sea incluida en el horario.

3.2. División del problema

El modelo descripto en la sección anterior es muy general y no contiene los detalles del problema que se desea estudiar en este trabajo. Por ese motivo, en esta sección se imponen más condiciones al HSSP, delimitándolo mejor y haciéndolo más específico. Sin embargo, antes de abocarse a ello, se presenta brevemente el plan de resolución del problema, el cual involucra una división del mismo en dos: confeccionar el horario de cursada y confeccionar el horario de extraclases. El primero, por la cantidad de restricciones y por su tamaño, representa mayor dificultad de resolución. Por esa razón, la mayoría del trabajo se enfoca en él y el siguiente capítulo se dedicará a detallarlo y a definirlo formalmente.

La confección del horario para una escuela de la Ciudad de Buenos Aires implica la programación tanto de las clases propiamente dichas (al que denominaremos de aquí en más *horario de cursada*) como de las extraclases (al que llamaremos *horario extracurricular*). Según los directivos y la docente encargada de la elaboración de los horarios, una práctica común es primero armar el horario de cursada y luego adaptar las extraclases al mismo. En otras palabras, el armado del horario de cursada tiene mayor prioridad que el extracurricular. Por tanto, se decidió que el programa resolvería el problema de la asignación de horarios siguiendo esa misma filosofía: considerar el horario escolar separado en horario de cursada y en horario extracurricular. La resolución del problema de confeccionar un horario escolar queda entonces dividida en dos etapas:

- Etapa I: encontrar un horario de cursada de buena calidad
- Etapa II: dado un horario de cursada obtenido en la Etapa I, asignar de manera óptima las horas de extraclase

Resolver el problema dividido en partes no garantiza una solución de igual calidad que la resolución del problema entero [20]. Sin embargo, en la programación de horarios, el objetivo principal suele ser la factibilidad y aceptabilidad, no la optimalidad [19] [13], debido a la numerosa cantidad de restricciones

del problema. Además, encontrar una buena solución en un periodo corto de tiempo también suele ser considerado más valioso que encontrar una solución óptima luego de un largo periodo de optimización [24].

Como se puede apreciar en la sección 2.3, la menor cantidad de restricciones y el hecho que cualquier alumno puede asistir a cualquier clase de apoyo o proyecto independientemente del curso al que pertenezca, hacen que el problema de asignar las horas extraclase tenga un tamaño mucho menor y que optimizarlo resulte más sencillo. En efecto, como se verá en el Capítulo 7, se puede resolver con un modelo de programación entera en muy poco tiempo. A continuación, retomando la notación presentada en la sección 3.1, se introduce la definición de multiconjunto de clases de cursada y multiconjunto de clases extracurriculares, que serán útiles para definir formalmente los conceptos de horario de cursada y horario extracurricular:

Definición 3.9. (Multiconjunto de clases de cursada y de clases extracurriculares) Sea $\mathcal{L}$ el multiconjunto de clases a programar, se definen $\mathcal{L}_C$ multiconjunto de clases de cursada y $\mathcal{L}_E$ multiconjunto de clases extracurriculares, respectivamente, como:

$$\mathcal{L}_C = [(A, m, P) \in \mathcal{L} \,:\, A \in \{C_i\}_{j=1}^{N}]$$

$$\mathcal{L}_E = [(A, m, P) \in \mathcal{L} \,:\, A = \mathcal{A} \vee A = \emptyset]$$

Definición 3.10. (Horario de cursada y horario extracurricular) Sea η un horario, se definen $\hat{\eta}$ su horario de cursada y $\breve{\eta}$ su horario extracurricular, respectivamente, como:

$$\hat{\eta} = [(l, h) \in \eta \,:\, l \in \mathcal{L}_C]$$

$$\breve{\eta} = [(l, h) \in \eta \,:\, l \in \mathcal{L}_E]$$

Recordar que en nuestro caso de estudio los estudiantes conforman grupos que son disjuntos entre sí, a los que hemos denominado *cursos*. Durante las horas de clase, cada alumno cursa una materia con su respectivo curso (o parte del mismo). Con respecto a las simultaneidades, aunque en la práctica se mezclen los alumnos de los cursos involucrados, se modela como que esas materias deben ser impartidas en esos cursos durante las mismas horas cátedras, puesto que el movimiento de los alumnos es irrelevante. Esto implica que los profesores que enseñen las materias de las simultaneidades serán excepciones a una de las restricciones duras del modelo: *cada profesor puede estar enseñando sólo a un curso por hora cátedra*. Se permite entonces que los docentes que están a cargo de las simultaneidades, durante las horas cátedras asignadas a ellas, estén en más de un curso. Por otro lado, teníamos el caso de las horas de extraclase, a las que los alumnos pueden asistir opcionalmente o a las que no asiste ningún alumno (actividades propias de los profesores). Por todo esto, de acuerdo con la notación introducida en la sección 3.1, en nuestro modelo se tiene que un conjunto de alumnos es un curso, es la totalidad de los alumnos (el caso de las clases de apoyo y proyectos) o es vacío. Es decir, se tiene que:

$$A \subseteq \mathcal{A} \Rightarrow A \in \{C_j\}_{j=0}^{N-1} \cup \emptyset \cup \mathcal{A} \quad \text{donde } N \text{ es la cantidad de cursos}$$

De acuerdo a las definiciones 3.9 y 3.10, se tiene entonces que $\mathcal{L} = \mathcal{L}_c \cup \mathcal{L}_E$ y, en consecuencia, que $\eta = \hat{\eta} \cup \breve{\eta}$. Luego, queda demostrado que las dos etapas antes propuestas resuelven la totalidad del problema de la confección de horarios. Los Capítulos 4 a 6 se enfocarán en modelar y resolver la Etapa I, mientras que el Capítulos 7 se dedica a modelar y resolver la Etapa II.

Capítulo 4

Etapa I: Implementación del Modelo

Puesto que la mayor dificultad en la resolución del problema que nos compete se presenta en el armado del horario de cursada, la mayoría del trabajo se enfocará en encontrar un método que encuentre en poco tiempo buenas soluciones de la Etapa I. El problema de minimización en el que se hará hincapié durante esta etapa queda formulado entonces como:

$$\begin{array}{ll} \text{Dados} & \mathcal{L}_C, \mathcal{R}_{1S}, \mathcal{H} \\ \text{minimizar} & \hat{\Psi}(\hat{\eta}, \mathcal{R}_{1S}) \\ \text{sujeto a} & \texttt{VALIDO}(\hat{\eta}) = 1, \\ & (l, h_l) \in \hat{\eta} \qquad l \in \mathcal{L}_C \, , \, h \in \mathcal{H} \end{array}$$

Donde $\mathcal{R}_{1S}$ es el conjunto de restricciones blandas, presentado en la sección 2.3, para el armado del horario de cursada, $\hat{\Psi}$ es la función de penalidad de $\hat{\eta}$ y consideramos `VALIDO` en este contexto como la función de validez de $\hat{\eta}$. La función de penalidad $\hat{\Psi}$ viene dada como la suma de funciones que miden cuánto se ha violado cada una de las restricciones blandas, se describirá con más detalle en la sección 4.3.

4.1. Estructuras de representación

Generalmente en las escuelas secundarias se representa el horario escolar con dos instrumentos: una grilla para cada turno que contiene el horario de cursada semanal de cada uno de sus respectivos cursos y una ficha de cada docente que detalla su horario semanal, a la que sólo tiene acceso el docente y los directivos (ver Figura 4.1 y Figura 4.2, respectivamente). Puesto que resultan formas que describen concisamente las distintas características de un horario, se decidió adaptarlas a la implementación del programa. Es importante notar cómo se traducen al modelo formal: cada columna de la grilla representa el horario reducido de cada curso ($\eta_{\mathcal{A}}(C)$) y cada ficha representa el horario reducido de cada docente ($\eta_{\mathcal{P}}(p)$).

Dia	Hora	CURSO 0	CURSO 1	CURSO 2	CURSO 3	CURSO 4	CURSO 5	CURSO 6	CURSO 7	CURSO 8	CURSO 9	CURSO 10	CURSO 11
LUNES	PH												
	1	M1	M0	M2	M22	M24	M2	M12	M22	M14	M0	M9	M29
	2	M1	M0	M2	M22	M24	M2	M12	M22	M14	M0	M9	M29
	3	M11	M2	M6	M0	M22	M22	M26	M13	M28	M4	M6	M0
	4	M11	M2	M6	M1	M22	M22	M26	M13	M28	M4	M6	M0
	5	M0	M1	M11	M2	M2	M6	M22	M0	M4	M29	M28	M18
	6	M3	M1	M11	M10	M2	M6	M22	M12	M4	M14	M28	M18
	7	M3	M11	M22	M10	M23	M1	M20	M12	M1	M14	M1	M28
MARTES	PH												
	1	M10	M6	M24	M0	M0	M22	M12	M15	M0	M4	M18	M1
	2	M10	M6	M24	M0	M0	M22	M12	M15	M0	M4	M18	M1
	3	M0	M24	M1	M2	M8	M2	M0	M12	M4	M1	M17	M9
	4	M0	M24	M23	M2	M8	M2	M0	M12	M4	M1	M17	M9
	5	M24	M2	M22	M3	M2	M1	M26	M20	M19	M28	M28	M0
	6	M22	M1	M2	M3	M2	M24	M26	M20	M19	M28	M29	M17
	7	M22	M0	M2	M23	M1	M24	M22	M13	M1	M29	M29	M17
MIERCOLE	PH												
	1	M2	M0	M7	M6	M22	M25	M25	M11	M27	M16	M17	M5
	2	M2	M0	M7	M6	M22	M25	M25	M11	M27	M16	M17	M5
	3	M0	M11	M25	M11	M6	M11	M7	M7	M0	M27	M28	M5
	4	M0	M11	M25	M11	M6	M0	M7	M7	M29	M27	M0	M28
	5	M22	M25	M11	M2	M11	M0	M11	M25	M29	M0	M5	M28
	6	M11	M25	M0	M0	M11	M7	M11	M25	M28	M21	M5	M6

Figura 4.1: Parte de la grilla del turno mañana utilizada por la escuela durante el 2017. Se han reemplazado el nombre de cada materia por su respectivo código de identificación.

Como resulta natural, cada grilla se representa con una matriz[1] cuyo número de columnas es la cantidad de cursos del turno correspondiente y la cantidad de filas es igual a la cantidad total de horas cátedra disponibles en una semana. Recordar que en nuestro caso de estudio, el turno tarde cuenta con prehora, lo cual supone que hay disponibles para asignar 8 horas cátedra de clase por día, sumando un total de 40 horas cátedra disponibles en una semana. Si bien no hay prehora para el turno mañana, igual se considera su existencia para homogeneizar la cantidad de horas cátedra disponibles en la semana y de esta manera simplificar la implementación de los algoritmos. Para la generación de horarios se considerará que todos los días en la prehora del turno mañana no está asignada ninguna materia para ningún curso. Entonces, sea G una grilla, en G_{ij} se encuentra qué materia se da en la hora i en el curso j; si $G_{ij} = 0$, significa que el curso j no tiene asignada ninguna materia en la hora i. A continuación definimos formalmente a qué nos referiremos por grillas en lo que sigue de este trabajo:

Definición 4.1. (Grilla) Sean $\hat{\eta}$ un horario de cursada válido, su grilla $G_{\hat{\eta}}$ es la matriz de dimensión $40 \times N$, con N la cantidad de cursos, tal que:

$$G_{\hat{\eta}}^{(ij)} = \begin{cases} m & \text{si} \quad ((C_j, m, P), h_i) \in \hat{\eta} \text{ para algún } P \subseteq \mathcal{P},\ P \neq \emptyset \\ 0 & c.c. \end{cases}$$

Observación. Es importante que un horario sea válido para poder definir su grilla, puesto que si, por ejemplo, el curso j tiene asignadas dos materias en la hora i, no se puede definir $G_{\hat{\eta}}^{(ij)}$.

Notación 4.1. Recordar que los cursos se dividen en dos turnos. Por tanto, se introduce la siguiente notación: C_{TM} denota al conjunto de índices de cursos del turno mañana y C_{TT} al del turno tarde.

[1] Entendiendo como matriz en este contexto a una tupla de tuplas o, mas específico en el caso de la implementación en Python, una lista de listas de objetos

En nuestro caso de estudio hay doce cursos en el turno mañana y diez en el turno tarde, por lo que $C_{TM} = \{0, \ldots, 11\}$ y $C_{TT} = \{12, \ldots, 21\}$.

Como se verá en los capítulos siguientes, la resolución de la Etapa I se basa en las grillas. Por ejemplo, las mismas son esenciales para definir la vecindad en el marco de los algoritmos de búsqueda.

NOMBRE: P0		TURNO: TM			
HORAS	LUNES	MARTES	MIERCOLES	JUEVES	VIERNES
PH					
1º		C2			C0
2º		C2			C0
3º		C1			C1
4º		C1			C2
5º		C0			C3
6º		PROY1			C3
7º					C3

Figura 4.2: Ficha del docente p_0 que detalla sus actividades durante el turno mañana.

Recordemos que puede ocurrir que un docente de clases tanto durante el turno mañana como durante el turno tarde. Por esta razón, dejando a la disponibilidad de lado por el momento, cada docente tiene 16 horas cátedra por día durante las cuales puede enseñar: las 8 horas del turno mañana más las 8 del turno tarde. Si bien la prehora del turno tarde y la séptima hora del turno mañana ocurren durante el mismo intervalo de tiempo, en el modelo son representadas como si esto no ocurriese. Entonces se tiene que, en total, los docentes tienen 80 horas cátedra por semana en las que es posible asignarles clases o extraclases. Por esta razón, se decidió representar a las fichas de los profesores como tuplas[2] de longitud 80. En la i-ésima coordenada de la ficha del docente p, F_p, se halla qué actividad está realizando el profesor, o 0 si no está en la escuela o si es una hora inactiva. Recordar que se categorizaron las tareas de los docentes en tres grupos: clases y extraclases. Si en la hora i-ésima el docente está enseñando, en $F_p^{(i)}$ se encuentra el conjunto de cursos donde lo está haciendo; si el docente se encuentra dando extraclases o en alguna actividad que no involucra a los alumnos, en $F_p^{(i)}$ se encuentra el nombre de la ocupación (proyecto, tarea, clase de apoyo, etc.). Presentamos así la definición de ficha que se utilizará de aquí en adelante:

Definición 4.2. (Ficha) Sea η un horario donde no se superpongan una clase y una extraclase que involucren al mismo docente, y sea $p \in \mathcal{P}$ un docente, se define F_p la ficha correspondiente al docente p como la tupla de longitud 80 cuya i-ésima coordenada viene dada por:

$$F_p^{(i)} = \begin{cases} \mathcal{C}_i & \text{si} \quad \mathcal{C}_i = \{C : (C, m, P), h_i) \in \eta,\ P \subseteq \mathcal{P},\ p \in P,\ C \in \{C_j\}_{j=0}^{N-1}\} \neq \emptyset \\ m & \text{si} \quad ((\emptyset, m, P), h_i) \in \eta,\ p \in P \quad \text{o si} \quad ((\mathcal{A}, m, P), h_i) \in \eta,\ p \in P \\ 0 & \text{c.c.} \end{cases}$$

Observación. En este caso no es necesario que el horario sea válido, puesto que la definición acepta que un docente se encuentre en distintos cursos al mismo tiempo incluso si no hay simultaneidades involucradas. Sí es importante que no se superpongan distintos tipos de actividades para el docente, de lo contrario $F_p^{(i)}$ no está bien definida.

Definición 4.3. Si se codifican los días de la semana como $0 =$ Lunes, $\ldots$, $4 =$ Viernes y los turnos como $0 =$ turno mañana, $1 =$ turno tarde, la función que 'traduce' una hora en el formato de $0 - 79$ a

[2]Implementado en Python como listas para aprovechar su mutabilidad.

día, hora cátedra y turno viene dada por $\tau : \{0, \ldots, 79\} \to \{0, \ldots, 4\} \times \{0, \ldots, 7\} \times \{0, 1\}$ y se define de la siguiente manera:

$$\tau(i) = (\tau_1(i),\, \tau_2(i),\, \tau_3(i)) \quad \text{donde}^3\text{:}$$

$$\begin{aligned} \tau_1(i) &= \left\lfloor \frac{i}{16} \right\rfloor \\ \tau_2(i) &= i \mod 8 \\ \tau_3(i) &= \left\lfloor \frac{i}{8} \right\rfloor \mod 2 \end{aligned}$$

Por otro lado, se define $\tau^{-1} : \{0, \ldots, 4\} \times \{0, \ldots, 7\} \times \{0, 1\} \to \{0, \ldots, 79\}$ como:

$$\tau^{-1}(d, h, t) = 16d + h + 8t$$

Ejemplo 4.1. Sean p un docente y F_p su ficha, tales que $F_p^{(20)} = \{C_6, C_7\}$, esto significa que p se encuentra dando clases en los cursos 6 y 7 durante la cuarta hora ($20 \equiv 4 \pmod 8$) del día Martes ($20 \equiv 1 \pmod{16}$) en el turno mañana ($\lfloor \frac{20}{8} \rfloor \equiv 0 \pmod 2$). Si se quisiera saber qué actividad desarrolla el docente p durante la cuarta hora del turno tarde del día Jueves, basta observar el valor de $F_p^{(60)}$, puesto que $60 = 16 \cdot 3 + 4 + 8 \cdot 1$.

Las fichas de los docentes resultarán de gran utilidad no sólo para calcular la penalidad por la violación de las restricciones que los incumben, sino también para evaluar la validez de un vecino en el contexto del algoritmo de búsqueda, como se verá en la sección 6.1.

Teniendo en cuenta las definiciones de horario de cursada (definición 3.10), de grilla (definición 4.1) y de ficha (definición 4.2), se deduce que a partir de un horario de cursada $\hat{\eta}$ se puede obtener su grilla, y a partir de ella, las fichas de los docentes (salvo horas extraclase). Asimismo, dada una grilla, se pueden obtener las fichas de los docentes (salvo horas extraclase) e inferir un horario de cursada. Por tanto, trabajar con un horario de cursada es equivalente a trabajar con una grilla.

Además, a partir de las grillas y de las fichas se puede obtener la información necesaria para calcular la penalidad por violar las restricciones blandas del horario de cursada. Por ejemplo, la cantidad de prehoras para el turno tarde el Martes puede ser calculado de la siguiente manera:

$$\text{Cantidad de prehoras} = |\{j \in C_{TT} \,:\, G_\eta^{(8,j)} \neq 0\}|$$

Asimismo la grilla permite calcular cuántas horas de clase tiene un curso durante un día, cuántos bloques tiene una materia en un determinado curso, cuánto se usa cierto recurso durante una hora cátedra, etc. Por otro lado, las fichas otorgan la información necesaria para conocer si se está asignando un docente a un horario en el que no está disponible, medir cuántas horas libres tienen los docentes, dónde ubicar las extraclases, etc. Luego, a partir de una grilla y de su correspondiente conjunto de fichas, se tiene suficiente información para describir, en términos de penalización, cuán bueno es el horario de cursada. Por esta razón, en la sección 4.3 se describirá la función de penalidad $\hat{\Psi}$ en base a la grilla y a las fichas correspondientes a un horario de cursada $\hat{\eta}$. Pero antes, es necesario introducir el concepto de *restricciones duras suavizadas*.

[3] La operación $x \mod y$ representa el resto positivo de dividir a x por y

4.2. Restricciones duras suavizadas

En la sección 2.3 se presentó $\mathcal{R}_1$ el conjunto de restricciones para la creación de un horario de cursada, de las cuales más de la mitad son restricciones duras. La experiencia proporcionada por la investigación en el campo del HSSP indica que, en general, lo más eficiente es tener la menor cantidad de restricciones duras posible [26]. Esto significa que dentro del modelo habrán restricciones blandas cuya violación no permitirá que un horario sea aplicable en la práctica y, por ende, no sea aceptable. Este tipo particular de restricciones reciben el nombre de *restricciones duras suavizadas*.

Hay varias razones para *suavizar* ciertas restricciones duras [26]. En primer lugar, imponer tantas restricciones sobre el horario hace mucho más difícil la tarea de crear uno que sea `VALIDO` y suele resultar más sencillo optimizar un horario que viole alguna de las restricciones duras suavizadas que armar un horario válido desde cero. Otra razón es que las restricciones duras obstaculizan a los algoritmos de optimización local que parten desde un horario válido: por un lado, con menos restricciones duras es más sencillo encontrar un 'camino' entre soluciones y, por el otro, es más fácil encontrar soluciones válidas en la vecindad.

Dentro del modelo, la penalidad asociada a violar una de las restricciones duras suavizadas es significativamente superior a la correspondiente a la violación de restricciones blandas. De esa manera, se espera que el algoritmo de búsqueda local priorice satisfacer todas las restricciones duras suavizadas y confeccionar un horario que sea aplicable en la práctica. En nuestro caso, se estableció que la penalidad por violar una restricción dura suavizada sea de 7000, mientras que las restricciones blandas tienen penalidades con valores que oscilan entre 80 y 600 por unidad.

En el caso de estudio que se analiza en este trabajo, no se encontró dificultad en el armado de horarios válidos considerando la totalidad de las restricciones duras. Sin embargo, resulta interesante incluir el concepto de restricciones duras suavizadas para permitir una exploración más amplia del espacio de búsqueda y generar horarios iniciales de manera casi instantánea. Por esa razón, en esta tesis se 'suavizan' las siguientes restricciones duras:

- disponibilidad de los docentes
- séptima hora del turno tarde del Lunes

La motivación detrás de la 'suavización' de esas restricciones es que durante la creación de un horario inicial existe la posibilidad de asignar materias a horas donde en la práctica no podrían ser asignadas. Si bien esto no resultará en un horario válido, el proceso de reparación del horario inicial podría guiar al algoritmo de búsqueda local hacia otras soluciones. Más aún, en el caso del *hill-climbing* estocástico, donde se acepta con cierta probabilidad la elección de un vecino con una penalidad más alta, podría permitir que la búsqueda explore otros caminos tomando, por ejemplo, un horario donde hay una séptima hora del Lunes en algún curso del turno tarde y posteriormente reparándola. La razón por la cual no se suavizaron más restricciones duras fue porque las estructuras de grilla y de ficha junto a la noción de vecindad hacen que evaluar la validez de un horario de cursada vecino sea sencillo y rápido. Además, otras restricciones duras como la que establece la cantidad máxima de horas cátedra diarias, no podrían ser suavizadas debido a que atentarían contra la estructura de las grillas y de las fichas. En el cuadro 4.1 queda descripta cómo se clasifican las restricciones presentadas en la sección 2.3.

En la sección 3.1 se introdujo el concepto de horario `VALIDO`: un horario que no violase ninguna restricción dura. Luego, en esta sección, se introdujo el concepto de restricción dura suavizada: restric-

Restricciones duras ($\mathcal{R}_{1H}$)

- Espacio y tiempo
- Simultaneidades
- Cantidad de horas diarias de cursada
- Cantidad diaria máxima de horas cátedra de una materia
- Carga horaria semanal
- Sin horas libres para los alumnos
- Cargos de los docentes

Restricciones blandas ($\mathcal{R}_{1S}$)

Restricciones duras suavizadas:

- Disponibilidad docente
- Séptima hora de los Lunes en el Turno Tarde

Restricciones blandas comunes:

- Sin huecos en el dictado de materias
- Cantidad de bloques
- Cantidad de triples
- Horas inactivas de los docentes
- Recursos
- Cantidad de prehoras
- Recreos

Cuadro 4.1: Clasificación de las restricciones para el armado del horario de cursada

ciones que son modeladas como restricciones blandas pero que en la práctica no pueden ser violadas. Esto supone una contradicción entre lo que se busca (un horario donde tanto restricciones duras como algunas restricciones blandas sean completamente satisfechas) y la definición dada en la sección 3.1 que sólo requería no que se violen las restricciones duras. Por esa razón, debemos introducir el concepto de un horario *aceptable*.

Definición 4.4. (Función de aceptabilidad) Sea $\texttt{ACEPTABLE} : \mathbb{P}(\mathcal{L}_C \times \mathcal{H}) \to \{0, 1\}$. `ACEPTABLE` es la función de aceptabilidad de un horario de cursada $\hat{\eta} \in \mathbb{P}(\mathcal{L}_C \times \mathcal{H})$, por la cual $\texttt{ACEPTABLE}(\eta) = 1$ significa que el horario de cursada es `VALIDO` y que ningún sumando que compone a la función de penalidad $\hat{\Psi}$ supere 7000.

Puesto que 7000 es la penalidad asignada a cada violación de una restricción dura suavizada, es claro que un horario de cursada `ACEPTABLE` no violará ninguna de ellas y por ende será aplicable en la práctica. Sin embargo, podría resultar que un horario de cursada no sea `ACEPTABLE` incluso si respeta todas las restricciones duras suavizadas (por ejemplo, podría tener demasiadas horas libres para los profesores).

Podemos ahora extender a definición formal del problema dada en la sección 3.2:

$$\begin{array}{ll} \text{Dados} & \mathcal{L}_C, \mathcal{R}_{1S}, \mathcal{H} \\ \text{minimizar} & \hat{\Psi}(\hat{\eta}, \mathcal{R}_{1S}) \\ \text{sujeto a} & \texttt{ACEPTABLE}(\hat{\eta}) = 1, \\ & (l, h_l) \in \hat{\eta} \qquad l \in \mathcal{L}_C \, , \, h \in \mathcal{H} \end{array}$$

Notar que el requerimiento de validez del horario está incluido en el requerimiento de aceptabilidad. Asimismo, esta formulación del problema establece que la minimización de $\hat{\Psi}(\hat{\eta}, \mathcal{R}_S)$ es secundaria a la aceptabilidad de un horario: se desea hallar el horario con la mejor penalidad posible, pero en cualquier caso se necesita que el horario sea aceptable.

4.3. $\hat{\Psi}$, Función de penalidad del horario de cursada

Como se ha mencionado en la sección 3.2, la penalidad de un horario de cursada es la suma de las penalidades en las que se incurrió por violación de las distintas restricciones blandas. Por otro lado, en la sección 4.1 se concluyó que un horario de cursada induce una única grilla, de la cual a su vez se pueden obtener las fichas de los docentes (salvo horas extraclase) y que una grilla puede inducir un único horario de cursada. Luego, se puede definir la función de penalidad de un horario de cursada en base a la grilla y a las fichas que lo representan.

Notación 4.2. Dado $\hat{\eta}$ un horario de cursada, se nota como $G_{\hat{\eta}}$ a la grilla que induce y como $\hat{F}_p$ a la ficha (salvo horas extraclase) del docente p que se obtiene de $G_{\hat{\eta}}$. Finalmente, puesto que la penalidad del horario de cursada es la misma que la penalidad de su grilla inducida y para no complicar aún más la notación, consideramos lo siguiente:

$$\hat{\Psi}(\hat{\eta}, \mathcal{R}_{1S}) = \hat{\Psi}(G_{\hat{\eta}})$$

Es decir, ahora notamos a $\hat{\Psi}$ como la función de penalidad de la grilla inducida por $\hat{\eta}$ con respecto al conjunto de restricciones blandas $\mathcal{R}_{1S}$.

Dada una grilla $G_{\hat{\eta}}$, $\hat{\Psi}(G_{\hat{\eta}})$ resulta de la suma de las penalidades por violar cada una de las restricciones en $\mathcal{R}_{1S}$. A continuación se detalla cada una de ellas:

Disponibilidad docente

> Puesto que se trata de una restricción dura suavizada, para que el horario de cursada resulte `ACEPTABLE` es necesario, pero no suficiente, que la penalidad incurrida por asignarle a un docente una clase en un periodo de tiempo en el que no está disponible sea 0. Para implementar esta función se requiere tener la información de la disponibilidad de un docente en un formato que permita ser comparable fácilmente con la ficha del mismo.

Definición 4.5. (Vector de disponibilidad) Sea p un docente, definimos $Disp_p$ como el vector que representa su disponibilidad horaria en una semana. $Disp_p \in \{0,1\}^{80}$ y está definido por:

$$Disp_p^{(i)} = \begin{cases} 1 & \text{si el docente } p \text{ está disponible en la hora } i \\ 0 & \text{cc} \end{cases}$$

Vale notar que se utiliza la función τ (definición 4.3) para traducir coloquialmente una hora en forma $0-79$. Por lo tanto, la coordenada i-ésima de $\hat{F}_p$ corresponde a la misma hora cátedra que la coordenada i-ésima de $Disp_p$. Ahora se está en condición de definir la función de penalidad por no respetar la disponibilidad del docente:

$$\hat{\Psi}_{disp}(G_{\hat{\eta}}) = \sum_{p \in \mathcal{P}} |\{i \in \{0,\ldots,79\} \,:\, Disp_p^{(i)} = 0 \ \wedge \ \hat{F}_p^{(i)} \neq 0\}|$$

Séptima Hora del Lunes para el Turno Tarde

Gracias a la representación del horario de cursada en la grilla, calcular la violación de esta restricción resulta sencillo:

$$\hat{\Psi}_{sept}(G_{\hat{\eta}}) = |\{j \in C_{TT} \,:\, \hat{G}_{\hat{\eta}}^{(7,j)} \neq 0\}|$$

Sin huecos en el dictado de materias

Sean $m \in \mathcal{M}$ una materia y C_j un curso, se puede considerar el conjunto de horas en las que C_j tiene asignada a m como $H_{(m,j)} = \{i \in \{0,\ldots,39\} \,:\, G_{\hat{\eta}}^{(i,j)} = m\}$. Veamos con un ejemplo el proceso por el cual se calcula la cantidad de huecos en el dictado de m en C_j a partir de $H_{(m,j)}$: supongamos que $H_m = \{8,9,13,25,26,33\}$. Se agrupan las horas de H_m según el día al que corresponde cada una: dos horas corresponden al mismo día si la división entera[4] por 8 da el mismo resultado. En el ejemplo, se tiene que $\{\{8,9,13\},\{25,26\},\{33\}\}$. Se ignoran los conjuntos resultantes con cardinal igual a uno (en el ejemplo $\{33\}$) puesto que esos días la materia se dicta sólo durante una hora y por ende no pueden haber huecos. En nuestro ejemplo la situación es ahora $\{\{8,9,13\},\{25,26\}\}$. En cada uno de los conjuntos, se extraen las horas que conforman bloques (es decir, que son consecutivas). En nuestro ejemplo:

$$\left.\begin{array}{ccc} \{8,9,13\} & \to & \{13\} \\ \{25,26\} & \to & \emptyset \end{array}\right\} \Rightarrow \{\{8,9,13\},\{25,26\}\} \to \{\{13\},\emptyset\}$$

Finalmente, se obtiene la cantidad de huecos para m en C_j sumando el cardinal de cada uno de los conjuntos. En el ejemplo, m tiene 1 hueco en C_j. Se nota como `cant_huecos` a la función que realiza este procedimiento sobre $H_{(m,j)}$. Si m no se dicta en C_j, entonces $H_{(m,j)} = \emptyset$ y se define que $\texttt{cant\_huecos}(H_{(m,j)}) = 0$. La función de penalidad para la cantidad de huecos en el dictado de las materias queda entonces definida como:

$$\hat{\Psi}_{huecos}(G_{\hat{\eta}}) = \sum_{j=0}^{N-1} \sum_{m \in \mathcal{M}} \texttt{cant\_huecos}(H_{(m,j)})$$

[4]Se entiende por la división entera de x por y a la operación $\left\lfloor \frac{x}{y} \right\rfloor$

Cantidad de bloques

Para calcular la cantidad de bloques de una materia $m \in \mathcal{M}$ en un curso C_j, también recurrimos al conjunto $H_{(m,j)}$ definido previamente. Tal y como se hizo en el caso anterior, se agrupan las horas cátedra por día. Luego, en cada conjunto, se cuentan la cantidad de subconjuntos maximales (de cardinal mayor que uno) de horas consecutivas, que representan a los bloques. Observar que los bloques pueden estar compuestos por más de dos horas consecutivas. En el ejemplo donde $H_{(m,j)} = \{8, 9, 13, 25, 26, 33\}$, se agrupan las horas según el día, obteniendo $\{\{8, 9, 13\}, \{25, 26\}, \{33\}\}$. En cada elemento de este conjunto se cuentan los subconjuntos maximales que representan a los bloques:

$$\begin{array}{rl} \{\mathbf{8}, \mathbf{9}, 13\} & \rightarrow \text{un bloque} \\ \{\mathbf{25}, \mathbf{26}\} & \rightarrow \text{un bloque} \\ \{33\} & \rightarrow \text{ningún bloque} \end{array}$$

Entonces, se tiene que m conforma 2 bloques en C_j. Notaremos la cantidad de bloques que conforma m en C_j como $b_{(m,j)}$. Si m no se dicta en C_j, se define $b_{(m,j)} = 0$. La cantidad óptima de bloques óptima para m en C_j viene dada por la división entera por 2 de la carga horaria semanal de m en C_j y se nota como $b^{opt}_{(m,j)}$. Se define la función `pen_bloques` como:

$$\texttt{pen\_bloques}(m, j) = \begin{cases} 0 & \text{si } b_{(m,j)} = b^{opt}_{(m,j)} \\ 1 & \text{si } (b^{opt}_{(m,j)} = 2 \wedge b_{(m,j)} = 1) \vee (b^{opt}_{(m,j)} = 3 \wedge b_{(m,j)} = 2) \\ 2 & \text{si } b^{opt}_{(m,j)} = 3 \wedge b_{(m,j)} = 1 \\ 5 & \text{si } b_{(m,j)} = 0 \end{cases}$$

El caso en el que una materia no forma ningún bloque es el más penalizado, seguido por los casos en los que no se forman todos los bloques posibles. Se define la función de penalidad por no conformar bloques como:

$$\hat{\Psi}_{bloques}(G_{\hat{\eta}}) = \sum_{j=0}^{N-1} \sum_{m \in \mathcal{M}} \texttt{pen\_bloques}(m, j)$$

Cantidad de triples

Una vez más, se utilizará el conjunto $H_{(m,j)}$ de las horas n las que se dicta m en el curso C_j. Al igual que antes, se agrupan los elementos de $H_{(m,j)}$ por día. Luego, se calcula el cardinal de cada grupo; se aplica una penalidad por cada uno que tenga tres elementos. Como se verá posteriormente, al ejecutar el algoritmo de búsqueda local se podría llegar a un horario de cursada donde hayan más de tres horas de una materia en el mismo día. Estos casos son contemplados imponiendo una penalidad más alta si la cantidad de horas de un día es mayor o igual que 4.

Retomando el ejemplo de $H_{(m,j)} = \{8, 9, 13, 25, 26, 33\}$, se agrupan las horas según el día, obteniendo $DH_{(m,j)} = \{\{8, 9, 13\}, \{25, 26\}, \{33\}\}$. Se calcula luego el cardinal de cada elemento; se suma una unidad por cada elemento con cardinal igual a 3 y tres unidades por cada elemento con cardinal mayor o igual que 4. Se introduce entonces la siguiente función:

$$\texttt{cant\_triples}(m, j) = |\{S \in DH_{(m,j)} \colon |S| = 3\}| + 3 \cdot |\{S \in DH_{(m,j)} \colon |S| \geq 4\}|$$

Se define la función que penaliza la cantidad de triples en un horario de cursada como:

$$\hat{\Psi}_{triples}(G_{\hat{\eta}}) = \sum_{j=0}^{N-1} \sum_{m \in \mathcal{M}} \texttt{cant\_triples}(m, j)$$

Horas inactivas de los docentes

Sea p un docente y sea $\hat{F}_p$ su ficha asociada a $G_{\hat{\eta}}$, se agrupan las coordenadas de $\hat{F}_p$ por día[5]. Se descartan los días, si los hay, en los que el docente no asiste a la escuela, pues resultan irrelevantes para calcular horas inactivas. En cada uno de los días restantes, se calcula el índice del primer y del último elemento no nulo; la cantidad de horas inactivas es igual a la cantidad de ceros entre ambos índices[6]. Finalmente, se suma la cantidad de horas inactivas de cada día. Se denomina como `idle` a la función que calcula las horas libres totales de p a partir de $\hat{F}_p$.

Para fijar ideas, se presenta el siguiente ejemplo de un docente que sólo asiste los Lunes:

$$\hat{F}_p = (0, \{C_1\}, \{C_1\}, \{C_2\}, 0, 0, \{C_3\}, \{C_3\}, 0, \{C_{14}\}, \{C_{14}\}, 0, \ldots, 0)$$

Se agrupan las coordenadas por días y se descartan las correspondientes a los días en los que el docente no asiste. Sólo sobrevive la tupla correspondiente al Lunes. Se calcula entonces la cantidad de horas libres de p:

$$\hat{F}_p^{Lunes} = (0, \underbrace{\{C_1\}}_{\text{Primer índice}}, \{C_1\}, \{C_2\}, \underbrace{0, 0}_{\text{horas libres}}, \{C_3\}, \{C_3\}, 0, \{C_{14}\}, \underbrace{\{C_{14}\}}_{\text{Último índice}}, 0, 0, 0, 0, 0)$$

Luego, $\texttt{idle}(\hat{F}_p) = 2$.

Para definir la función de penalidad se debe tener en cuenta que el costo de tener una hora libre para un docente que sólo enseña dos horas en una escuela no es el mismo que para un docente con veinticuatro horas. Es decir, existe una relación entre el cargo y cuánto afecta a cada docente tener horas libres. Además, se requiere que la penalidad crezca rápidamente a medida que la cantidad de horas inactivas se acerca a la cantidad de horas activas. Naturalmente, sin importar el tamaño del cargo del docente, tener más horas inactivas que horas activas representa una situación muy poco favorable. Por eso, la función de penalidad realiza un salto pronunciado cuando la cantidad de horas inactivas iguala al cargo del docente. Sean c_p el cargo del docente p y ϖ_p la cantidad de horas extraclase (que, en la Etapa II se intentarán asignar a las horas libres que hayan quedado en la ficha resultante de la Etapa I), se define la siguiente penalidad para las horas inactivas de un docente en función de su cantidad de horas libres:

[5] En este caso, dos coordenadas pertenecen al mismo día si su división entera por 16 da el mismo resultado

[6] En la implementación de este cálculo se tomó especial cuidado con la séptima hora del turno mañana y la prehora del turno tarde: puesto que ocurren durante el mismo periodo de tiempo, si el docente está libre durante ambas, se cuenta como una sóla hora libre; asimismo si el docente se encuentra ocupado durante alguna de las dos, no se cuenta a la otra como una hora inactiva.

$$\texttt{cuant\_idle}_p(\hat{F}_p) = \begin{cases} \dfrac{1}{1-\exp\left(\sqrt{\dfrac{\text{máx}\{\texttt{idle}(\hat{F}_p)-\varpi_p,0\}}{c_p/2}}-1\right)} - \dfrac{1}{1-\exp(-1)} & \text{si } \texttt{idle}(\hat{F}_p)-\varpi_p < \dfrac{c_p}{2} \\ \left(2\cdot\dfrac{\texttt{idle}(\hat{F}_p)}{c_p/2}\right)^4 + \dfrac{1}{1-\exp\left(\sqrt{\dfrac{c_p/2-1}{c_p/2}}-1\right)} - \dfrac{1}{1-\exp(-1)} & \text{c.c.} \end{cases}$$

En la figura 4.3 se puede observar el comportamiento de $\texttt{cuant\_idle}_p$ para un docente p cuyo cargo es de 12 horas y en el cuadro 4.2 se observa cómo varía su valor para docentes de distinto cargo.

Cargo \ $\texttt{idle}(\hat{F}_p)-\varpi_p$	2	4	8	16
2	91.78	1291.78	20491.78	327691.78
4	11.78	111.97	1311.97	20511.97
12	3.28	6.6	21.91	364.89
18	2.37	4.27	9.73	82.04
24	1.91	3.28	6.6	21.91

Cuadro 4.2: Valores de $\texttt{cuant\_idle}_p$ para distintos cargos y distintos valores de $\texttt{idle}(\hat{F}_p)-\varpi_p$

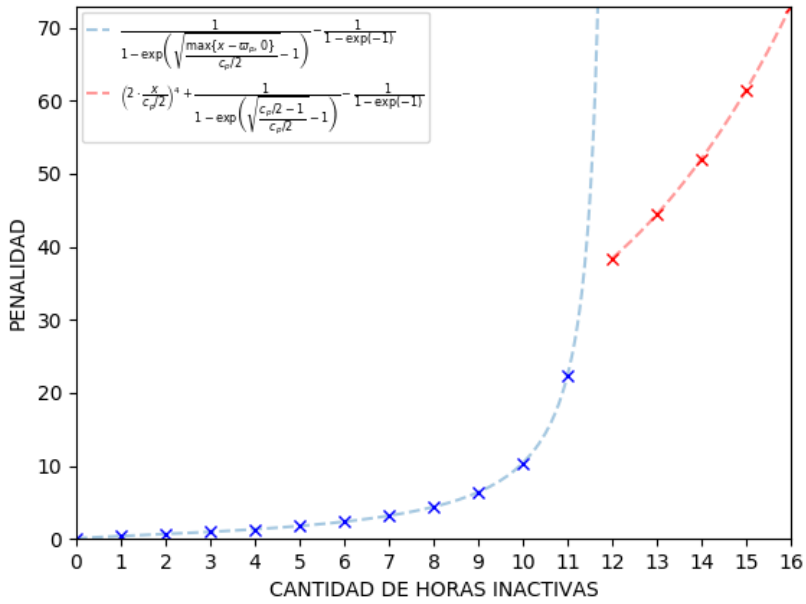

Figura 4.3: Gráfico de la función de penalidad de un docente con un cargo de 12 horas

A partir de `cuant_idle` se define la función que penaliza las horas inactivas de los docentes como:

$$\hat{\Psi}_{idle}(G_{\hat{\eta}}) = \sum_{p \in \mathcal{P}} \texttt{cuant\_idle}(\hat{F}_p)$$

Recursos

Cada escuela cuenta con una cierta cantidad de recursos que tienen distintos niveles de prioridad según cuán necesarios son para dar una clase. Podría ocurrir entonces que las restricciones sobre la utilización de ciertos recursos sean consideradas restricciones duras. En el caso de estudio no se presentó esa situación, pero de haber ocurrido, se puede modelar como una restricción dura suavizada introduciendo un coeficiente de prioridad muy alto, como se verá debajo. Para la mayoría de los recursos se puede permitir que su demanda exceda su capacidad durante una hora cátedra, aunque es preferible que esto ocurra la menor cantidad de veces posible. Esto es contemplado en la penalidad.

Dada una lista de materias, los recursos que requiere cada una y la prioridad de la utilización de los mismos (es decir, cuán necesario es el recurso para desarrollar una clase de esa materia), a partir de $G_{\hat{\eta}}$ se pueden obtener los datos necesarios para calcular en cuántas horas cátedra hay una demanda excesiva. Por ejemplo, si la escuela cuenta con una sala de computación, basta con identificar las horas cátedra durante las cuales se cursan las materias de Informática y cuantificar la penalización acorde la cantidad de horas donde la requieren dos o más cursos. Naturalmente, recibirá mayor penalización el uso excesivo de un recurso con mayor prioridad que el de uno con prioridad baja.

La función que cuantifica la penalidad por uso excesivo de un recurso se define teniendo en cuenta los aspectos mencionados. Dado un recurso, se calcula en cada hora cátedra por cuánto la demanda excede a la capacidad. Por ejemplo, si en la tercera hora del Lunes hay tres cursos cursando Informática en una escuela con una sala de informática, hay un exceso de dos. Luego se suman los excesos y se los multiplica por la cantidad de veces que la demanda supera a la capacidad. Se multiplica el resultado por la prioridad del recurso para amplificar la penalidad según su prioridad. Este procedimiento se lleva a cabo para ambos turnos por separado, puesto que el uso de recursos del turno mañana no influye en el del turno tarde[7]. A continuación se construye la función que cuantifica la penalidad.

Sea R el conjunto de recursos con los que cuenta la escuela, y para cada $r \in R$ sea $M_r \subseteq \mathcal{M}$ el conjunto de materias que requieren al recurso r. Se puede calcular la cantidad de veces que se utiliza al recurso r en la hora cátedra h de cada turno con las siguientes funciones:

$$U_{TM}(G_{\hat{\eta}}, r, h) = |\{G_{\hat{\eta}}^{(h,j)} \colon\ G_{\hat{\eta}}^{(h,j)} = m,\ m \in M_r,\ j \in C_{TM}\}|$$

$$U_{TT}(G_{\hat{\eta}}, r, h) = |\{G_{\hat{\eta}}^{(h,j)} \colon\ G_{\hat{\eta}}^{(h,j)} = m,\ m \in M_r,\ j \in C_{TT}\}|$$

Por otro lado, notando a la capacidad del recurso r como cap_r, se puede calcular durante cuántas horas cátedra su demanda excede a su capacidad en cada turno como:

$$O_{TM}(G_{\hat{\eta}}, r) = |\{h \in \mathcal{H} \colon\ U_{TM}(G_{\hat{\eta}}, r, h) > cap_r\}|$$

[7] Salvo en el caso de las séptimas horas del turno mañana y las prehoras del turno tarde. Este caso especial se tuvo en cuenta en la implementación.

$$O_{TT}(G_{\hat{\eta}}, r) = |\{h \in \mathcal{H} \colon\ U_{TT}(G_{\hat{\eta}}, r, h) > cap_r\}|$$

Sea $prioridad_r$ la prioridad del recurso r, se define la siguiente penalidad para el uso excesivo de recursos en una hora cátedra h para cada uno de los turnos:

$$\hat{\Psi}_{recs}(G_{\hat{\eta}}) = \sum_{r \in R} prioridad_r \cdot \Bigg(O_{TM}(G_{\hat{\eta}}, r) \cdot \sum_{h \in \mathcal{H}} \text{máx}\{U_{TM}(G_{\hat{\eta}}, r, h) - cap_r, 0\} + O_{TT}(G_{\hat{\eta}}, r) \cdot \sum_{h \in \mathcal{H}} \text{máx}\{U_{TT}(G_{\hat{\eta}}, r, h) - cap_r, 0\} \Bigg)$$

Cantidad de prehoras

En el caso de estudio, se prefiere que no hayan más de 6 prehoras diarias del turno tarde, por lo tanto la función que cuantifica la penalización por exceso de prehoras se define como:

$$\hat{\Psi}_{prehoras}(G_{\hat{\eta}}) = \sum_{\substack{h \in \mathcal{H} \\ h \equiv 0 \pmod 8}} \text{máx}\{|\{G^{(h,j)}_{\hat{\eta}} \colon G^{(h,j)}_{\hat{\eta}} \neq 0 \ \wedge\ j \in C_{TT}\}| - 6, 0\}$$

Recreos

En el caso de estudio, hay dos recreos y estos ocurren entre la segunda y la tercera hora y entre la cuarta y la quinta de cada turno. Se prefiere que los recreos no se encuentren en medio del bloque de una materia. Entonces se define la función que cuantifica la penalidad por recreos que interrumpen bloques como:

$$\hat{\Psi}_{recreos}(G_{\hat{\eta}}) = \sum_{\substack{h \in \mathcal{H} \\ h \equiv 2 \pmod 8}} |\{G^{(h,j)}_{\hat{\eta}} \colon G^{(h,j)}_{\hat{\eta}} = G^{(h+1,j)}_{\hat{\eta}},\ j \in C_{TM} \cup C_{TT}\}| + \sum_{\substack{h \in \mathcal{H} \\ h \equiv 4 \pmod 8}} |\{G^{(h,j)}_{\hat{\eta}} \colon G^{(h,j)}_{\hat{\eta}} = G^{(h+1,j)}_{\hat{\eta}},\ j \in C_{TM} \cup C_{TT}\}|$$

A partir de las funciones que cuantifican la penalidad por la violación de cada una de las restricciones blandas, se define la función que calcula la penalidad de una grilla como:

$$\hat{\Psi}(G_{\hat{\eta}}) = k_{disp} \cdot \hat{\Psi}_{disp}(G_{\hat{\eta}}) + k_{sept} \cdot \hat{\Psi}_{sept}(G_{\hat{\eta}}) + k_{huecos} \cdot \hat{\Psi}_{huecos}(G_{\hat{\eta}}) + k_{bloques}\hat{\Psi}_{bloques}(G_{\hat{\eta}}) + k_{triples} \cdot \hat{\Psi}_{triples}(G_{\hat{\eta}}) + k_{idle} \cdot \hat{\Psi}_{idle}(G_{\hat{\eta}}) + k_{recs} \cdot \hat{\Psi}_{recs}(G_{\hat{\eta}}) + k_{prehoras} \cdot \hat{\Psi}_{prehoras}(G_{\hat{\eta}}) + k_{recreos} \cdot \hat{\Psi}_{recreos}(G_{\hat{\eta}})$$

donde k representan los coeficientes de las penalidades. Mientras mayor es k, mayor es la pena por unidad de penalidad incurrida por violar la correspondiente restricción blanda. Por ejemplo, como la disponibilidad docente es una restricción dura suavizada, $k_{disp} = 7000$. Los coeficientes se ajustan para representar la prioridad, teniendo en cuenta cómo escalan las unidades de penalidad de las distintas funciones cuantificadoras. Por ejemplo, $\hat{\Psi}_{idle}(G_{\hat{\eta}})$ escala mucho más rápido que $\hat{\Psi}_{triples}(G_{\hat{\eta}})$, entonces $k_{triples}$ deberá ser significativamente mayor que k_{idle} (en el caso de estudio, $k_{idle} = 80$ y $k_{triples} = 650$).

Una vez definidos el modelo y la función de penalidad, se procede a generar una variedad de horarios de cursada sobre los que se aplicarán más tarde los algoritmos de optimización local. El siguiente capítulo describe el procedimiento de generación de horarios de cursada iniciales.

Capítulo 5

Generación de un horario de cursada inicial

Como se ha adelantado en la sección 3.2, la generación de horarios de cursada válidos es el primer paso en la búsqueda de horarios de cursada aceptables de buena calidad. Se desea generar varios horarios de cursada válidos para tomarlos como puntos iniciales de los algoritmos de búsqueda local y de esa manera poder explorar más ampliamente el espacio de horarios de cursada. Recordar que el objetivo no es hallar el mínimo global de la función de penalización, sino encontrar mínimos locales que resulten horarios de cursada aceptables. Se generan horarios de cursada iniciales admisibles[1] y no necesariamente aceptables para estudiar el comportamiento de los algoritmos de optimización local y para explorar más ampliamente el espacio de búsqueda. La construcción de ambos tipos de horarios de cursada se lleva a cabo mediante la resolución de un modelo de programación lineal entera. Los horarios de cursada no necesariamente aceptables son construídos extrayendo algunas restricciones del modelo utilizado para elaborar a los amisibles, por eso nos enfocaremos en estos últimos.

5.1. Notación

En esta sección se adaptará la notación presentada en la formulación formal del modelo a la formulación de los modelos de programación lineal entera (PLE) que se utilizarán para elaborar horarios de cursada. En el cuadro 3.1, se detalló que la materias en $\mathcal{M}$ no sólo estaban diferenciadas por ser asignaturas distintas, sino también por pertenecer a cursos distintos (por ejemplo, Matemática para segundo año es una materia distinta en $\mathcal{M}$ que Matemática para quinto año). Con el fin de simplificar el planteo del problema de programación lineal entera, se considerará en este contexto que dos materias son distintas sólo si la asignatura lo es; es decir, por ejemplo, Matemática para segundo es la misma materia que Matemática para quinto. Luego, se asigna un número de identificación a cada materia. Se notará M al conjunto de números de identificación de materias.

Otro conjunto que será redefinido en este contexto es el de las horas cátedra. En el cuadro 3.1 se introduce $\mathcal{H}$ como el conjunto de horas cátedra semanales de un curso. En el caso de estudio se tiene

[1] Que no violan restricciones duras ni restricciones duras suavizadas

que $\mathcal{H} = \{0, \ldots, 39\}$, donde 0 es la prehora del Lunes y 39 es la séptima hora del Viernes. Sin embargo, al querer construir horarios admisibles se debe tener en cuenta la disponibilidad del docente y, por lo tanto, este formato de horas cátedra no resulta práctico. Por ende, se considerará el formato de horas cátedra que se utiliza para las fichas de los docentes, el formato de horas en $0 - 79$, que es traducido a lenguaje coloquial mediante la función τ (definición 4.3). Notamos como H al conjunto de horas cátedra en formato $0 - 79$, $H = \{0, \ldots, 79\}$.

Puesto que se desean generar horarios de cursada válidos, se debe tener en cuenta que un docente no puede estar en dos cursos al mismo tiempo (excepto en el caso de las simultaneidades). Se denominará como *conjunto conflictivo* a un conjunto de tuplas de materias y cursos que comparten docentes. Por ejemplo, si la materia $m_1 \in M$ es dictada en el curso 3 por el mismo docente que dicta la materia $m_{12} \in M$ en el curso 8, luego el conjunto conflictivo que representa esta situación es $\{(m_1, 3), (m_{12}, 8)\}$. En otras palabras, cada conjunto conflictivo agrupa a las materias que son dictadas por un mismo docente y los cursos donde las dicta. Por esta razón, para cierta hora cátedra h y conjunto conflictivo K, no puede ocurrir que más de una tupla $(materia, curso) \in K$ sea asignada a h; de lo contrario, un docente estaría siendo asignado a dos cursos al mismo tiempo. Las simultaneidades son consideradas excepciones y, por lo tanto, se incluye en el conjunto conflictivo una sola tupla que representa a los cursos que forman parte de la simultaneidad. Por ejemplo, si m_7 se debe dar simultáneamente en los cursos 6 y 7, se agrega sólo la tupla $(m_7, 6)$ a los grupos de conflictos de los profesores que dan esa materia en esos cursos. Se notará como CK_{TM} al conjunto de conjuntos de conflictos del turno mañana y CK_{TT} al del turno tarde.

Por otro lado, como hay docentes que enseñan durante los dos turnos, se debe evitar que estén dando clases simultáneamente durante la séptima hora del turno mañana y la prehora del turno tarde, pues ambas ocurren durante el mismo periodo de tiempo. Por esta razón, se construye un conjunto de tuplas que agrupa materias que son dictadas por un docente que enseña en ambos turnos siguiendo la siguiente idea: sea CK_{PH} el conjunto en cuestión, supongamos que un mismo docente dicta la materia m_4 en el curso 0 (que corresponde al turno mañana) y a los cursos 12 y 13 (que corresponden al turno tarde); entonces, se agregará a CK_{PH} las tuplas $(m_4, 0, m_4, 12)$ y $(m_4, 0, m_4, 13)$. De esta manera, se podrá indicar con el modelo que si la materia m_4 en el curso 0 es asignada en una séptima hora, entonces en la prehora de ese día no puede asignarse m_4 en los cursos 12 ni 13, o, viceversa, si es asignada en la prehora de los cursos 12 o 13, entonces no puede ser asignada en el curso 0 durante la séptima hora de ese día. Se nota al conjunto de estas tuplas como CK_{PH} y se denominará *conjunto de conflictos por prehora*.

Con respecto a las simultaneidades, se notará cada una como s. La simultaneidad está representada por una tupla cuya primera componente es la materia a dictar y la segunda es el conjunto de cursos donde debe ser dictada simultáneamente. Por ejemplo, si la materia $m \in M$ debe ser dictada simultáneamente en los cursos 6 y 7, entonces $s = (m, \{6, 7\})$ representa este requerimiento. Se nota como S al conjunto de simultaneidades.

Finalmente, si se desea generar un horario amisible, se debe tener en cuenta la disponibilidad de los docentes. Se elabora entonces un diccionario D donde, dada una materia i y un curso c, guarda el conjunto de horas durante las cuales está disponible el o los docentes que dictan i en c. Se nota este conjunto como $D_{(i,c)}$.

Además, se utiliza la construcción de bloques para ciertas materias en ciertos cursos con el fin de introducir variabilidad en la generación de horarios de cursada: mientras más bloques se construyen, más disminuye la penalidad por falta de los mismos (e incluso podría ser menor la penalidad total). Con este propósito, a partir de cada $D_{(i,c)}$ se obtiene $B_{(i,c)}$ el conjunto de horas que pueden encabezar

un bloque de i en c: $B_{(i,c)} = \{h \in D_{(i,c)} \colon (h+1) \in D_{(i,c)} \ \wedge \ h \not\equiv 7 \ (\text{mód } 8)\}$. Por ejemplo, si $D_{(i,c)} = \{5, 6, 7, 8, 17, 18, 20\}$, entonces $B_{(i,c)} = \{5, 6, 17\}$. Para elegir qué materia en qué curso formará bloques, se ordenan las tuplas $(materia, curso)$ en orden ascendente según su carga horaria semanal, excluyendo a las simultaneidades y materias con carga semanal menor que 2, conformando así una lista que se notará como E. Para generar distintos horarios de cursada, se formarán bloques para los primeros σ elementos de E, para distintos valores de σ. Notaremos como $E_{(\sigma)}$ al conjunto de los primeros σ elementos de E.

En la el cuadro 5.1 se resume la notación de los conjuntos y de las constantes utilizados en el modelo de programación lineal entera y algunas características, sean sencillas de describir, en el caso de estudio.

5.2. Modelo de Programación Lineal Entera

Sean $i \in M$, $h \in H$, $c \in C$, $d \in \Delta$, se introducen a continuación las variables de decisión del problema:

$$x_{ihc} = \begin{cases} 1 & \text{si la materia } i \text{ se dicta en la hora } h \text{ en el curso } c \\ 0 & \text{c.c.} \end{cases}$$

$$w_{icd} = \begin{cases} 1 & \text{si la materia } i \text{ tiene 3 horas cátedra en el curso } c \text{ en el día } d \\ 0 & \text{c.c.} \end{cases}$$

$$u_{ihc} = \begin{cases} 1 & \text{si la materia } i \text{ tiene un bloque en el curso } c \text{ que comienza en la hora } h \\ 0 & \text{c.c.} \end{cases}$$

y_{ic}: cantidad de bloques de la materia i que no pudieron ser formados en el curso $c, y_{ic} \in \mathbb{Z}$

Puesto que no se desea asignar materias que no se dan en ciertos cursos, que se prefiere que no hayan más de dos horas diarias de un materia y que se forme la mayor cantidad de bloques para las materias en $E_{(\sigma)}$, la función a minimizar será la siguiente:

$$\sum_{i \in M} \sum_{h \in H} \sum_{c \in C} x_{ikc} + 20 \cdot \sum_{i \in M} \sum_{c \in C} \sum_{d \in \Delta} w_{icd} + 25 \cdot \sum_{(i,c) \in E_{(\sigma)}} y_{ic}$$

A continuación se describen los conjuntos de restricciones a las que están sujetas las variables de decisión.

R1: Los cursos del turno mañana no cursan durante las horas del turno tarde y viceversa:

$$\sum_{i \in M} \sum_{h \in H_{TT}} x_{ihc} = 0 \quad \forall c \in C_{TM} \tag{1}$$

$$\sum_{i \in M} \sum_{h \in H_{TM}} x_{ihc} = 0 \quad \forall c \in C_{TT} \tag{2}$$

R2: Cantidad de horas cátedra máximas y mínimas por día. En el caso de estudio, la cantidad máxima y mínima para el turno mañana es 7 y para el turno tarde 6 y 8 respectivamente:

$$\sum_{i \in M} \sum_{h \in H_d} x_{ihc} \leq 7 \quad \forall c \in C_{TM}, \forall d \in \Delta \tag{3}$$

Notación	Descripción	Característica en el caso de estudio
M	Conjunto de las materias	$M = \{0, \ldots, 29\}$
Δ	Conjunto de días de la semana	$\Delta = \{0, \ldots, 4\}$
H	Conjunto de horas cátedra	$H = \{0, \ldots, 79\}$
H_{TM}	Conjunto de horas cátedra del turno mañana	$H_{TM} = \{h \in H \colon \left\lfloor \frac{h}{8} \right\rfloor \equiv 0 \pmod 2\}$
H_{TT}	Conjunto de horas cátedra del turno tarde	$H_{TT} = \{h \in H \colon \left\lfloor \frac{h}{8} \right\rfloor \equiv 1 \pmod 2\}$
H_d	Conjunto de horas cátedra correspondientes al día $d \in \Delta$	$H_d = \{h \in H \colon h \equiv d \pmod{16}\}$
P	Conjunto de profesores	$P = \{0, \ldots, 49\}$
C	Conjunto de cursos	$C = \{0, \ldots, 21\}$
C_{TM}	Conjunto de cursos del turno mañana	$C_{TM} = \{0, \ldots, 11\}$
C_{TT}	Conjunto de cursos del turno tarde	$C_{TT} = \{12, \ldots, 21\}$
$cgs_{i,c}$	Constante que representa la carga horaria semanal de horas de la materia i en el curso c. Si la materia i no se dicta en el curso c, entonces $cgs_{i,c} = 0$.	
K	Conjunto conflictivo	
CK_{TM}	Conjunto de conjuntos conflictivos del turno mañana	$\lvert CK_{TM} \rvert = 40$
CK_{TT}	Conjunto de conjuntos conflictivos del turno tarde	$\lvert CK_{TT} \rvert = 42$
CK_{PH}	Conjunto de conflictos por prehora	$\lvert CK_{PH} \rvert = 379$
S	Conjunto de simultaneidades	$\lvert S \rvert = 4$
D	Diccionario donde se guarda para cada curso y materia el conjunto de horas durante las cuales los docentes que la imparten están disponibles.	$\lvert D \rvert = 242$
$D_{(i,c)}$	Conjunto de horas en las cuales la materia i puede ser dictada en el curso c	
$B_{(i,c)}$	Conjunto de horas de $D_{(i,c)}$ que pueden encabezar bloques.	
$\beta_{(i,c)}$	Cantidad de bloques que puede formar la materia i en el curso c	$\beta_{(i,c)} = \left\lfloor \frac{cgs_{(i,c)}}{2} \right\rfloor$
E	Lista de tuplas $(materia, curso)$ en orden ascendente según carga horaria semanal	$\lvert E \rvert = 223$
$E_{(\sigma)}$	Primeros σ elementos de E	$0 \leq \sigma \leq 223$

Cuadro 5.1: Notación de los conjuntos y constantes utilizados en el modelo de programación lineal entera

$$\sum_{i \in M} \sum_{h \in H_d} x_{ihc} \geq 7 \quad \forall c \in C_{TM}, \forall d \in \Delta \tag{4}$$

$$\sum_{i \in M} \sum_{h \in H_d} x_{ihc} \leq 8 \quad \forall c \in C_{TT}, \forall d \in \Delta \tag{5}$$

$$\sum_{i \in M} \sum_{h \in H_d} x_{ihc} \geq 6 \quad \forall c \in C_{TT}, \forall d \in \Delta \tag{6}$$

R3: Cantidad máxima de horas cátedra diarias de una materia

$$\sum_{h \in H_d} x_{ihc} \leq 2 + w_{icd} \quad \forall c \in C, \forall i \in M, \forall d \in \Delta \tag{7}$$

R4: Una materia puede tener triple hora a lo sumo una vez en la semana

$$\sum_{d \in \Delta} w_{icd} \leq 1 \quad \forall i \in M, \forall c \in C \tag{8}$$

R5: A lo sumo una materia por hora cátedra

$$\sum_{i \in M} x_{ihc} \leq 1 \quad \forall c \in C, \forall k \in H \tag{9}$$

R6: Se debe satisfacer la carga horaria semanal de la materia y las materias deben ser enseñadas cuando el docente esté disponible

$$\sum_{h \in D_{(i,c)}} x_{ihc} = cgs_{i,c} \quad \forall (i,c) \in \{(i,c) \in M \times C \colon cgs_{i,c} > 0\} \tag{10}$$

R7: si no hay prehora, no se deben dictar materias en la prehora. En el caso de estudio, el turno mañana no tiene prehoras

$$\sum_{h \in \{0,16,32,48,64\}} x_{ihc} = 0 \quad \forall i \in M, \forall c \in C \tag{11}$$

R8: los alumnos no pueden tener horas libres. Para esto, se pide que se asigne una materia a todas las hora entre la primera y la sexta, inclusive. De esa manera, las únicas horas en las que puede no asignarse una materia son la prehora y la séptima

$$\sum_{i \in M} x_{ihc} = 1 \quad \forall c \in C_{TM}, \forall h \in \{h \in H_{TM} \colon h \not\equiv 0 \pmod 8 \wedge h \not\equiv 7 \pmod 8\} \tag{12}$$

$$\sum_{i \in M} x_{ihc} = 1 \quad \forall c \in C_{TT}, \forall h \in \{h \in H_{TT} \colon h \not\equiv 0 \pmod 8 \wedge h \not\equiv 7 \pmod 8\} \tag{13}$$

R9: el turno tarde no tiene séptima hora los lunes

$$\sum_{i \in M} x_{i,15,c} = 0 \quad \forall c \in C_{TT} \tag{14}$$

R10: un docente no puede estar en distintos cursos al mismo tiempo. Para esto utilizaremos los conjuntos de conjuntos de conflictos de ambos turnos y los conflictos por prehora

$$\sum_{(i,c)\in K} x_{ihc} \leq 1 \quad \forall h \in H, \forall K \in CK_{TM} \tag{15}$$

$$\sum_{(i,c)\in K} x_{ihc} \leq 1 \quad \forall h \in H, \forall K \in CK_{TT} \tag{16}$$

$$x_{k_1,h,k_2} + x_{k_3,h+1,k_4} \leq 1 \quad \forall h \in \{h \in H_{TM} :\ h \equiv 7 \pmod 8\}, \forall k = (k_1, k_2, k_3, k_4) \in CK_{PH} \tag{17}$$

R11: se fija la simultaneidad del dictado de materias cuando es requerido. Recordar que cada simultaneidad s venía dada como una tupla (*materia*, *conjunto de cursos*), luego $s_1 \in M$ y $s_2 \subseteq C$

$$x_{s_1,h,s_k} = x_{s_1,h,s_j} \quad \forall h \in H, \forall s_k,\ s_j \in s_2,\ s_k \neq s_j, \forall s \in S \tag{18}$$

R12: se determina la formación de bloques para las simultaneidades. Puesto que este tipo de materias exige coordinar el horario de varios docentes y de varios cursos, resulta ventajoso asignarles un buen horario. De lo contrario, se corre el riesgo de que no formen ningún bloque, y el algoritmo de búsqueda no podrá enmendar esta situación[2]. Dada s una simultaneidad, se tomará como representante del conjunto de cursos de s a $s_r = \min s_2$. Con (19) establece que, si h no puede encabezar un bloque, u_ihc debe valer cero; (20) obliga que se formen todos los bloques posibles; (21) determina que si un bloque empieza en h, otro no puede empezar en $h+1$ (es decir, los triples se cuentan como un solo bloque) y, finalmente, (22) relaciona a las variables u con las variables x.

$$u_{s_1,h,s_r} = 0 \quad \forall h \in H \setminus B_{(s_1,s_r)}, \forall s \in S \tag{19}$$

$$\sum_{h\in B_{(s_1,s_r)}} u_{s_1,h,s_r} = \beta_{(s_1,s_r)} \quad \forall s \in S \tag{20}$$

$$u_{s_1,h,s_r} + u_{s_1,h+1,s_r} \leq 1 \quad \forall h \in B_{(s_1,s_r)}, \forall s \in S \tag{21}$$

$$x_{s_1,h,s_r} + x_{s_1,h+1,s_r} \geq 2 - (1 - u_{s_1,k,s_r}) \cdot 30 \quad \forall h \in B_{(s_1,s_r)}, \forall s \in S \tag{22}$$

R13: se determina la formación de bloques para las primeras σ tuplas de E. Si $\sigma = 0$, no se priorizará la creación de bloques. Las condiciones son análogas a $(19-22)$, sólo que en este caso no se fuerza la creación de bloques. Un conjunto (*materia,curso*) podría no formar bloques, pero eso incurre en que $y_{ic} > 0$ y, por ende, mayor valor de la función objetivo.

$$u_{ihc} = 0 \quad \forall h \in H \setminus B_{(i,c)}, \forall (i,c) \in E_{(\sigma)} \tag{23}$$

$$\sum_{h\in B_{(i,c)}} u_{ihc} = \beta_{(i,c)} - y_{(ic)} \quad \forall (i,c) \in E_{(\sigma)} \tag{24}$$

$$y_{ic} \leq \beta_{(i,c)} \quad \forall (i,c) \in E_{(\sigma)} \tag{25}$$

$$u_{ihc} + u_{i,h+1,c} \leq 1 \quad \forall h \in B_{(i,c)}, \forall (i,c) \in E_{(\sigma)} \tag{26}$$

$$x_{ihc} + x_{i,h+1,c} \geq 2 - (1 - u_{ihc}) \cdot 30 \quad \forall h \in B_{(i,c)}, \forall (i,c) \in E_{(\sigma)} \tag{27}$$

[2]Como se verá en el capítulo siguiente, el algoritmo de búsqueda realiza cambios dentro del horario de cursada de un curso por vez. Por esta razón, si mueve una hora correspondiente a una simultaneidad, estaría incurriendo en la violación de una restricción dura. Luego, nunca moverá las horas de simultaneidades.

σ	Tiempo de solución	Tiempo de poblamiento	Tiempo total	Optimalidad
0	0.96	2.85	3.81	*
20	1.43	6.18	7.61	*
60	1.82	13.16	14.98	*
130	22.27	174.73	197.01	*
160	70.10	269.13	339.24	*
170	5836.94	385.05[(1)]	6221.99	*
200	-	241.55	14641.58	**8.5
223	-	13916.1	28316.12	**18.5

* indica que la solución óptima del problema fue hallada
**n indica que se agotó el tiempo límite de resolución (cuatro horas) y la solución hallada hasta ese momento tenía un *gap* de optimalidad del n %
(1) El solver no pudo poblar el pool de soluciones con diez, sólo con dos

Cuadro 5.2: Tiempo de resolución del modelo para distintos valores de σ

La formulación completa del problema se puede apreciar en el Apéndice I, página 73. El modelo utilizado para la creación de un horario de cursada válido (no necesariamente aceptable) se obtiene relajando este modelo. No se consideran los conjuntos de restricciones R3 y R4 (por lo que las variables w_{icd} no forman parte del modelo), R6 (sólo se pide que la carga horaria semanal de la materia sea satisfecha), R9 y R13.

En este trabajo, el modelo para horarios de cursada admisibles se corrió con en una computadora equipada con el siguiente hardware: procesador Intel Core i5 7400,3.5 GHZ, 8-GB RAM. El modelo fue implementado y resuelto utilizando la API para Python del *solver* CPLEX 12.8.1.0 . Se generaron diez horarios de cursada para cada uno de los valores de σ entre 0 y 170 , en intervalos de 10, y diez horarios de cursada iniciales para $\sigma = 200$ y para $\sigma = 223$ (aunque, como se observa en el cuadro 5.2, en estos casos no se llegó a la solución óptima), generándose en total 192 horarios de cursada iniciales. Para poblar el *pool* de soluciones para cada valor de σ, se permite que estas soluciones tengan un *gap* de optimalidad del 5 % con respecto a la mejor solución hallada. En el cuadro 5.2 se muestran el tiempo (en segundos) de resolución y de poblamiento del *pool* de soluciones para distintos σ.

Notar que el tiempo de resolución aumenta considerablemente cuando σ es mayor o igual que 170. Un hipótesis que explicaría este comportamiento se basa en que la cantidad de materias que deben formar bloque por curso comienza a crecer de tal manera que resulta más difícil hallar soluciones que maximicen bloques para todas ellas. En $E_{(170)}$ se encuentran todas las materias que deben formar un bloque y el 53 % de las materias que deben formar dos; por otro lado, $E_{(170)}$ abarca al 76 % de los pares (*materia*, *curso*) de la escuela. No es extraño que el problema se complique conforme σ se acerca a 223; si se podría resolver el problema para $\sigma = 223$ se obtendría un horario de cursada que minimizaría la cantidad de triples y maximizaría la cantidad de bloques para cada materia en cada curso. Sin embargo, incluso en tal escenario, no se podría asegurar que es un horario óptimo, dado que en el modelo no se han considerado muchas de las restricciones blandas. Por tal motivo, en el desarrollo de este trabajo se optó por limitar el tiempo que el programa dedica a la generación de horarios de cursada admisibles y permitirle consumir más tiempo mejorando los horarios de cursada obtenidos. Por este mismo motivo se preferirá que las soluciones obtenidas con los algoritmos de búsqueda local hayan partido de un horario generado con $\sigma < 200$.

Resolver el problema de la elaboración de un horario de cursada utilizando exclusivamente un modelo de PLE implica, entre otras cosas, resolver el modelo para $\sigma = 223$ puesto que se desea maximizar la cantidad de bloques de las materias, lo cual consume más de cuatro horas. Recordar

que en este modelo no se consideran muchas de las demás restricciones blandas, como la cantidad de prehoras y las horas inactivas de los docentes. Si se añadieran, quedaría conformado un modelo con una cantidad mucho mayor de restricciones y, por ende, el tiempo requerido para obtener la solución no podría satisfacer uno de los objetivos de este trabajo: elaborar horarios en plazos de tiempo relativamente cortos. Por otro lado, como ya se ha mencionado, se desean encontrar horarios de calidad, mas no necesariamente óptimos.

Por otro lado, se generaron también veintiseis horarios de cursada válidos con el modelo relajado (el máximo de solucones con la que el *solver* puedo poblar el *pool*). Debido a la menor cantidad de variables y de restricciones, el tiempo requerido para hallar soluciones óptimas es notablemente menor. Con el hardware y el *solver* mencionados anteriormente, la resolución óptima del problema relajado tomó 0,22 segundos y la población del *pool* de soluciones tomó 0,35 segundos.

Una vez generados los horarios de cursada iniciales, es momento de aplicar a cada uno de ellos un algoritmo de búsqueda local que permita mejorarlos en base a la función de penalidad. En el siguiente capítulo se introducen algunos algoritmos y se exponen resultados de sus respectivos desempeños en el caso de estudio.

Capítulo 6

Algoritmos de búsqueda local

En numerosos problemas, como es el caso de muchos que se clasifican dentro del *scheduling*, el camino hacia el óptimo no es relevante. Los algoritmos de búsqueda local operan utilizando un estado actual y generalmente se mueven sólo a vecinos de ese estado. En la mayoría de los casos el camino transitado no se guarda, lo cual permite que esta clase de algoritmos utilice muy poca memoria [12]. Otra ventaja es que usualmente permiten hallar soluciones razonables en un espacio de búsqueda muy extenso, donde los algoritmos exactos resultan inapropiados.

Para entender mejor los algoritmos de búsqueda suelen hacerse analogías a paisajes naturales y a sus rasgos topológicos. Un paisaje tiene una "ubicación" (definida por el estado actual) y una "elevación" (definida por el valor de la función de penalidad/costo o de la función objetivo). Si el problema es de minimización, el objetivo es alcanzar el valle más bajo; de lo contrario, si se trata de maximizar una función objetivo, la meta es alcanzar el pico más alto. A partir de un estado inicial, los algoritmos de búsqueda local exploran el paisaje en búsqueda del objetivo correspondiente. En la figura 6.1 se puede apreciar un paisaje unidimensional en el cual la elevación se corresponde al valor de una función que se desea maximizar y sus correspondientes rasgos topológicos. Todos estos conceptos se adaptan fácilmente al problema de minimización, pues puede considerarse a este último como un problema de maximización de la función cambiada de signo.

Para poder aplicar un algoritmo de búsqueda local, es necesario especificar la representación del problema, el objetivo y la función de evaluación [17]. Estos tres elementos son esenciales para manipular las soluciones candidatas a óptimos, definir el propósito del problema y establecer una forma de comparar la calidad de dos soluciones. En los capítulos anteriores, se introdujo a la grilla como forma de representar un horario de cursada, se especificó que el objetivo es minimizar la cantidad de violaciones de las restricciones blandas comunes y cumplir con todas las restricciones duras suavizadas duras; finalmente se definió a la función $\hat{\Psi}$ como la función de evaluación que mapea cada grilla a un número real que representa la calidad de la misma.

A partir de esos tres elementos, se puede definir el problema de búsqueda de la siguiente manera: dado un espacio de búsqueda $\mathcal{S}$ y su subconjunto factible $\mathcal{F} \subseteq \mathcal{S}$, hallar $x \in \mathcal{F}$ tal que $eval(x) \leq eval(y) \quad \forall y \in \mathcal{F}$, en el caso de un problema de minimización. Notar que nuestro problema ya fue definido en estos términos en la sección 4.2. Para ciertos problemas, la tarea de encontrar el máximo (o mínimo) global resulta menos difícil si se considera un espacio relativamente pequeño del espacio de búsqueda total [17], de aquí surge la idea de vecindad.

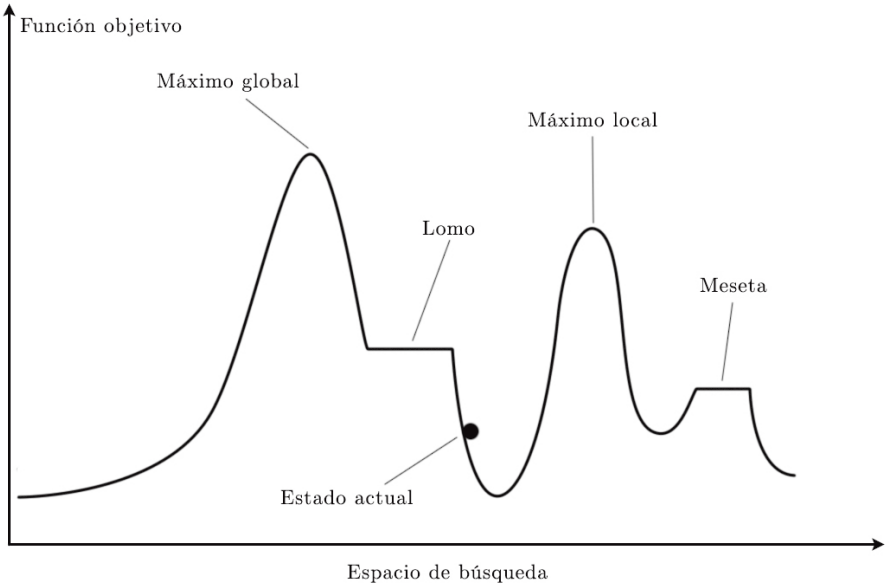

Figura 6.1: Paisaje de un espacio de búsqueda unidimensional

Intuitivamente, dado un punto $x \in \mathcal{S}$, la vecindad $N(x)$ de x puede definirse como un conjunto de elementos de $\mathcal{S}$ que son *cercanos* a x de acuerdo a una noción de distancia. A continuación se define el concepto de vecindad en el marco de la resolución de la Etapa I:

Definición 6.1. (Vecindad) Dos grillas $G_{\hat{\eta},1}$ y $G_{\hat{\eta},2}$ son vecinas si y sólo si son iguales coordenada a coordenada salvo por dos horas cátedra en un mismo curso:

$$G_{\hat{\eta},1}^{(i,j)} = G_{\hat{\eta},2}^{(i,j)} \quad \forall (i,j)/(i,j) \notin \{(i_0,j_0),(i_1,j_0)\}$$
$$G_{\hat{\eta},1}^{(i_0,j_0)} = G_{\hat{\eta},2}^{(i_1,j_0)}$$
$$G_{\hat{\eta},1}^{(i_1,j_0)} = G_{\hat{\eta},2}^{(i_0,j_0)}$$

Es decir, se puede obtener una grilla vecina de $G_{\hat{\eta}}$ intercambiando de lugar dos horas cátedra de un mismo curso.

Una vez definida la noción de vecindad en el marco del problema, se puede introducir el concepto de óptimo local: una solución $x \in \mathcal{F}$ es un óptimo local con respecto al vecindario $N(x)$ si y sólo si $eval(x) \leq eval(y)\ \forall y \in N(x)$ (en el caso del problema de minimización). Este concepto es relevante para el problema que intenta resolver este trabajo, pues, como se ha mencionado antes, el objetivo no es hallar un mínimo global, sino hallar varios mínimos locales y quedarse con el menor de ellos.

Lo métodos de búsqueda local presentan un *trade-off* entre el tamaño del vecindario y la eficiencia de la búsqueda [17]. Si el tamaño del vecindario es relativamente pequeño, el algoritmo podrá recorrerlo rápidamente; sin embargo, aumentan las posibilidades de quedar atrapado en un óptimo local. En contraposición, si se toma un vecindario de gran tamaño, se amplía el abanico de elecciones, lo cual conlleva a una mejor toma de decisiones en el proceso de búsqueda; no obstante, el costo de recorrer un vecindario vasto es mayor tiempo de cómputo. En el problema que se analiza en este trabajo, el cardinal

del espacio de búsqueda es $|\mathcal{S}| \approx 9{,}91 \times 10^{774}$ y dado que el cardinal del vecindario es $|N(G_{\hat{\eta}})| = 15926$ (teniendo en cuenta horarios válidos e inválidos), resulta que, en proporción, el tamaño del vecindario es muy pequeño. En el Apéndice II se describe cómo se calculó cada cardinal. Dado que este trabajo no tiene como objetivo hallar un óptimo global y se desea que el algoritmo sea relativamente rápido, la definición de vecindad elegida resulta adecuada.

6.1. Método de *hill-climbing*

El procedimiento de *hill-climbing*[1] se encuentra entre los más básicos de los métodos de búsqueda local. Como muchos de ellos, emplea una técnica de mejoramiento iterativa sobre el estado actual, examinando a sus vecinos y desplazándose hacia el que tenga un mejor valor de la función de evaluación[2]. Dado que *hill-climbing* pasa a un vecino sin pensar a futuro, también es conocido como el algoritmo de búsqueda local *greedy*. El método termina si no existen vecinos que mejoren el valor de la función o si se alcanza el límite de iteraciones o de tiempo. Es relevante remarcar que el algoritmo no guarda un árbol de búsqueda, por lo que el camino desde el punto inicial no es recordado. En otras palabras, "*hill-climbing se asemeja a tratar de encontrar el pico del Monte Everest en medio de una neblina espesa y padeciendo amnesia*" [12].

Existen varias versiones de algoritmos de *hill-climbing*, y en este trabajo se aplicarán algunas de ellas. En el Algoritmo 1 se muestra la más básica de las versiones, que es también la más *greedy*, aplicada a un problema de minimización. Dado el estado actual v_c, se compara el valor de la función de evaluación con el de los elementos de su vecindario. Al encontrarse con el primer vecino v_n tal que $eval(v_n)$ resulta mejor que $eval(v_c)$, entonces v_n pasa a ser el estado actual. De lo contrario, si se recorre todo el vecindario de v_c sin encontrar a alguien que mejore $eval(v_c)$, el algoritmo ha alcanzado un óptimo local o global.

Lo atractivo del algoritmo de *hill-climbing* es la sencillez de su implementación. No obstante, esa sencillez también es la causa de una serie de desventajas. Resulta evidente que este tipo de método está destinado a converger a un óptimo local y que este depende del punto de partida. Además,generalmente no es posible calcular el error relativo con respecto al óptimo global, puesto que su valor es desconocido. En vista de que no se puede garantizar que el método converja al óptimo local, una práctica frecuente es iniciar *hill-climbing* desde una variedad de puntos iniciales, metaheurística conocida como *restart hill-climbing*, con la esperanza de que alguno de ellos siga un camino que lo guíe hasta el óptimo global[3] [17]. La elección de los puntos iniciales puede ser al azar o teniendo en cuenta algún tipo de información sobre la estructura del problema. En nuestro caso, por ejemplo, se construyeron los puntos iniciales minimizando progresivamente uno de los sumandos de la función de penalidad y, posiblemente, ubicándolos más próximos a mínimos locales.

Además, *hill-climbing* también presenta desventajas relacionadas al *paisaje* del espacio de búsqueda. Ya se ha mencionado que este algoritmo tiende a estancarse en los óptimos locales, lo cual se traduce, en el problema de maximización, como que alcanza un pico que es más alto que cualquiera de los

[1] El término *hill-climbing* hace alusión a un problema de maximización. Sin embargo es muy sencillo aplicarlo a un problema de minimización, pues basta con invertir un signo de desigualdad. Por simplicidad, se utilizará el término hill-climbing para referirse al método empleado para resolver los dos tipos de problemas, sin pérdida de generalidad.

[2] En el contexto de un problema de maximización, sería un mayor valor de la función de evaluación; en un problema de minimización, sería un menor valor de la función de evaluación.

[3] Este procedimiento se asemeja ala segunda fase de la metaheurística conocida como GRASP. Sin embargo, el proceso de construcción de las soluciones iniciales utilizado en este trabajo es diferente al empleado en la primera fase de GRASP, por lo que este procedimiento no encaja completamente en la metaheurística mencionada.

Algoritmo 1 Hill-climbing

Input: v_c punto inicial en $\mathcal{S}$, max_iter cantidad máxima de iteraciones

```
1: procedure HILL-CLIMBING(v_c, max_iter)
2:     iter ← 0
3:     mejor ← eval(v_c)
4:     N ← vecindario de v_c
5:     local ← FALSE
6:     while iter ≤ max_iter and local =FALSE do
7:         if N ≠ ∅ then
8:             v_n ← vecino de v_c
9:             if eval(v_n) < eval(v_c) then
10:                mejor ← eval(v_n)
11:                v_c ← v_n
12:                N ← vecindario de v_c
13:                iter ← iter + 1
14:            else
15:                N ← N − {v_n}
16:            end if
17:        else
18:            local ←TRUE
19:        end if
20:    end while
21: end procedure
```

estados vecinos y se detiene allí, pero no es un óptimo global. Otro aspecto topológico del espacio de búsqueda que genera complicaciones al *hill-climbing* son las crestas o cordilleras: una secuencia de máximos locales que no están conectados directamente entre ellos; puesto que desde cada máximo local todas las posibles acciones empeoran la función de evaluación, el algoritmo se detiene. Finalmente, las mesetas también causan problemas al algoritmo, puesto que se tratan de regiones donde la función de evaluación vale lo mismo para todos los vecinos; es posible incluso que se trate de un lomo, y que fuese posible seguir ascendiendo.

Es claro que el éxito de *hill-climbing* depende mucho de la cantidad óptimos locales y de mesetas que presenta el paisaje del espacio de búsqueda. Generalmente, los problemas NP-Hard tienen una cantidad de óptimos locales exponencial, pero un óptimo local razonablemente bueno puede encontrarse con un pequeño número de corridas del *hill-climbing* [12]. Más aún, se ha probado que es mejor, en algunos casos, reiniciar el algoritmo de búsqueda local en nuevos puntos luego de una cantidad de tiempo fija y que esto es más eficiente que permitir que cada búsqueda continúe indefinidamente [12].

Es en base a las desventajas que a lo largo del tiempo han surgido numerosas variantes del algoritmo, como *steepest-ascent hill-climbing* y *stochastic hill-climbing*, y se han desarrollado otros métodos más complejos en base al mismo (como búsqueda tabú, *simulated annealing* o los algoritmos genéticos). Sin embargo, estos métodos más sofisticados tienen como objetivo hallar el óptimo global y su complejidad implica más tiempo de cómputo. Teniendo en cuenta que el objetivo de este trabajo no es hallar un óptimo global, se consideró que la mejora potencial que ofrecen los métodos más sofisticados no alcanza para compensar el tiempo de cómputo y la complejidad de implementación. Por tanto, en este trabajo se estudiará el resultado de aplicar *hill-climbing* y algunas de sus variantes al problema de HSSP que se desea resolver, intentando sortear las dificultades que presenta este método.

6.2. Consideraciones previas

La primera aclaración que se hará con respecto a la implementación es la numeración de los vecinos de una grilla G. Dada tal grilla, se enumeran los posibles intercambios de horas que pueden llevarse a cabo en cada curso. Como el intercambio de horas es simétrico (es lo mismo cambiar la hora 4 por la 2 que la 2 por la 4), siempre la primera hora del intercambio será menor que la segunda. Así, en un primer momento, los intercambios posibles en un curso j vienen dados por:

$$\mathcal{E}_j = \{(h_1, h_2, j) \in \mathcal{H} \times \mathcal{H} \times \mathcal{C} \colon \ h_2 > h_1\}$$

donde (h_1, h_2, j) significa que se intercambiarán h_1 y h_2 en el curso j. Los elementos de $\mathcal{E}_j$ se ordenan de la siguiente manera: sean $s_1 = (h_1, h_2, j)$, $s_2 = (h_3, h_4, j)$, $s_1, s_2 \in \mathcal{E}_j, s_1 \neq s_2$, luego s_1 tiene una numeración que precede a la de s_2 si $h_1 < h_3 \vee (h_1 = h_3 \wedge h_2 < h_4)$; de lo contrario, la numeración de s_2 precede a la de s_1. De esta manera, se ordena al conjunto $\mathcal{E}_j$. Este procedimiento se realiza para todos los cursos y, salvo que se indique lo contrario, el orden del conjunto $\mathcal{E}$ de los posibles intercambios queda establecido de la siguiente manera: $s_1 = (h_1, h_2, j)$, $s_2 = (h_3, h_4, k)$, $s_1, s_2 \in \mathcal{E}$, $s_1 \neq s_2$, la numeración de s_1 precede a la de s_2 si $j < k$ o si $j = k \wedge h_1 < h_3$ o si $j = k \wedge h_1 = h_3 \wedge h_2 < h_4$; de lo contrario, s_2 precede a s_1. En términos menos técnicos, a partir de G se comienza analizando los posibles intercambio del curso 0, luego los del curso 1 y así sucesivamente.

Uno de los problemas del *hill-climbing* es que, como muchos algoritmos *greedy*, al trasladarse de un punto a un vecino no tiene en cuenta movimientos futuros. Recordar que teníamos que dos grillas son vecinas si y sólo si son iguales salvo por dos horas cátedra de un mismo curso, que han intercambiado lugares.

Consideremos el ejemplo que se ilustra a continuación, en la figura 6.2. Sea G una grilla y G' una grilla vecina, que se obtiene intercambiando de lugar las horas 1 y 10 del curso j. Como puede observarse, con ese intercambio se destruyen dos bloques: el de Matemática y el de Lengua. Dado que el armado de bloques tiene una prioridad relativamente alta entre las restricciones blandas, es probable que el algoritmo tienda a evitar desarmarlos. En consecuencia, es posible que decida no moverse a G'. Sin embargo, resultaría interesante explorar G'', el vecino de G' que se obtiene intercambiando las horas 2 y 11 del horario del curso j en G':

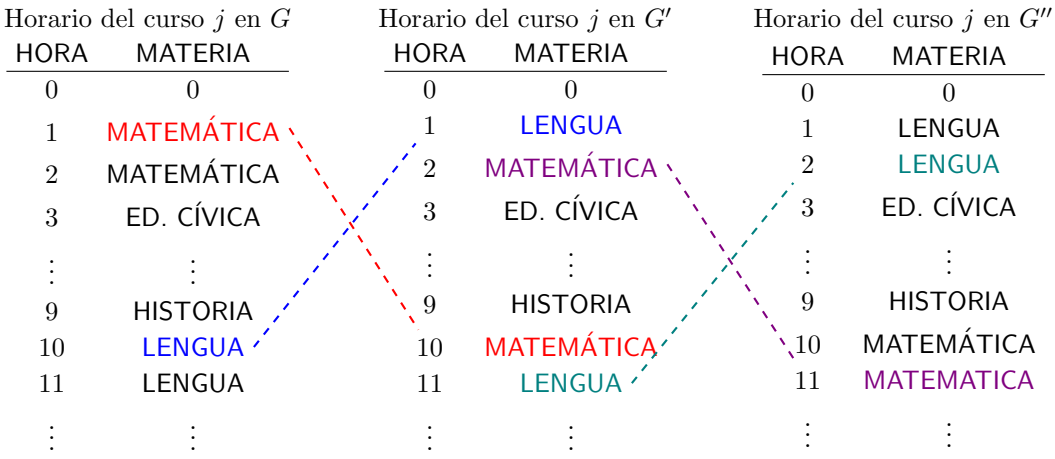

Figura 6.2

Lo interesante de G'' es que se mantienen los bloques que existían en G, pero han intercambiado lugares. Esto permite analizar una nueva situación que podría ser favorable para los docentes (por ejemplo, porque permite compactar sus horarios) o para los alumnos (podrían eliminarse casos de triple hora de alguna materia). Sin embargo, podría ocurrir que nunca se llegue a G'', puesto que el algoritmo decidió que pasar a G' no era un movimiento beneficioso.

Para evitar este tipo de situaciones se decidió lo siguiente: si una hora que encabeza un bloque (en el ejemplo, la hora 1) va a intercambiar materias con otra que encabeza un bloque (en el ejemplo, la hora 10), entonces se realiza el intercambio de ambos bloques. Es decir, se evalúa G'' en vez de G' y si $\hat{\Psi}(G'') < \hat{\Psi}(G)$, G'' será el nuevo estado actual. En lo que sigue del trabajo se hará un abuso de la definición de vecindad y se incluirá a G'' cuando se mencionen a los vecinos de G.

Sin embargo, generalizar este procedimiento sin más contemplaciones tiene sus desventajas. En la figura 6.3 se propone otro ejemplo. Supongamos que se desean intercambiar las horas 2 y 10 del curso j en la grilla G. Como ambas encabezan bloques, de acuerdo a lo explicado anteriormente, deberían directamente intercambiarse los bloques, dando lugar a la grilla G'. No obstante, no sólo el nuevo intercambio produjo dos *triples* sino que también previno la reestructuración de los bloques: si efectivamente se hubiesen intercambiado sólo las horas 2 y 10, las materias seguirían formando la misma cantidad de bloques pero en una nueva configuración, como ocurre en la grilla $\bar{G}'$.

Horario del curso j en G

HORA	MATERIA
0	0
1	GEOGRAFÍA
2	MATEMÁTICA
3	MATEMÁTICA
⋮	⋮
9	MATEMÁTICA
10	GEOGRAFÍA
11	GEOGRAFÍA
⋮	⋮

Horario del curso j en G'

HORA	MATERIA
0	0
1	GEOGRAFÍA
2	GEOGRAFÍA
3	GEOGRAFÍA
⋮	⋮
9	MATEMÁTICA
10	MATEMÁTICA
11	MATEMÁTICA
⋮	⋮

Horario del curso j en $\bar{G}'$

HORA	MATERIA
0	0
1	GEOGRAFÍA
2	GEOGRAFÍA
3	MATEMÁTICA
⋮	⋮
9	MATEMÁTICA
10	MATEMÁTICA
11	GEOGRAFÍA
⋮	⋮

Figura 6.3

Por esta razón, se creyó conveniente imponer otra condición para efectuar un intercambio de bloques: si se van a intercambiar $h_1, h_2 \in \mathcal{H}$, no sólo h_1 y h_2 deben encabezar bloques, sino que también debe ocurrir que la materia en $h_1 - 1$ sea distinta a la que se encuentra en h_2 y lo análogo para $h_2 - 1$. Esto motiva la definición de la función `bloqFlag` (ver Algoritmo 2) que, dadas dos horas a intercambiar, determina si se efectuará un intercambio común o un intercambio de bloques. De aquí en adelante, se denominará *swap* al intercambio de dos horas o dos bloques.

Es claro que podrían tenerse en cuenta más escenarios que los mostrados en los ejemplos y quizás imponer más condiciones o modificaciones a la manera en la que el algoritmo se desplaza de una grilla a otra; pero, por otro lado, se desea restringir la libertad de exploración del espacio de búsqueda lo menos posible y preservar la sencillez característica del *hill-climbing*. Empíricamente se observó que considerar swaps de bloques mejora el resultado obtenido con el algoritmo, por lo que se decidió

Algoritmo 2 Función que determina el tipo de intercambio de horas

Input: G grilla, c curso, h_1, h_2 horas cátedra, B conjunto de horas cabeza de bloque en c
Output: $bloq\_flag$ indicador del tipo de intercambio

```
1: function bloqFlag(G, c, B, h_1, h_2)
2:     bloq_flag ← FALSE
3:     if h_1 ∈ B and h_2 ∈ B then
4:         if G^(h_1,c) ≠ G^(h_2−1,c) and G^(h_2,c) ≠ G^(h_1−1,c) then
5:             bloq_flag ← TRUE
6:         end if
7:     end if
8:     return bloq_flag
9: end function
```

preservar el criterio.

También se añadieron algunas mejoras al algoritmo básico para disminuir el tiempo de cómputo. Si bien la grilla es representada por una matriz de 40×22 (en el caso de estudio), es costoso generar a un vecino G' copiando la grilla G y luego efectuar el swap sobre G' y evaluar $\hat{\Psi}$, puesto que este procedimiento debe realizarse miles de veces por iteración. En consecuencia, se decidió implementar el swap de horas *in situ*: en la misma grilla G se efectúa el swap, dando a lugar a la grilla G'; si G' no tiene una penalidad menor que G, el swap se revierte, volviendo al estado original. Por otro lado, como pasar de una grilla a una vecina implica unos pocos cambios locales, es decir, se intercambian dos (o a lo sumo cuatro) horas de un sólo curso, no es eficiente recalcular la penalidad total de la misma. Por ejemplo, si los docentes afectados por el swap son p_1 y p_3, ¿por qué recalcular la penalidad por horas inactivas del resto de los docentes? Con esto en mente, se implementaron los diccionarios de penalidad: un diccionario para cada elemento sujeto a restricciones (docentes, materias, recursos, prehoras, etc.) que guarda la información de la penalidad en la que incurre cada uno de ellos; luego, sólo es necesario actualizar la información en los diccionarios de penalidad para los sujetos que se vieron afectados por el swap. Además, se preserva una *discriminación de penalidad*, una lista que guarda la penalidad de cada sumando de $\hat{\Psi}$, que será esencial para determinar si un horario es aceptable.

Asimismo, para el problema en cuestión, no todas las grillas del espacio de búsqueda resultan soluciones válidas. Así pues, se resolvió no visitar las grillas vecinas que resultaran inválidas. Recordar que las grillas iniciales se confeccionaron respetando todas las restricciones duras y por lo tanto todas resultan válidas. El cumplimiento de muchas de esas restricciones está garantizado (por ejemplo, los cargos de los docentes y la cantidad de horas diarias de cursada) mientras que el respeto de otras (por ejemplo, que un curso no curse dos materias a la vez) se mantiene debido a que pasar de una grilla a su vecina sólo implica un swap de horas. Sin embargo, hay dos restricciones duras que pueden violarse: que los alumnos no tengan horas libres y que los docentes no estén en dos cursos al mismo tiempo (salvo simultaneidades).

Para no incurrir en la transgresión de la primera, se restringe el conjunto de posibles swaps para cada curso, es decir, se restringe $\mathcal{E}_j$. En el caso de estudio, para los cursos del turno mañana basta sacar de $\mathcal{E}_j$ a los swaps que incumban prehoras[4]. Para los cursos del turno tarde se impide que las horas en las que no hayan clase intercambien lugar con horas entre la primera y la sexta. También se excluye el intercambio entre dos horas libres, pues no tiene ningún impacto en la penalidad de la grilla. En la figura 6.4 se ilustra un ejemplo. Para cada curso, el conjunto de swaps que no provocan

[4]Puesto que, excluyendo a las prehoras, la cantidad de horas cátedra restantes es igual a la cantidad de horas de clase que deben ser asignadas, no es posible que ocurran horas libres

la infracción de esta restricción será denominado en el pseudocódigo como *swaps_validos*.

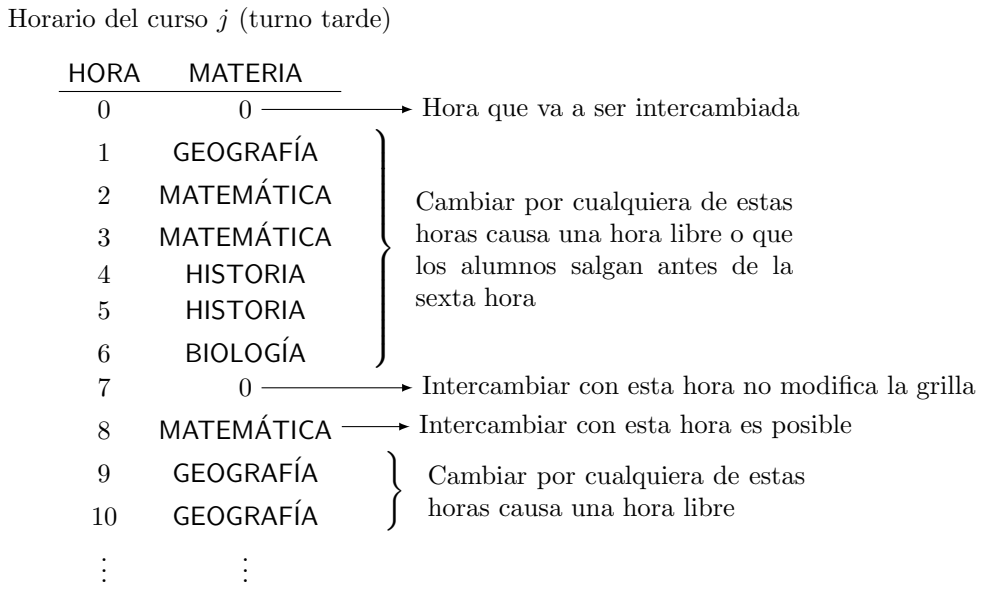

Figura 6.4: Ejemplo de restricción de posibilidades de intercambio para las horas en las que no se asignan materias

Evitar asignar a un docente a dos cursos durante la misma hora cátedra requiere analizar su ficha una vez efectuado el swap. Para esto se implementa la función *validar*. que, una vez efectuado el swap, examina cómo afectó a los docentes implicados. Si alguno de ellos estuviese siendo asignado a dos cursos simultáneamente, la grilla resulta inadmisible y se revierten los cambios realizados. Obviamente, se tiene en cuenta la excepción por simultaneidades.

En síntesis, comenzando en una grilla G, se inicializan su conjunto de fichas F, sus diccionarios de penalidad D y su discriminación de penalidad P. Cada swap que se evalúa implica una modificación parcial de todos ellos; si la grilla resultante del swap es inválida o no es mejor, se revierten los cambios.

Finalmente se implementó la función `negarCambio` que, dadas dos horas cátedra, indica si no vale la pena evaluar ese swap. Esto es útil si sucede alguno de los siguientes acontecimientos: realizar el swap implica violar una simultaneidad, la materia que hay en las dos horas es la misma o, en los casos en los que exista, la grilla resultante de efectuar el swap se encuentra en la lista tabú. Ahora, sin más preámbulos, se mostrará la implementación y los resultados de aplicar variantes de *hill-climbing* al problema que nos ocupa.

6.3. *Hill-climbing* simple

Se denominará *hill-climbing* simple a la variante más sencilla de *hill-climbing*, introducida en la sección 6.1. En el Algoritmo 3 (Apéndice III, página 80) se presenta su implementación para resolver el problema de minimizar la penalidad de una grilla. El input *datos* representa a toda la información necesaria para calcular la penalidad de la grilla: disponibilidad de docentes, carga horaria semanal

de cada materia en cada curso, requerimientos de recursos, simultaneidades, etc. Por esa razón, en el pseudocódigo se nota su utilización al momento de verificar la validez de una grilla y al inicializar la penalidad y los diccionarios así como también cuando se actualizan cada uno de ellos.

Naturalmente, esta implementación acarrea muchas de las desventajas que se mencionaron en la sección 6.1. Por empezar, se toma la primera grilla vecina válida que mejore la penalidad sin terminar de recorrer el vecindario, lo cual implica que podrían haber otros movimientos que permitan una mejora superior, pero que no están siendo considerados. Observar también que la manera en la que se recorren los vecinos de cada grilla conlleva a que se mejore el horario de los cursos en orden, ya que, por ejemplo, no se pasa al curso 1 hasta que todas las mejoras posibles hayan sido efectuadas al horario de cursada del curso 0.

Para aplicar este algoritmo a las grillas admisibles iniciales se tomó $iter\_max = 1000$ y se agregó además un indicador para informar si el algoritmo se detuvo porque alcanzó el límite de iteraciones o porque ningún vecino podía mejorar la penalidad (se llegó a un mínimo local o a una meseta). El límite de iteraciones se estableció de manera tal que el algoritmo no tome demasiado tiempo en ejecutarse y siguiendo la filosofía, mencionada al principio del capítulo, de preferir más ejecuciones cortas a partir de varios puntos iniciales que unas pocas ejecuciones largas.

Se ejecutó el algoritmo de forma paralela para las 192 grillas admisibles iniciales. La ejecución se llevó a cabo en la nube, en un servidor de DigitalOcean[5], con un quad-core vCPU. Los datos sobre los tiempos de ejecución se encuentran en el cuadro 6.1. El máximo tiempo requerido corresponde a una grilla generada con $\sigma = 20$ y el mínimo a una grilla generada con $\sigma = 223$. En la figura 6.5 se puede observar el tiempo promedio según el valor de σ. Además, para el 100 % de las grillas, el algoritmo se detuvo por no poder continuar mejorando. A partir de estos datos se puede corroborar una hipótesis planteada al momento de armar las grillas: mayores σ colocan a los puntos iniciales más cerca de mínimos locales o de mesetas.

	Tiempo (h:mm:ss)
Promedio	0: 16: 32
Desvío estándar	0: 07: 21
Máximo	0: 31: 52
Mínimo	0: 01: 57

Cuadro 6.1: Tiempos de ejecución del algoritmo

Luego de haberles aplicado el algoritmo, el 21,4 % (41) de las grillas iniciales logró mejorar la penalidad de la grilla que confeccionaron las autoridades de la escuela para el año 2017. En la figura 6.6 se observa el boxplot de la penalidad total de las grillas luego de haberles aplicado el algoritmo, considerando al conjunto de todas las grillas iniciales admisibles y en la figura 6.7 se muestra un boxplot del conjunto del mismo dato, pero con las grillas iniciales admisibles separadas según el σ que se utilizó para generarlas; se grafican allí también los promedios de cada grupo. En ambos gráficos, la línea roja punteada representa al valor de penalidad de la grilla elaborada manualmente.

De las 41 grillas que lograron mejorar a la elaborada manualmente (de aquí en adelante referida como *grilla estándar*), 25 resultaron aceptables (es decir, aproximadamente el 13 % de las grillas iniciales). De esas 25 grillas aceptables, una fue generada con $\sigma = 130$, una con $\sigma = 150$, tres con $\sigma = 160$,

[5] `https://www.digitalocean.com/`

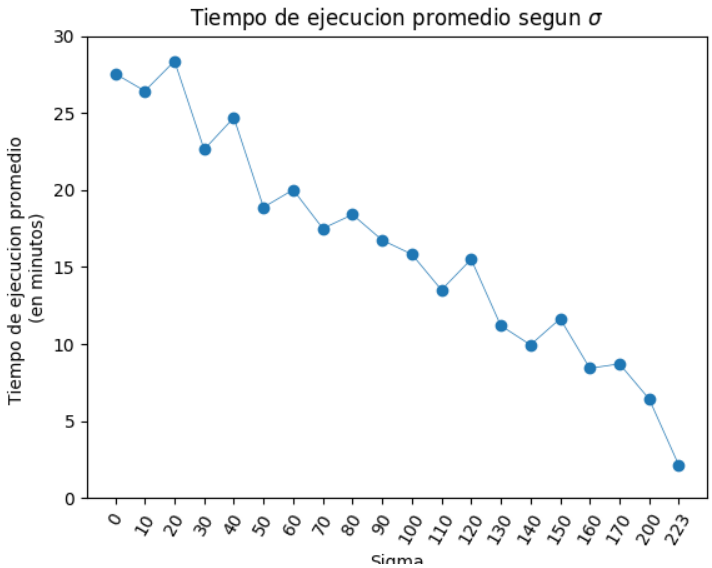

Figura 6.5

diez con $\sigma = 200$ y diez con $\sigma = 223$. A partir de los tiempos de ejecución y de las características del algoritmo, es claro que las grillas creadas con $\sigma = 223$ no sufrieron demasiados cambios. Además, recordar que para elaborar las grillas con $\sigma = 200$ y $\sigma = 223$ se requirió un total de ocho horas; por ende, en términos de tiempo de cómputo, podrían resultar más valiosas las cinco grillas optimizadas obtenidas con los otros valores de σ.

En síntesis, el método contribuyó al hallazgo de 25 grillas que superan a la estándar, de las cuales 5 surgieron a partir de grillas iniciales elaboradas en menos de diez minutos (ver cuadro 6.2 para la mejora de cada una). Es interesante notar también que todas las grillas aceptables que superan a la estándar partieron de grillas generadas con un σ relativamente alto, lo cual indica que es favorable confeccionar puntos iniciales con menor penalidad.

id	σ	Penalidad respecto a grilla estándar
130_003	130	-13.79 %
160_009	160	-18.93 %
150_000	150	-18.59 %
160_002	160	-6.97 %
160_003	160	-10.19 %

Cuadro 6.2: Mejora con respecto a la penalidad de la grilla estándar

A las grillas construidas con el modelo relajado se les aplicó el algoritmo con un límite de iteraciones de 3000, dado que la penalidad inicial era mucho mayor que la de las grillas generadas con el modelo completo. Una vez más, todas las ejecuciones arribaron a un mínimo local o a una meseta. El tiempo de ejecución promedio ronda la hora y cinco minutos. Ninguna de las grillas obtenidas luego de las ejecuciones resultó aceptable y ninguna pudo mejorar la penalidad de la grilla estándar. En la figura 6.8 se grafica el boxplot de la penalidad total de este conjunto de grillas. Si bien la generación de estas

grillas fue instantánea, la notable diferencia de penalidad con la grilla estándar y el largo tiempo de ejecución hacen que no resulten puntos iniciales efectivos.

Es claro que el algoritmo puede ser mejorado en muchos aspectos. Por ejemplo, podría resultar beneficioso recorrer una meseta ya que esta podría tratarse de un lomo; es decir, se podría seguir descendiendo. La siguiente sección se avoca a efectuar esa modificación.

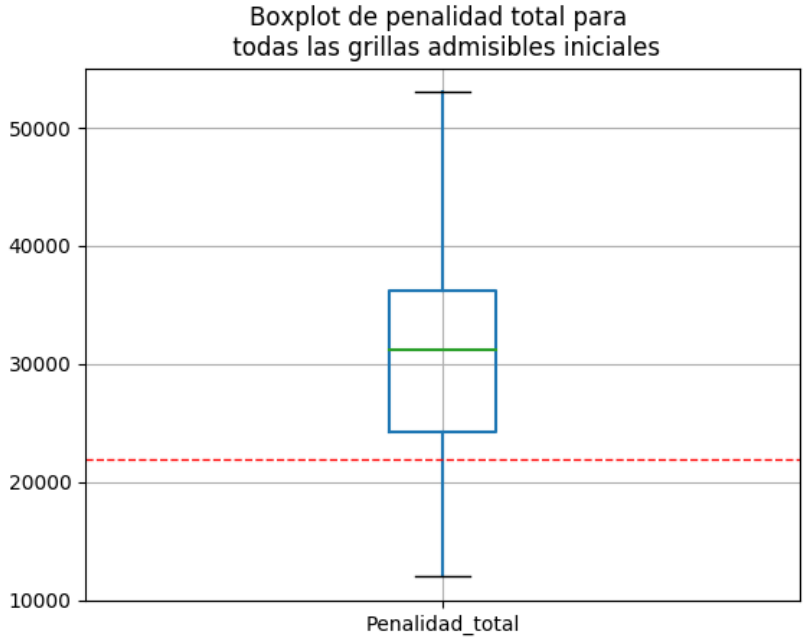

Figura 6.6

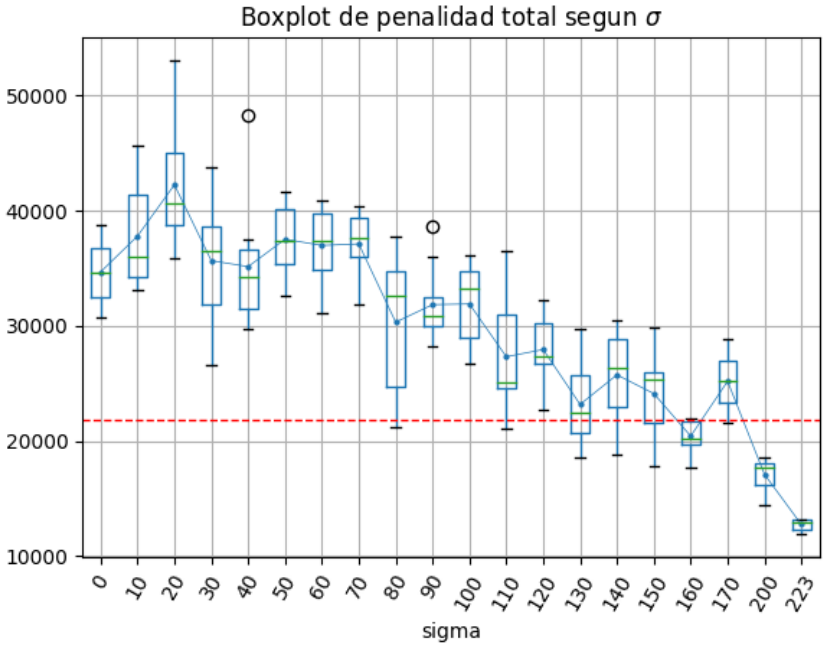

Figura 6.7

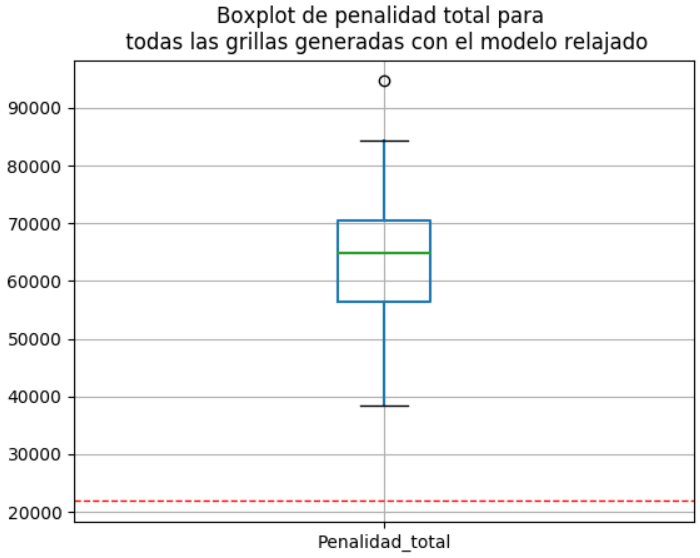

Figura 6.8

6.4. *Hill-climbing* con movimientos laterales y búsqueda tabú

Para afrontar el problema del recorrido de mesetas, suele ser una buena idea permitir movimientos laterales con la esperanza de que la meseta sea realmente un lomo. En otras palabras, se le permite al algoritmo que se desplace a un vecino con igual valor de la función de evaluación. Sin embargo, hay que tomar ciertos recaudos.

Si se permite una cantidad infinita de pasos laterales cuando no hay ninguna dirección en la que se pueda descender, ocurrirá un *loop* infinito cuando el algoritmo alcance una meseta que no es un lomo. La solución más común a este problema es limitar la cantidad de movimientos laterales permitidos [12]. En este trabajo, se decidió establecer 1000 como ese límite.

La manera en que se recorre el vecindario de una grilla es causa de otra dificultad para los movimientos laterales. Supongamos que de una grilla G se pasa a una vecina $\bar{G}$ intercambiando las dos primeras horas del curso 0, ya que $\hat{\Psi}(G) = \hat{\Psi}(\bar{G})$. Luego, ocurrirá que, al momento de recorrer el vecindario de $\bar{G}$, vuelvan a intercambiarse la primera y la segunda hora del curso 0, puesto que la función de evaluación tiene igual valor en G y en $\bar{G}$. Este proceso se repetirá hasta que se agote la cantidad de pasos laterales permitidos. Para evitar esta situación, resulta útil otorgarle *memoria* al algoritmo, introduciendo la búsqueda tabú. Esta variante del *hill-climbing* mantiene una lista tabú con los k últimos estados transitados que no pueden volver a ser visitados.

La gran ventaja que otorga la búsqueda tabú es aumentar las posibilidades de no transitar un camino ya recorrido. Estas posibilidades aumentan con el tamaño de la lista tabú, en detrimento de la memoria que ocupa el programa. Además, mientras más grande sea la lista, mayor es el tiempo requerido para chequear si un elemento se encuentra en ella. En el caso de estudio, el segundo aspecto provocó gran impacto en el tiempo de cómputo del algoritmo y, en consecuencia, se ajustó k empíricamente para balancear las ventajas y desventajas. Finalmente, se optó por tomar $k = 50$.

En el Algoritmo 4 (Apéndice III, página 81) se describe la implementación de este método, que supone unos pocos cambios sobre el Algoritmo 3. Principalmente se añadió la lista lista tabú para recordar las últimas 50 grillas visitadas. Cuando la lista está completa, en el caso de querer agregar una nueva grilla a ella, se descarta el primer elemento. Así, la lista tabú funciona como una cola. Si el algoritmo encuentra un vecino que mejora estrictamente la penalidad, la lista tabú se vacía, puesto que el algoritmo no volverá a ninguna de las grillas que se encontraban listadas pues tienen una penalidad menos favorable. Además, con una mejora estricta se reinicia el contador de pasos laterales pues el algoritmo logró escapar de la meseta. De esta manera, se cuenta una iteración cada vez que la mejora en la penalidad se da por una disminución estricta en la misma. Para este algoritmo se impuso un máximo de 1000 iteraciones. También se añadió otra posibilidad al indicador, ahora también registra si el algoritmo se detuvo por haber agotado la cantidad de pasos laterales permitidos.

El algoritmo se ejecutó de forma paralela para las 192 grillas iniciales admisibles. Como ocurrió con el algoritmo de la sección anterior, todas las ejecuciones se detuvieron antes de alcanzar el máximo de iteraciones, pero esta vez no por haber encontrado un mínimo local sino por haber agotado la cantidad de pasos laterales. Esto puede ser un indicio de que el paisaje del espacio de búsqueda cuenta con numerosas mesetas, que son relativamente extensas. Es posible que ampliando la capacidad de la lista tabú y permitiendo una mayor cantidad de pasos laterales se otorgue un mayor margen para escapar de ellas. No obstante, ambas medidas repercutirían negativamente en el tiempo de cómputo. En el siguiente cuadro se detallan los tiempos de ejecución para las corridas:

	Tiempo (h:mm:ss)
Promedio	0 : 31 : 00
Desvío estándar	0 : 09 : 14
Máximo	0 : 50 : 25
Mínimo	0 : 14 : 06

Cuadro 6.3: Tiempos de ejecución del algoritmo

Como ocurrió anteriormente, el tiempo máximo se dio en la aplicación del algoritmo a una grilla confeccionada con $\sigma = 20$ y el tiempo mínimo en la aplicación a una grilla confeccionada con $\sigma = 223$. El tiempo promedio de ejecución según σ sigue una tendencia muy similar a la observada en la figura 6.5, descendiendo a medida que aumenta σ.

De las 192 grillas iniciales, 42 (aproximadamente el 22 %) lograron mejorar la penalidad de la grilla estándar luego de aplicar el algoritmo. De esas 42, 25 resultaron aceptables. Las proporciones de los σ dentro de estas 25 grillas es similar al caso del *hill-climbing* simple: diez provienen de ser generadas por $\sigma = 223$, diez de $\sigma = 200$, una de $\sigma = 150$, tres de $\sigma = 160$ y una de $\sigma = 170$. Una vez más, las grillas obtenidas con los σ menores que 200 resultan más interesantes pues fueron generadas en poco tiempo. En la figura 6.9 se muestra la comparación entre los dos métodos analizados hasta ahora con respecto a la penalidad total del conjunto de todas las grillas admisibles. Si bien la mediana es similar, es interesante notar la disminución de la variabilidad de los datos, lo cual podría indicar que las grillas mejoraron más con este algoritmo. En efecto, el 66 % de las grillas lograron una mejora superior de la penalidad con el nuevo algoritmo respecto al anterior, el 4 % logró una mejora menor[6] y el 30 % restante la misma mejora.

La discriminación según σ de la penalidad total luego de aplicar el algoritmo es similar a la del caso anterior, por lo que resulta redundante presentar un gráfico parecido al de la figura 6.7. En el cuadro 6.4 se comparan las mejores[7] grillas obtenidas de ambos métodos hasta ahora analizados. Es notable que cuatro de las cinco coincidan para ambos.

id	σ	MP1	MP2
130_003	130	-13.79 %	5.75 %
160_009	160	-18.93 %	-19.92 %
150_000	150	-18.59 %	-18.83 %
160_002	160	-6.97 %	-6.98 %
160_003	160	-10.19 %	-10.21 %
170_000	170	-1.40 %	-4.19 %

MP1 y MP2 se refiere a la mejora respecto a la penalidad de la grilla estándar utilizando el primer y el segundo algoritmo, respectivamente

Cuadro 6.4: Mejora con respecto a la penalidad de la grilla estándar

[6] la diferencia del signo de desigualdad entre los dos algoritmos al momento de evaluar si se traslada o no al vecino de la grilla actual, provoca que el algoritmo recorra distintos caminos. Para este 4 % de las grillas, resultó que el camino más ambicioso era también más favorable.

[7] Por *mejores* se entiende que tienen una penalidad menor que la grilla estándar, que son aceptables y que partieron de una grilla generada con un σ menor que 200

Tal como ocurría para el anterior algoritmo, las grillas generadas a partir del modelo relajado, a pesar de contar con mayor límite de iteraciones, no lograron mejorar la penalidad de la grilla estándar. Al igual que ocurría en el caso de las grillas generadas con el modelo completo, el algoritmo se detuvo siempre por superar la cantidad máxima permitida de pasos laterales. Vuelve a darse que la variabilidad de la penalidad luego de aplicar el algoritmo a las grillas de este grupo es menor que el de aplicar el primer algoritmo. Para este grupo de grillas, el 31 % logró una mejora superior, el 23 % una mejora menor y el 46 % una mejora igual. Puesto que el tiempo de cómputo es ampliamente superior al del empleado para las grillas admisibles, una vez más resulta poco beneficioso iniciar el algoritmo en grillas que no garantizan aceptabilidad.

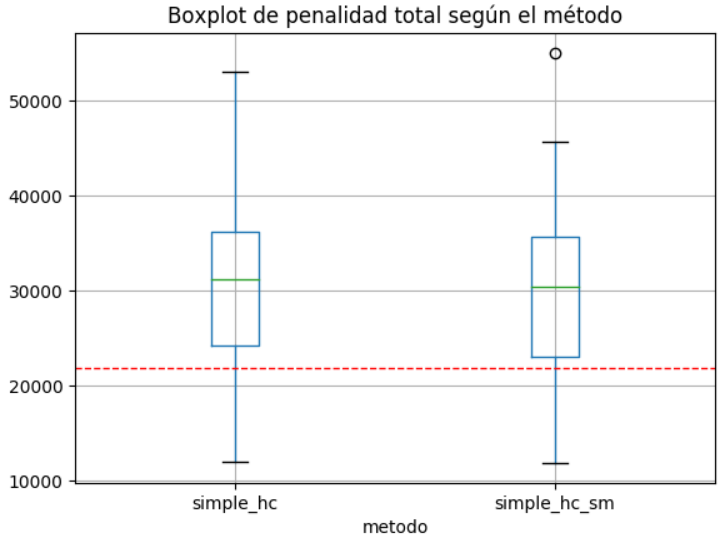

Figura 6.9: A la izquierda, el boxplot del *hill-climbing* simple y, a la derecha, el de *hill-climbing* con movimientos laterales y búsqueda tabú

6.5. *Hill-climbing* híbrido

Hasta el momento, el algoritmo orienta la búsqueda hacia el primer vecino cuya penalidad es menor o igual que la del estado actual. Sin embargo, vale la pena tener en cuenta otro enfoque: evaluar cuánto mejora la situación actual cada uno de los vecinos y efectuar el mejor movimiento posible. En eso consiste la variable de *hill-climbing* conocida como $steepest - ascent\ hill\text{-}climbing$.

En la implementación de este nuevo algoritmo se mantuvieron las mejoras introducidas anteriormente: pasos laterales y búsqueda tabú. Claro está que cada movimiento consumirá una cantidad de tiempo mayor respecto de las variables anteriores de *hill-climbing*. Sin embargo, se cuenta con la ventaja de analizar completamente el vecindario antes de decidir hacia donde moverse. En el Algoritmo 5 (Apéndice III. página 83) se muestra la implementación de este método. En favor de no extender demasiado este apartado del trabajo, no se presentarán resultados detallados de la aplicación de este algoritmo dada su pobre eficacia tanto en tiempo de cómputo como en mejora de la penalidad. Sin embargo, se incluye el boxplot con los resultados obtenidos de su aplicación a las grillas iniciales

admisibles en las siguientes comparaciones de métodos.

A partir de la idea que contribuye *steepest-ascent hill-climbing* y teniendo en cuenta el orden en que se ha recorrido el vecindario de cada grilla, se construyó una nueva variante, que se denominará *hill-climbing* híbrido. La idea de este nuevo método consiste en comenzar por un curso y evaluar todos los posibles swaps dentro de su horario de cursada, tomar el que más disminuya la penalidad y repetir el proceso con el curso siguiente. La iteración se completa cuando el proceso es aplicado al último curso. En otras palabras, se emplea *steepest-ascent hill-climbing* al horario de cursada de cada curso. La gran diferencia con los dos primeros algoritmos que se implementaron, es que este nuevo método permite ir mejorando gradualmente el horario de cursada de cada curso. Por otra parte, el método híbrido permite aplicar mayor cantidad de cambios en menos tiempo que el *steepest-ascent hill-climbing*.

En el Algoritmo 6 (Apéndice III, página 85) se presenta el pseudocódigo de la implementación del nuevo método. Notar que se mantuvieron los pasos laterales, ahora limitados a 1500, y la lista tabú, cuya memoria fue ampliada a 110. El método se aplicó en paralelo para las 192 grilla iniciales admisibles, con un tope de 1000 iteraciones. En la tabla 6.5 se muestran los datos de los tiempos de ejecución. En la figura 6.11 se observa que, a diferencia de métodos anteriores, el tiempo de ejecución no necesariamente desciende a medida que aumenta σ. Esto podría atribuirse a que el algoritmo recorre las mesetas y que, por esa razón, incluso si ya se encontraba en un punto favorable (como una grilla inicial generada con $\sigma = 223$) no concluía la exploración del espacio de búsqueda en poco tiempo. Por otro lado, como en cada iteración intenta realizar un cambio en todos los cursos, es menos posible que se 'desperdicien' pasos laterales efectuando swaps que no impacten positivamente en la penlidad en un mismo curso.

	Tiempo (h:mm:ss)
Promedio	0 : 20 : 32
Desvío estándar	0 : 06 : 27
Máximo	1 : 03 : 13
Mínimo	0 : 11 : 19

Cuadro 6.5: Tiempos de ejecución del algoritmo

Como ha ocurrido en los demás casos, el algoritmo se detuvo en todas las ejecuciones por agotar la cantidad máxima de pasos laterales, corroborando una vez más la hipótesis de que el espacio de búsqueda presenta numerosas y extensas mesetas. En la figura 6.10 se grafican boxplots comparando el resultado de aplicar el algoritmo al grupo de las grillas iniciales admisibles. La mediana provista por este método resulta ser menor que la de los antes analizados, por lo que indica que podría ser más efectivo.

Con respecto al desempeño del algoritmo, esta vez se obtuvieron 43 grillas que mejoran la penalidad de la grilla estándar, de las cuales 39 son grillas aceptables. Las proporciones según σ entre las fichas aceptables y mejores que la estándar son las siguientes: dos grillas con $\sigma = 150$, cuatro con $\sigma = 130$, seis con $\sigma = 160$, siete con $\sigma = 140$, diez con $\sigma = 200$ y diez con $\sigma = 223$. Consideramos a las mejores de esas grillas a aquéllas que provienen de grillas iniciales generadas con $\sigma < 200$. Es notable que este método proporciones mayor cantidad de grillas aceptables y mejores que la grilla estándar. La razón de esto es que, al mejorar progresivamente el horario de cada curso, la disminución de la función de penalidad no se ve concentrada en un sólo curso, sino que logra repartirse entre muchos. Además, como se ha mencionado antes, la nueva manera de recorrer los vecinos de una grilla permite que unos pocos cursos no "acaparen" todos los pasos laterales. Por otro lado, también es importante remarcar, como

ha ocurrido hasta ahora, que las mejores grillas se obtuvieron a partir de grillas iniciales generadas con valores relativamente altos de σ.

No sólo la cantidad de grillas mejores que la estándar es mayor, sino que su calidad también es mejorada con respecto a las obtenidas con los otros métodos. Como se observa en la tabla 6.6, la mayoría de las grillas que superan a la grilla estándar lo hacen por más del 15 %, mientras que para los métodos ya examinados, el promedio de mejora estaba más cerca del 10 %.

id	sigma	diferencia_pen_std
150_008	150	-20.24
140_000	140	-19.86
140_005	140	-19.07
160_005	160	-18.80
160_006	160	-18.80
160_007	160	-18.80
160_008	160	-18.80
130_008	130	-16.12
140_001	140	-15.98
140_009	140	-15.98
160_003	160	-14.01
140_002	140	-13.90
160_004	160	-13.05
150_000	150	-11.55
130_002	130	-7.81
140_004	140	-4.48
140_003	140	-3.11
130_006	130	-2.39
130_003	130	-1.29

Cuadro 6.6: Mejora respecto a la grilla estándar una vez aplicado el algoritmo

Como ocurría en los casos de los demás algoritmos, la aplicación a las grillas obtenidas con el modelo relajado no ha otorgado ninguna grilla que mejore la penalidad de la estándar a pesar de contar con un número mayor de iteraciones máximas. No obstante, como se puede apreciar en la figura 6.12, la mejora obtenida respecto a los demás métodos es notable. Este hecho puede atribuirse, una vez más, a la mejora progresiva que se lleva a cabo con cada iteración.

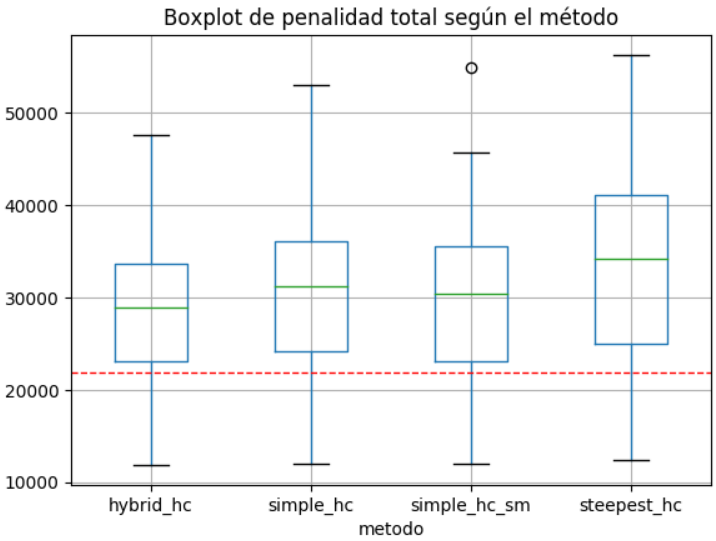

Figura 6.10: Comparación de las penalidades aplicadas al grupo de grillas iniciales admisibles

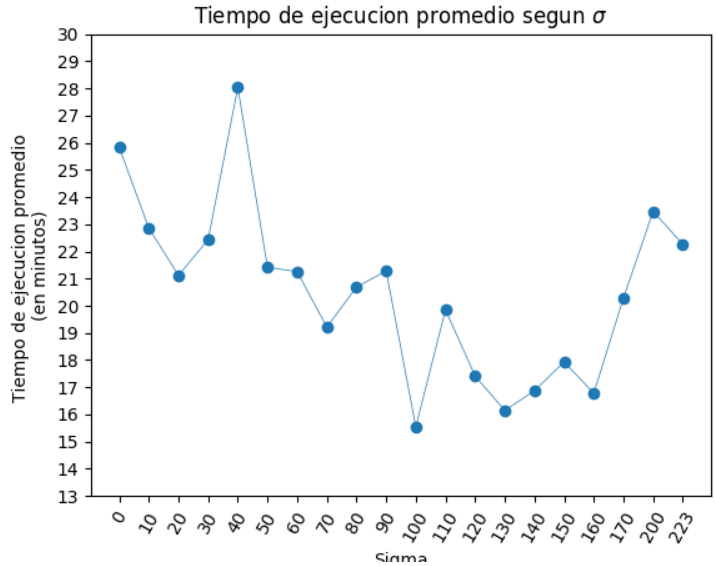

Figura 6.11

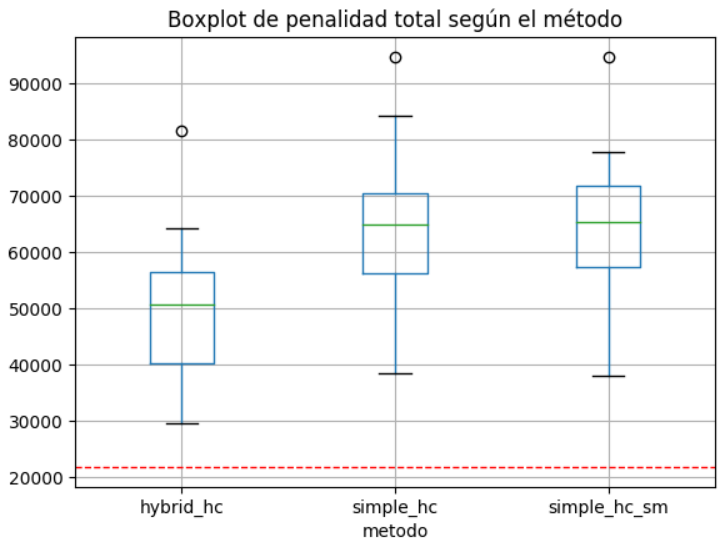

Figura 6.12: Comparación de las penalidades aplicadas al grupo de grillas iniciales no admisibles

6.6. *Hill-climbing* estocástico

Hasta ahora, el criterio para pasar de una grilla a una vecina ha sido que la penalidad no empeore. Sin embargo, resulta interesante experimentar qué sucede si, con cierta probabilidad, se permitiesen movimientos que empeoren la penalidad. Este tipo de método se conoce como *hill-climbing* estocástico: dado un vecino, se evalúa la función de penalidad en él y, en base a una probabilidad que depende de la diferencia de penalidad con la correspondiente al estado actual, se decide si este vecino pasa a ser el estado actual. Notar que si la probabilidad p de que eso ocurra es constantemente 0,5, entonces se tiene un paseo al azar; por otro lado, si $p = 0$, se tiene el método de *hill-climbing* básico.

Hill-climbing estocástico se inspira en el método Metropolis, que es un método de Cadenas de Markov Monte Carlo empleado para obtener una secuencia de muestras aleatorias a partir de una distribución de probabilidad para la cual obtener muestras es complejo. La probabilidad de elegir al nuevo vecino v_n para ser el nuevo estado actual tiene una distribución Bernoulli con parámetro p, donde p depende de la diferencia de penalidad entre el estado actual v_c y el vecino en cuestión y viene dado de la siguiente manera [17]:

$$p(v_c, v_n) = \frac{1}{1 + \exp\left(\dfrac{eval(v_n) - eval(v_c)}{T}\right)}$$

Luego, la regla para moverse de una estado actual a un vecino es ahora probabilística. Por ende, el punto inicial no determina dónde concluirá el algoritmo; incluso, distintas ejecuciones del algoritmo sobre el mismo punto inicial proporcionan distintos resultados. Desde esta perspectiva, la creación de numerosos puntos iniciales puede sonar innecesaria. Sin embargo, se decidió ejecutar el algoritmo para varios puntos iniciales en vez de realizar varias ejecuciones sobre el mismo punto inicial dado que el costo temporal de haberlos generado (a aquellos con $\sigma < 200$) no es excesivo para el objetivo del trabajo.

Notar también la inclusión del parámetro T, que permanece constante a lo largo de la ejecución del algoritmo. Mientras más grande sea el valor de T, menor será la importancia de la mejora relativa entre los puntos v_n y v_c. En particular, si $T \to \infty$, la probabilidad de aceptación se aproxima a 0,5, transformando al *hill-climbing* estocástico en un paseo al azar. Por otro lado, si $T \to 0$, el *hill-climbing* estocástico pasa a comportarse como el *hill-climbing* ordinario. Por tanto, es claro que el parámetro T agrega una capa de dificultad: hay que encontrar T que no sea demasiado grande ni demasiado pequeño.

Para hallarlo, en este trabajo se consideró la escala que manejan las penalidades totales, que se encuentran en el orden de 10^4, y se estudió la función que define a p. Se debe tener en cuenta cuánto se le permitirá empeorar el valor de la función de penalidad a un vecino. Por ejemplo, si se considerara T tal que hay probabilidades no nulas de elegir un vecino que empeore la penalidad por 7000 podría ocurrir que en este vecino, se asigne una hora a un docente en la que no está disponible. Teniendo en cuenta este factor y con un poco de experimentación, se concluyó que el intervalo $[50, 100]$ proporciona valores apropiados para T. Más precisamente, se eligió $T = 50$ pues demostró descender más rápido y al mismo tiempo aceptar vecinos peores (en promedio, entre todos los swaps aceptados, el 33 % empeora la penalidad). En la figura 6.13 se muestra cómo varía la probabilidad de aceptar un vecino nuevo en base a la diferencia entre la nueva penalidad y la actual. Notar la probabilidad de elegir un vecino con la misma penalidad que la actual es de 0,5, sin importar T.

En el Algoritmo 7 (Apéndice III, página 87) se presenta la implementación del *hill-climbing* es-

tocástico al problema que ocupa a este trabajo. Se mantuvo la idea de la mejora progresiva introducida con el *hill-climbing* híbrido, pero con una leve modificación: una vez aceptado a un vecino, se pasa a mejorar el horario del siguiente curso; es poco probable que se evalúan todos los cambios posibles. Además, dado el aspecto probabilístico de este método, no es necesario utilizar pasos laterales ni búsqueda tabú. Ya que este algoritmo no se detiene en mínimos locales ni en mesetas, lo único que limita su ejecución es la cantidad máxima de iteraciones. Lo nuevo que se introduce es un registro de cuál fue la grilla con menor penalidad del recorrido. El problema se ejecutó con 1000 iteraciones máximas. En la tabla 6.7 se muestran los datos sobre los tiempos de ejecución.

	Tiempo (h:mm:ss)
Promedio	0: 30: 46
Desvío estándar	0: 03: 48
Máximo	0: 40: 35
Mínimo	0: 23: 03

Cuadro 6.7: Tiempos de ejecución del algoritmo

Los resultados obtenidos fueron muy satisfactorios: luego de ser expuestas al algoritmo, todas las grillas iniciales admisibles resultaron tener una penalidad menor que la de la grilla estándar. De esas 192 grillas, 186 (el 97 %) resultaron ser aceptables. No sólo se obtuvo una gran cantidad de horarios aceptables, sino que la calidad de la mejora supera a las obtenidas hasta ahora. En la tabla 6.8 se encuentran las mayores mejoras. Notar que, a diferencia de los otro métodos, el valor de σ utilizado para generar la grilla inicial no necesariamente es alto. En la figura 6.14 se muestra la comparación de los resultados de todos los métodos analizados en este trabajo.

id	sigma	Diferencia con Penalidad Estándar
120_009	120	-55.82 %
020_006	20	-55.63 %
030_008	30	-54.03 %
150_002	150	-53.83 %
040_009	40	-53.66 %
140_000	140	-53.33 %
110_004	110	-53.24 %
030_002	30	-52.96 %
000_003	0	-52.84 %
020_002	20	-52.70 %

Cuadro 6.8: Mayores mejoras con respecto a la penalidad de la grilla estándar

Finalmente, otro dato que refleja la ventaja de cambiar la regla de aceptación de un vecino es que este método permitió que, de las 26 grillas iniciales no admisibles, 21 superaran la penalidad de la grilla estándar luego de haberles aplicado el algoritmo. De esas 21, 18 resultaron aceptables. Las mejores 10 se muestran en la tabla 6.9. Dado que la mejora es comparable a la obtenida con las grillas iniciales admisibles, al menos para el caso de estudio, basta generar horarios iniciales con el modelo relajado, lo cual impacta positivamente en el tiempo de ejecución del programa total, puesto que generar los 26 tomó menos de un segundo.

Todo indica que para la confección de horarios el algoritmo de *hill-climbing* estocástico es el más ventajoso, tanto en tiempo de ejecución como en calidad de solución. Por lo menos en el caso de

id	Diferencia con Penalidad Estándar
020	-41.49 %
005	-33.03 %
013	-32.53 %
001	-31.27 %
024	-29.84 %
014	-29.33 %
023	-27.74 %
010	-27.63 %
004	-26.08 %
022	-25.06 %

Cuadro 6.9: Mayores mejoras de las grillas iniciales no admisibles con respecto a la penalidad de la grilla estándar

estudio, la proporción que se obtuvo de horarios aceptables indica que no hacen falta más que un par de ejecuciones del algoritmo para que el programa pueda confeccionar un horario de cursada con penalidad relativamente baja. Así pues, el programa tiene la posibilidad de sugerir, no sólo uno, sino una variedad de "buenos" horarios de cursada, ordenados según su penalidad, al usuario.

Ahora bien, si el *hill-climbing* estocástico es tan superior al resto de los métodos analizados, ¿por qué siquiera se han presentado los resultados de todos ellos en este trabajo? En primer lugar, porque se pretendía mostrar la construcción que condujo a la aplicación del estocástico: primero la idea general del *hill-climbing*, luego la introducción de la lista tabú y los pasos laterales y finalmente la creación de un método *hill-climbing* híbrido que inspiró a mejorar progresivamente y de manera más pareja el horario de cursada. En segundo lugar, porque el *hill-climbing* estocástico resulta poco efectivo al momento de mejorar un horario de cursada ya existente. Supongamos que la disponibilidad de un docente es modificada; crear un nuevo horario significa cambiar la rutina de muchos cursos y docentes, por lo que la opción más sensata es reparar el horario ya existente. Si en esta situación se empleara el algoritmo estocástico, sería muy difícil tener control sobre la cantidad de swaps que se llevarán a cabo. En cambio, aplicando el algoritmo híbrido (con algunas modificaciones), se puede aspirar a realizar la menor cantidad de cambios posibles. Reparar un horario no se encuentra dentro de los objetivos de este trabajo, pero sí es una meta para cumplir a futuro.

Con el desarrollo de un método que permita hallar buenos horarios de cursada en poco tiempo culmina el trabajo dedicado a resolver la Etapa I del problema. A continuación, se llevará a cabo la resolución de la Etapa II: distribuir las horas extracurriculares.

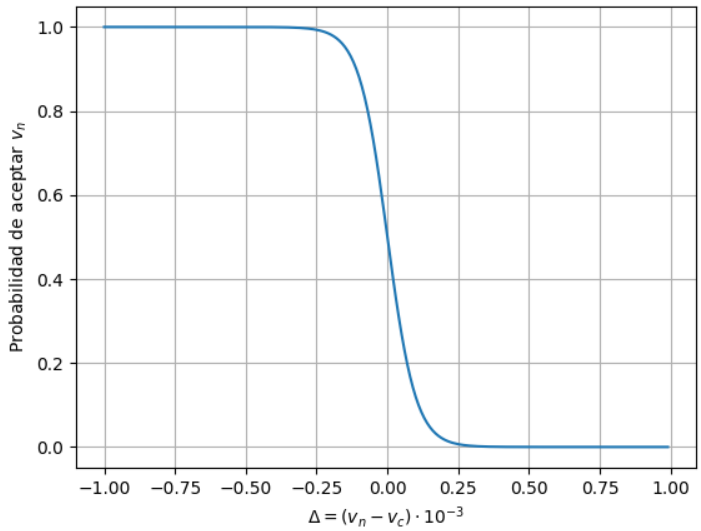

Figura 6.13: Probabilidad de aceptación del vecino en función de la diferencia en la penalidad con respecto al estado actual

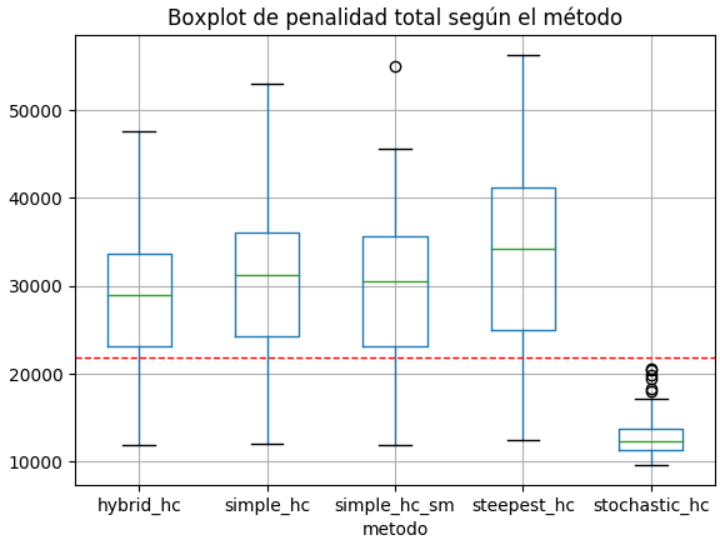

Figura 6.14: Comparación del resultado de aplicar los métodos al conjunto de fichas iniciales admisibles

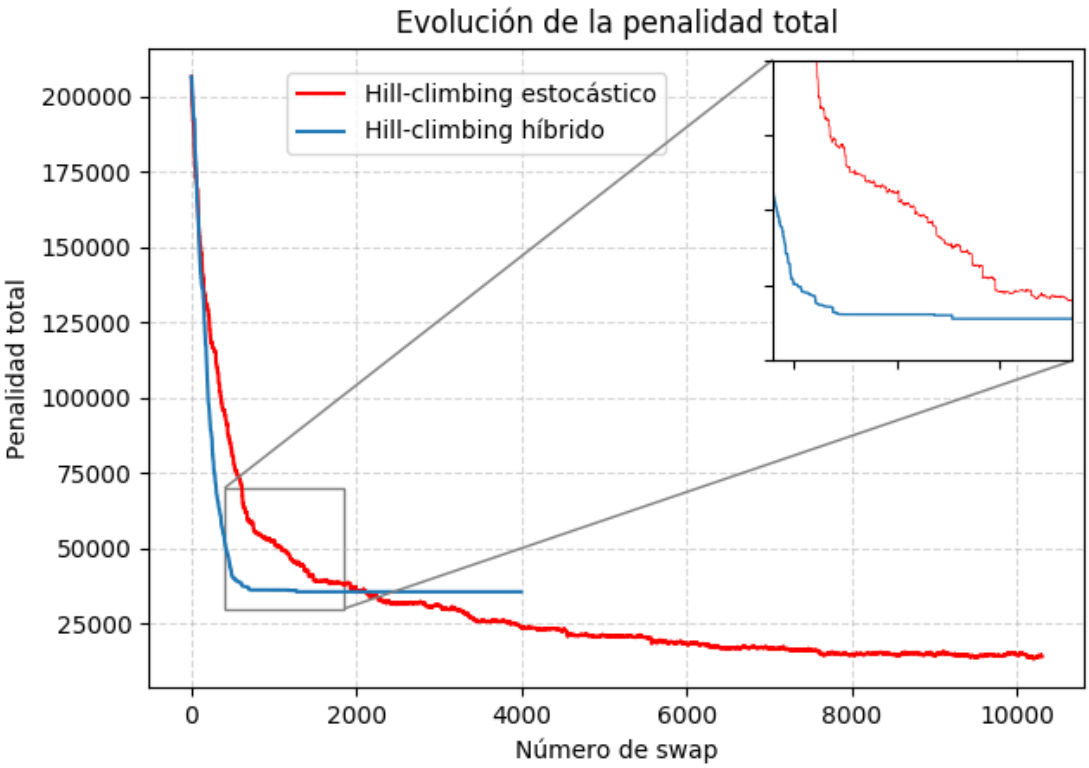

Figura 6.15: Comparación de la evolución de la penalidad total entre el *hill-climbing* estocástico y el híbrido para una grilla inicial admisible generada con $\sigma = 60$. Es interesante notar cómo el *hill-climbing* híbrido permite escapar a las mesetas. También se puede observar la oscilación en el gráfico correspondiente al estocástico, gracias a la aceptación de vecinos que no necesariamente mejoran la penalidad total.

Capítulo 7

Etapa II: Implementación del modelo y resolución

Una vez confeccionado el horario de cursada $\hat{\eta}$ por medio de su modelización en grillas y fichas, la meta de esta etapa es construir el horario extracurricular $\breve{\eta}$. Como se ha anticipado, se recurrirá a resolver problemas de programación lineal entera para determinar la asignación de las actividades de extraclase. Según las autoridades de la escuela, se le otorga mayor prioridad a la asignación de clases de apoyo que a los proyectos, dado que las primeras pueden servir de soporte para mejorar el rendimiento escolar de los alumnos. Por otro lado, las actividades que deban llevar a cabo los docentes que no involucren a los alumnos pueden ser ubicadas en las horas libres que quedaron a partir de la confección del horario de cursada.

Por esta razón, resulta razonable que la solución de la Etapa II se divida en dos partes: primero la asignación de clases de apoyo y luego la asignación de proyectos. Una vez finalizada la segunda tarea, queda confeccionado el horario extracurricular $\breve{\eta}$ y, por ende, el horario escolar completo η. Considerando que a esta altura ya se tiene construido un horario de cursada, se utilizarán las fichas de los docentes para ubicar las horas extracurriculares. De esta manera, se podrá controlar que un docente no quede asignado a varias clases o extraclases al mismo tiempo y además se podrán ocupar algunas de las horas inactivas que hayan quedado luego de la confección del horario de cursada. Sin embargo, especialmente en el caso de las clases de apoyo, se priorizará que estén asignadas en horarios que sean convenientes para los alumnos.

7.1. Clases de apoyo: Modelo de Programación Lineal Entera

La notación empleada para plantear el problema de Programación Lineal Entera que permitirá asignar las clases de apoyo, se presenta en el cuadro 7.1. En este modelo se introducen algunos conjuntos y diccionarios que relacionados a las restricciones blandas presentadas en la sección 2.3.

Por un lado se introduce el diccionario T donde se establece un tope para la cantidad de horas de clases de apoyo diarias de una materia con el fin de lograr una mejor distribución semanal. Esto es relevante ya que los horarios de gimnasia ocupan a los estudiantes en distintos horarios y días, por

Notación	Descripción	Característica en el caso de estudio
M	Conjunto de las materias sobre las que se dan clases de apoyo	$M = \{0, \ldots, 19\}$
M_{TM}	Conjunto de las materias sobre las que se dan clases de apoyo para el turno mañana	$\lvert M_{TM}\rvert = 16$
M_{TT}	Conjunto de las materias sobre las de que se dan clases de apoyo para el turno tarde	$\lvert M_{TT}\rvert = 14$
Δ	Conjunto de días de la semana	$\Delta = \{0, \ldots, 4\}$
H	Conjunto de horas cátedra	$H = \{0, \ldots, 79\}$
H_{TM}	Conjunto de horas cátedra de la mañana	$H_{TM} = \{h \in H \colon \left\lfloor \frac{h}{8} \right\rfloor \equiv 0 \pmod 2\}$
H_{TT}	Conjunto de horas cátedra de la tarde , salvo las prehoras, pues el turno mañana siempre tiene séptima	$H_{TT} = \{h \in H \colon \left\lfloor \frac{h}{8} \right\rfloor \equiv 1 \pmod 2 \wedge h \not\equiv 0 \pmod 8\}$
H_d	Conjunto de horas cátedra correspondientes al día $d \in \Delta$	$H_d = \{h \in H \colon h \equiv d \pmod{16}\}$
$H_d^{(TM)}$	Conjunto de horas cátedra correspondientes a la mañana del día $d \in \Delta$	$H_d \cap H_{TM}$
$H_d^{(TT)}$	Conjunto de horas cátedra correspondientes a la tarde del día $d \in \Delta$	$H_d \cap H_{TT}$
P	Conjunto de profesores que dictan clases de apoyo	$P = \{0, \ldots, 26\}$
K_{TM}	Conjunto de horas que resultan inconvenientes para el turno mañana	$K_{TM} = \{h \in H_{TT} \colon h \bmod 16 \geq 12\}$
K_{TT}	Conjunto de horas que resultan inconvenientes para el turno tarde	$K_{TM} = \{h \in H_{TM} \colon h \bmod 16 \leq 4\}$
lim	Constante que representa el límite preferible de clases de apoyo en la misma hora, por cuestiones de espacio.	$lim = 3$
T	Diccionario cuya clave es una materia y un turno y su valor es la cantidad de horas diarias máximas de apoyo preferibles.	
D	Diccionario cuya clave es un docente y su valor son las horas en las que puede dar clases de apoyo	
F	Diccionario cuya clave es un docente y su valor es un conjunto de horas favorables para dar clases de apoyo	$F_{(p)} \subseteq D_{(p)}$
C	Diccionario con el cargo de clases de apoyo de cada docente	

Cuadro 7.1: Notación de los conjuntos y constantes utilizados en el modelo de programación lineal entera

lo que la mejor manera de maximizar su accesibilidad a las clases de apoyo es procurar que estas estén bien distribuidas. El tope se determina según la cantidad de horas de apoyo que se brindan de la materia en ese turno: a menor oferta, menor es el tope. En el caso de estudio se utilizaron los siguientes topes en función de la oferta $O_{(i,t)}$ de clases de apoyo de la materia i para el turno t:

$$T_{(i,t)}(O_{(i,t)}) = \begin{cases} 0 & \text{si } O_{(i,t)} < 2 \\ 1 & \text{si } O_{(i,t)} = 2 \\ 2 & \text{si } O_{(i,t)} \in \{3,4,5\} \\ 3 & \text{si } O_{(i,t)} \in \{6,7,8\} \\ 4 & \text{c.c.} \end{cases}$$

Naturalmente, al tratarse de una restricción blanda, se aplicarán penalidades a la violación de los topes, pero no determinarán la factibilidad de una solución.

Asimismo, se introducen dos diccionarios que incumben a los docentes. Por un lado, un diccionario D con las horas en las que cada docente puede dictar clases de apoyo; es decir, las horas de disponibilidad del docente menos las horas que ya tiene ocupadas con clases. Por otro lado, un diccionario F con las horas favorables para cada docente: las horas favorables son horas en las que el docente está disponible y que son horas inactivas u horas próximas a a horas activas. Por tanto, para cada docente p, $F_{(p)} \subseteq D_{(p)}$.

Por último, se introduce un diccionario C con el cargo de cada docente en cuanto a clases de apoyo. En C, la clave es un docente y el valor es un conjunto de ternas $(materia, turno, carga)$ que representa que ese docente debe dar *carga* horas de clase de apoyo de *materia* para *turno*.

Sean $p \in P$, $i \in M$, $h \in H$, $d \in \Delta$ y $t \in \{0,1\}$, se definen las variables de decisión del problema:

$$x_{pih} = \begin{cases} 1 & \text{si el profesor } p \text{ da apoyo de } i \text{ en la hora } h \\ 0 & \text{c.c.} \end{cases}$$

y_{it}: cantidad de clases de apoyo de la materia i en horas poco convenientes para el turno t, $y_{it} \in \mathbb{Z}$

v_{itd}: cantidad de horas excesivas de clase de apoyo de la materia i del turno t en el dia d, para (i,t) tales que $O_{(i,t)} \geq 2, v_{itd} \in \mathbb{Z}$

u_{ip}: cantidad de horas de clase de apoyo de la materia i que no son convenientes para el profesor $p, u_{ip} \in \mathbb{Z}$

w_h: sobrecarga de horas de clase de apoyo en la hora $h, w_h \in \mathbb{Z}$

A continuación se presentan los conjuntos de restricciones del problema:

R1: un profesor no puede estar en dos lugares al mismo tiempo:

$$\sum_{i \in M} x_{pih} \leq 1 \quad \forall p \in P, \forall h \in H$$

R2: la séptima hora del turno mañana ocurre al mismo tiempo que la prehora del turno tarde:

$$\sum_{i \in M} x_{p,i,h} + \sum_{i \in M} x_{p,i,h+1} \leq 1 \quad \forall p \in P, \forall h \in \{h \in H : h \equiv 7 \pmod{16}\}$$

R3: no se pueden dictar clases de apoyo durante la séptima hora del lunes del turno tarde:

$$x_{p,i,15} = 0 \quad \forall p \in P, \forall i \in M$$

R4: debe cumplirse la cantidad de horas de apoyo semanales. Las clases de apoyo para el turno mañana deben darse durante las horas de la tarde y viceversa. Las clases de apoyo solo pueden darse en las horas en las que el profesor no esté dando clase. Recordar que los elementos de $C_{(p)}$ son de la forma $c = (c_1, c_2, c_3)$, $c_1 \in M$, $c_2 \in 0, 1$ y c_3 es la carga horaria:

$$\sum_{h \in H_{TT} \cap D_{(p)}} x_{p,c_1,h} = c_3 \quad \forall p \in P, \forall c \in \{c = (c_1, c_2, c_3) \in C_{(p)} \colon c_2 = 0\}$$

$$\sum_{h \in H_{TM} \cap D_{(p)}} x_{p,c_1,h} = c_3 \quad \forall p \in P, \forall c \in \{c = (c_1, c_2, c_3) \in C_{(p)} \colon c_2 = 1\}$$

R5: se desea ubicar las clases de apoyo en horarios convenientes para los alumnos de ambos turnos:

$$\sum_{p \in P} \sum_{h \in K_{TM}} x_{pih} \leq y_{i0} \quad \forall i \in M_{TM}$$

$$\sum_{p \in P} \sum_{h \in K_{TT}} x_{pih} \leq y_{i1} \quad \forall i \in M_{TT}$$

R6: se penaliza la concentración de clases de apoyo de una materia en el mismo día:

$$\sum_{p \in P} \sum_{h \in H_d^{(TT)}} x_{pih} \leq T_{(i,0)} + v_{i0d} \quad \forall d \in \Delta \, \forall i \in M, \, O_{(i,0)} > 1$$

$$\sum_{p \in P} \sum_{h \in H_d^{(TM)}} x_{pih} \leq T_{(i,1)} + v_{i1d} \quad \forall d \in \Delta \, \forall i \in M, \, O_{(i,1)} > 1$$

R7: se desea que las horas de clase de apoyo se integren bien a las fichas de los docentes, preferentemente llenando horas inactivas que hayan quedado de la confección del horario de cursada.:

$$\sum_{h \in H \setminus F_{(p)}} x_{pih} = u_{ip} \quad \forall i \in M \, \forall p \in P$$

R8: es preferible que no hayan demasiadas clases de apoyo dictándose al mismo tiempo:

$$\sum_{p \in P} \sum_{i \in M} x_{pih} \leq lim + w_h \quad \forall h \in H$$

La función objetivo a minimizar para el caso de estudio es la siguiente:

$$\sum_{p \in P} \sum_{i \in M} \sum_{h \in H} x_{pih} + 25 \cdot \sum_{i \in M} \sum_{t \in \{0,1\}} y_{it} + 25 \cdot \sum_{\substack{i \in M \\ O_{(i,t)} > 1}} \sum_{t \in \{0,1\}} \sum_{d \in \Delta} v_{itd} + 20 \cdot \sum_{i \in M} \sum_{p \in P} u_{ip} + 15 \cdot \sum_{h \in H} w_h$$

La formulación completa del problema de Programación Lineal Entera se encuentra en el Apéndice I, en la página 75. Este es un problema de menores dimensiones que el resuelto para generar grillas admisibles. En efecto, con el hardware mencionado en la sección 5.2, la asignación de clases de apoyo es resuelta en 0,08 segundos. Se continúa entonces con la asignación de los proyectos.

7.2. Proyectos: Modelo de Programación Lineal Entera

Recordar que habían dos tipos de actividades que se consideraban proyectos: en las que participan los estudiantes y en las que no. Luego, como ocurría en el caso de las clases de apoyo, se penalizará que la asignación de proyectos en los que participan alumnos a horarios que les resultan inconvenientes. También se le otorgará prioridad a ocupar horas inactivas de los docentes que hayan quedado luego de la distribución de las horas de clase y de las horas de apoyo.

Con respecto a los proyectos, en el caso de estudio no se impusieron límites a cuántos pueden haber por día o por hora cátedra. Si existieran tales límites, las restricciones del problema de PLE que las representan serían similares a las de la sección anterior. Para resolver este problema, se introducen algunos diccionarios y conjuntos.

Se define como R al diccionario donde las claves son los docentes y los valores es el conjunto de proyectos que tiene a cargo cada uno. Así pues, $R_{(p)}$ es un conjunto de tuplas $r = (r_1, r_2, r_3, r_4)$ donde r_1 es el proyecto, r_2 es el turno, r_3 es la carga horaria semanal y r_4 vale 1 si participan los alumnos y 0 en caso contrario.

Además, se introducen distintas penalidades por asignar proyectos donde deben participar alumnos a horarios poco favorables para los ellos. Para ambos turnos se definen las horas: κ_{TM} a partir de las cuales es demasiado tarde para el turno mañana y κ_{TT} antes de la cual es demasiado temprano para el turno tarde. En el caso de estudio, se estableció que κ_{TM} es la tercera hora de la tarde y que κ_{TT} es la cuarta hora del turno mañana. En consecuencia, la función de penalidad que depende de la hora h se define como:

$$\rho(h) = \begin{cases} (\kappa_{TT} + 1 - (h \bmod 16))^2 & \text{si } h \in K_{TT} \\ ((h \bmod 16) - \kappa_{TM} + 1)^2 & \text{si } h \in K_{TM} \\ 0 & \text{c.c.} \end{cases}$$

Así, por ejemplo, la penalidad por asignar un proyecto en la segunda hora del turno mañana del martes se calcula como:

$$\rho(18) = (4 + 1 - (18 \bmod 16))^2 = 9$$

La notación utilizada para este modelo se encuentra en el cuadro 7.2.

Sean $p \in P$, $i \in M$, $h \in H$ y $t \in \{0, 1\}$ se definen las variables de decisión del problema:

$$x_{pih} = \begin{cases} 1 & \text{si el profesor } p \text{ da el proyecto } i \text{ en la hora } h \\ 0 & \text{c.c.} \end{cases}$$

y_{itk} :cantidad de veces que el proyecto i ocurre en la hora inconveniente h para el turno t (definido para $i \in M_{\mathcal{A}}$), $y_{itk} \in \mathbb{Z}$

u_{ip}: cantidad de horas de proyecto i que no son convenientes para el profesor $p, u_{ip} \in \mathbb{Z}$

Los conjuntos de restricciones para el problema son los siguientes:

R1: un profesor no puede estar en dos lugares al mismo tiempo:

$$\sum_{i \in M} x_{pih} \leq 1 \quad \forall p \in P, \forall h \in H$$

Notación	Descripción	Característica en el caso de estudio
M	Conjunto de proyectos	$M = \{0, \ldots, 20\}$
M_{TM}	Conjunto de proyectos para el turno mañana	$\lvert M_{TM}\rvert = 14$
M_{TT}	Conjunto de proyectos para el turno tarde	$\lvert M_{TT}\rvert = 13$
$M_{\mathcal{A}}$	Conjunto de proyectos donde participan alumnos	$\lvert M_{\mathcal{A}}\rvert = 11$
H	Conjunto de horas cátedra	$H = \{0, \ldots, 79\}$
H_{TM}	Conjunto de horas cátedra de la mañana	$H_{TM} = \{h \in H \colon \left\lfloor \frac{h}{8} \right\rfloor \equiv 0 \pmod 2\}$
H_{TT}	Conjunto de horas cátedra de la tarde , salvo las prehoras, pues el turno mañana siempre tiene séptima	$H_{TT} = \{h \in H \colon \left\lfloor \frac{h}{8} \right\rfloor \equiv 1 \pmod 2 \land h \not\equiv 0 \pmod 8\}$
P	Conjunto de profesores que llevan a cabo proyectos	$P = \{0, \ldots, 27\}$
K_{TM}	Conjunto de horas que resultan inconvenientes para el turno mañana	$K_{TM} = \{h \in H_{TT} \colon h \bmod 16 \geq 12\}$
K_{TT}	Conjunto de horas que resultan inconvenientes para el turno tarde	$K_{TM} = \{h \in H_{TM} \colon h \bmod 16 \leq 4\}$
D	Diccionario cuya clave es un docente y su valor son las horas en las que puede llevar a cabo proyectos	
F	Diccionario cuya clave es un docente y su valor es un conjunto de horas favorables para llevar a cabo proyectos	$F_{(p)} \subseteq D_{(p)}$
C	Diccionario con el cargo de clases de apoyo de cada docente	
R	Diccionario donde las claves son los docentes y los valores es el conjunto de proyectos que tiene a cargo cada uno	

Cuadro 7.2: Notación de los conjuntos y constantes utilizados en el modelo de programación lineal entera

R2: la séptima hora del turno mañana ocurre al mismo tiempo que la prehora del turno tarde:

$$\sum_{i \in M} x_{p,i,h} + \sum_{i \in M} x_{p,i,h+1} \leq 1 \quad \forall p \in P, \forall h \in \{h \in H : h \equiv 7 \pmod{16}\}$$

R3: no se pueden dictar proyectos durante la séptima hora del Lunes del turno tarde:

$$x_{p,i,15} = 0 \quad \forall p \in P, \forall i \in M$$

R4: debe cumplirse la cantidad de horas de proyecto. Los proyectos para el turno mañana deben darse en el turno tarde y viceversa. Los proyectos solo pueden darse en las horas en las que el profesor no esté dando clase:

$$\sum_{h \in D_{(p)} \cap H_{TT}} x_{p,r_1,h} = r_3 \quad \forall p \in P, \forall (r_1, 0, r_3, r_4) \in R_{(p)}$$

$$\sum_{h \in D_{(p)} \cap H_{TM}} x_{p,r_1,h} = r_3 \quad \forall p \in P, \forall (r_1, 1, r_3, r_4) \in R_{(p)}$$

R5: se desea ubicar los proyectos en los que participan alumnos en horarios convenientes para los alumnos de ambos turnos:

$$\sum_{p \in P} x_{pih} \leq y_{i0h} \quad \forall h \in K_{TM} \forall i \in M_{\mathcal{A}}$$

$$\sum_{p \in P} x_{pih} \leq y_{i1h} \quad \forall h \in K_{TT} \forall i \in M_{\mathcal{A}}$$

R6: se desea que las horas de proyectos se integren bien a las fichas de los docentes, preferentemente ocupando horas inactivas:

$$\sum_{h \in H \setminus F_{(p)}} x_{pih} = u_{ip} \quad \forall i \in M \, \forall p \in P$$

La función a minimizar el la siguiente:

$$\sum_{p \in P} \sum_{i \in M} \sum_{h \in H} x_{pih} + 10 \cdot \sum_{i \in M} \sum_{t \in \{0,1\}} \sum_{h \in H} \rho(h) y_{ith} + 20 \cdot \sum_{i \in M} \sum_{p \in P} u_{ip}$$

La formulación completa del modelo se encuentra en el Apéndice I, en la página 76. Al igual que ocurría con el modelo para asignar las clases de apoyo, este problema es de rápida resolución: el solver necesitó sólo 0,1 segundos. La asignación de horas para los proyectos es el último paso en la confección del horario escolar, por lo tanto el procedimiento de construcción termina una vez obtenido el resultado de este problema de PLE.

Capítulo 8

Resultados finales y conclusiones

Resumiendo lo desarrollado a lo largo de este trabajo, en un principio se quería encontrar una solución al siguiente problema de minimización:

$$\begin{array}{ll} \text{Dados} & \mathcal{L}, \mathcal{R}_S, \mathcal{H} \\ \text{minimizar} & \Psi(\eta, \mathcal{R}_S) \\ \text{sujeto a} & \texttt{VALIDO}(\eta) = 1, \\ & (l, h_l) \in \eta \qquad l \in \mathcal{L}, h \in \mathcal{H} \end{array}$$

Donde η es un horario escolar según lo definido en la sección 3.1. El problema fue luego dividido en dos etapas: encontrar un horario de cursada aceptable con penalidad mínima y construir un horario extracurricular que se adapte a él. Sin embargo, acorde a lo sugerido en la literatura y en pos de un tiempo de cómputo no demasiado extenso, se ajustó el objetivo de la primera etapa a encontrar horarios de cursada aceptables con baja penalidad, pero no necesariamente mínimos globales. Para hallarlos se generaron numerosos horarios de cursada iniciales y se les aplicaron diversas variantes de *hill-climbing*. Se concluyó que la estocástica era la más eficiente para obtener horarios de cursada aceptables y de buena calidad, tomando como parámetro de comparación al horario de cursada elaborado manualmente. Finalmente, se resolvió la segunda etapa del problema mediante dos modelos de PLE.

En las siguientes dos secciones se exponen los resultados obtenidos al aplicar el programa para la confección de un horario escolar del caso de estudio y la comparación con los horarios elaborados manualmente. Luego se desarrollarán las conclusiones obtenidas y el trabajo futuro.

8.1. Aplicación al horario escolar de 2017

A partir de los resultados obtenidos en la sección 6.6 con la aplicación del *hill-climbing* estocástico, se preservaron los diez mejores horarios de cursada. A ellos se les aplicó la asignación de las horas extraclase y se conservó el horario con menor penalidad por horas inactivas de docentes. Claro está, como ha sido mencionado anteriormente, el programa puede devolver tantos horarios como el usuario necesite, clasificándolos acorde a su penalidad total.

En el Apéndice IV se exhiben el horario de cursada estándar y el generado por el programa desarro-

llado en este trabajo. En las figuras IV.1 y IV.2 (páginas 89 y 90) se muestran el horario de clases del turno mañana y tarde del horario elaborado manualmente, respectivamente, mientras que en las figuras IV.3 y IV.4 (páginas 91 y 92) se muestran el horario de clases del turno mañana y tarde generados por el progama.

En el cuadro 8.1 se encuentra la comparación entre las penalidades del horario de cursada estándar (grilla manual) y del generado con el programa (grilla automática), así como también la diferencia relativa entre ambos, calculada como:

$$\left(\frac{\texttt{penalidad\_grilla\_automatica}}{\texttt{penalidad\_grilla\_manual}} - 1\right) \cdot 100$$

	Grilla automática	Grilla manual	Diferencia relativa
Penalidad por no respetar disponibilidad	0	0	0 %
Penalidad por séptima hora del Lunes TT	0	0	0 %
Penalidad por horas inactivas	1877,73	5267,55	−64,35 %
Penalidad por recreos interruptores	950	1500	−36,67 %
Penalidad por falta de bloques de materias	4485	6325	−29,09 %
Penalidad por cantidad de triples	600	4200	−85,71 %
Penalidad por horas separadas de materias	0	2600	−100 %
Penalidad por cantidad excesiva de prehoras	0	0	0 %
Penalidad por uso excesivo de recursos	1760	2000	−12 %
Penalidad total	9672,73	21892,55	−55,82 %

Cuadro 8.1: Comparación de valores de penalidad para la grilla manual y la generada automáticamente

Como se puede observar, el horario de cursada generado automáticamente supera al elaborado manualmente en todas las categorías, salvo en las que el horario estándar no incurría en ninguna penalidad. Por ejemplo, en el horario de cursada estándar ocurría siete veces que se dictaban tres horas de la misma materia en un sólo día, mientras que esto ocurre una sola vez en el horario generado automáticamente. Asimismo, se logró reducir completamente la ocurrencia de materias que se dictaban en horas separadas a lo largo de un mismo día y se mejoró el uso de los recursos. Tal y como se requería por la definición de aceptabilidad, el horario generado automáticamente no incurrió en penalidad por asignar docentes a horarios en los que no estuviesen disponibles ni por asignar materias a la séptima hora del turno tarde los Lunes, donde suele tomar lugar el Taller Docente. Asimismo, se puede observar en las grillas que el horario generado con el programa respeta las simultaneidades: la materia $M7$ debe darse simultáneamente en los pares de cursos 6 y 7, 8 y 9 y 10 y 11 del turno mañana y en los cursos 4 y 5 del turno tarde.

Con respecto a las horas extracurriculares, en el horario generado automáticamente sólo dos horas de apoyo y cinco horas de proyectos fueron asignadas a horarios poco convenientes para los alumnos. El programa genera también archivos de Excel con la ubicación de las clases de apoyo de cada materia, para que sea más fácil indicarle a los alumnos cuando pueden asistir a ellas. Un ejemplo de cronograma de clases de apoyo se encuentra en la figura 8.1. El programa no sólo devuelve los horarios de cursada en formato legible en Excel, tal como se observa en la figura IV.3, sino que también produce las fichas de cada docente para que estén a disposición de los profesores y de los directivos. En la figura 8.2 se muestra la ficha de uno de los docentes a partir del horario generado automáticamente[1].

[1] En la utilización normal del programa se devuelven los archivos de Excel con los nombres de cada materia y docente.

MATERIA: LENGUA					
HORA	LUNES	MARTES	MIERCOLES	JUEVES	VIERNES
PH					
1º					
2º					
3º					
4º					
5º	X		X		X
6º			X	X	
7º					
PH					
1º	X			X	
2º	X	X			X
3º		X	X		X
4º					
5º					
6º					
7º					

Figura 8.1: Horarios de clases de apoyo de Lengua

DOCENTE: P19					
HORA	LUNES	MARTES	MIERCOLES	JUEVES	VIERNES
PH					
1º	C0				C1
2º	C0				C1
3º	C5		C2		C0
4º	C5		C2		C0
5º	C1		C1		C5
6º	C2		C1		
7º	C2		C0		C2
PH					
1º	P7		C1		P4
2º	P7		C1		APOYO M11
3º	C1		APOYO M11		APOYO M11
4º	C1				C1
5º			C0		C0
6º	C0		C0		C0
7º					

Figura 8.2: Ficha de un docente para el horario generado automáticamente

8.2. Aplicación al horario escolar de 2018

Para aplicar el programa a la elaboración automática del horario escolar para el 2018, en favor de menor tiempo de cómputo, se decidió sólo generar horarios iniciales con el modelo relajado. El solver

Aquí se utilizó una codificación por lo acordado con la escuela.

logró resolver el problema y poblar el *pool* de soluciones en 0,63 segundos. Se obtuvieron así 28 horarios iniciales no necesariamente admisibles a los que se les aplicó paralelamente el método de *hill-climbing* estocástico. Como era de esperar, la etapa de optimización fue la que consumió la mayor cantidad de tiempo: una hora y cuarenta y ocho minutos. De los 28 horarios iniciales, todos los obtenidos luego de aplicar *hill-climbing* estocástico lograron superar al confeccionado manualmente. Sin embargo, 18 resultaron aceptables. Al horario aceptable con menor penalización se le aplicó la distribución de horas de extraclase que, como ocurrió en el caso del horario de 2017, ocupó menos de un segundo. Todo este procedimiento se llevó a cabo en una computadora equipada con el siguiente hardware: procesador Intel Core i5 7400,3.5 GHZ, 8-GB RAM. En total, la generación del horario escolar tardó casi dos horas.

En las figuras IV.5 y IV.6 (Apéndice IV, páginas 93 y 94) se muestran los horarios generados manualmente del turno mañana y turno tarde, respectivamente. Asimismo, en las figuras IV.7 y IV.8 (Apéndice IV, páginas 95 y 96) se exhiben los horarios generados automáticamente. En el cuadro 8.2 se encuentra la comparación de las penalidades, tal como se mostró para el horario del anterior año.

	Grilla automática	Grilla manual	Diferencia relativa
Penalidad por no respetar disponibilidad	0	0	0 %
Penalidad por séptima hora del Lunes TT	0	0	0 %
Penalidad por horas inactivas	1688,10	4602,62	−63,32 %
Penalidad por recreos interruptores	1150	1700	−32,35 %
Penalidad por falta de bloques de materias	3105	6095	−49,06 %
Penalidad por cantidad de triples	1200	7200	−83,33 %
Penalidad por horas separadas de materias	650	5200	−87,50 %
Penalidad por cantidad excesiva de prehoras	0	0	0 %
Penalidad uso excesivo de recursos	480	6160	−92,21 %
Penalidad total	8273,10	30957,62	−73,28 %

Cuadro 8.2: Comparación de valores de penalidad para la grilla manual y la generada automáticamente

Es importante aclarar que entre los dos años hubieron varios cambios. Por un lado, hubieron modificaciones en el plantel docente debido a renuncias y a jubilaciones. Los cargos de los docentes que no están más en la escuela se presentan a concurso. Si para Febrero todavía ningún docente reclama las horas correspondientes a esos cargos, se confecciona el horario escolar sin tener en cuenta su posible disponibilidad horaria. Una vez confeccionado el horario, el docente que desee concursar por esas horas deberá adaptar su disponibilidad horaria a ellas. Por esta razón, en el programa se considera que las materias que no tienen asignados profesores todavía, son dictadas por docentes con total disponibilidad horaria. De esta manera, en el proceso de optimización hay mayor libertad para intercambiar horarios y, al mismo tiempo, queda conformado para el futuro docente un horario que no posea muchas horas inactivas. Debido a la cantidad de cargos que debe ser cubiertos y la ductilidad que esto le proporciona al proceso de elaboración del horario, se han podido formar más bloques de materias.

Otra modificación que vale la pena mencionar es la introducción de nuevas materias debido a la disposición de la NES. Muchas de esas materias requieren el uso de las salas de computación. A pesar de esto, la escuela no ha sido adecuada a la nueva demanda, lo que significa un mayor requerimiento de los mismos recursos con los que se contaba el año anterior. La mayor diferencia en cuanto a penalidad respecto al horario estándar se encuentra en esta categoría, por lo que se demuestra que el algoritmo ha contribuido al uso eficiente de los recursos.

Entre otros logros, se ha logrado disminuir la cantidad de triples: en el horario estándar habían doce, mientras que en el horario obtenido con el programa hay sólo dos. Con respecto a las horas extraclase, sólo han quedado asignadas a horarios poco convenientes para los alumnos dos horas de clases de apoyo y cuatro horas de proyectos.

8.3. Conclusiones y trabajo futuro

En el caso de estudio y en otras escuelas, según lo hablado con varios docentes, la elaboración manual de horarios es una tarea que comienza alrededor de Noviembre y debe estar terminada para mediados de Febrero, cuando comienza el nuevo año escolar y debe comunicarse a los docentes cómo quedan conformadas sus fichas. No es que la tarea requiera la dedicación exclusiva de todo ese tiempo, sino que, como debe efectuarla un directivo o un docente, quienes a su vez tienen otras ocupaciones mas urgentes, la confección del horario se lleva a cabo espaciada y progresivamente. En comparación, el programa desarrollado en este trabajo permite confeccionar un horario en poco menos de dos horas. De hecho, brinda la posibilidad de elegir entre varios horarios candidatos. Por lo tanto, si debieran modificarse algunas características como la disponibilidad de ciertos docentes, se podría volver a ejecutar el programa para que elabore nuevos horarios acorde a las nuevas exigencias; siempre y cuando el ciclo lectivo no haya comenzado aún. Como se ha mostrado en las secciones anteriores, el programa no sólo efectúa la tarea rápidamente sino que también genera horarios que son mejores que los elaborados manualmente, según las prioridades que establecieron los directivos. Por tanto, puede concluirse que el programa cumple con el objetivo propuesto: armar horarios de calidad rápidamente.

En el Capítulo 5, donde se generaron los horarios de cursada admisibles iniciales, quedó en evidencia cuán complejo es el problema si la meta es encontrar un mínimo global. Recordar que el *solver* tomaba más de cuatro horas para resolver el problema para $\sigma = 223$. En ese modelo se buscaba maximizar la cantidad de bloques que forman las materias, pero no se consideraban muchas otras de las restricciones blandas como, por ejemplo, las horas inactivas de los docentes o la cantidad de prehoras del turno tarde. Todo esto indica que si se hubiese planteado el problema de elaboración de un horario de cursada como un problema de PLE, el tiempo de obtención de un horario de cursada, suponiendo que hubiese sido razonablemente aceptable, hubiese superado ampliamente al del programa desarrollado.

En contraste, se consideró que la utilización de métodos de búsqueda local permiten proponer una función de penalidad que represente mejor a los criterios que guían la elaboración de horarios en la vida real. Entre las ventajas de este enfoque, se encuentran también que la función de penalidad no necesariamente debe ser lineal y que su implementación es más sencilla.

Además, la aplicación de los algoritmos de búsqueda local permitió plantear ciertas hipótesis sobre el *espacio* de los horarios de cursada: todo parece apuntar a que existen extensos lomos y que los valles son planos, dado que todas las ejecuciones culminaron al agotar la cantidad máxima de pasos laterales y no por la ausencia de vecinos de igual o menor penalidad (mínimo local). En este contexto, la variante estocástica del *hill-climbing* es la más próspera, puesto que ofrece una probabilidad de aceptar un empeoramiento razonable de la penalidad, permitiendo el escape de las mesetas, amén de tener un tiempo de ejecución que se ajusta a la meta de este trabajo.

Como ocurre en todos los casos en los que se quiere automatizar la programación de horarios, es esencial tener presente la opinión y el procedimiento que realiza la persona encargada de elaborar los horarios manualmente. Ya que debe llevar a cabo el trabajo regularmente, ha adquirido mucha destreza en esta tarea y muchas de las ideas o técnicas que desarrolló pueden servir de inspiración

al momento de automatizar el proceso. Tal es el caso de este trabajo. La ubicación prioritaria de las materias que se dictan con simultaneidad y la división de la elaboración del horario en dos etapas, así como también los valores elegidos para las penalizaciones, fueron decisiones basadas en técnicas que utilizaban los directivos y la profesora que armaba manualmente el horario. Se mantuvo un diálogo con ellos a lo largo de la elaboración del trabajo y quedaron muy satisfechos con los resultados que brindó este programa; lamentablemente, fueron obtenidos a mitad del presente año por lo que no pudieron ser aplicados.

Tomando al programa desarrollado en este trabajo como base, se pueden añadir ciertas mejoras. Ya se ha adelantado la necesidad de implementar un algoritmo que permita reparar un horario efectuando la menor cantidad de swaps posible. Otra mejora consisten en incluir en el modelo la preferencia que tienen los docentes por dictar clases en ciertas horas del turno mañana o del turno tarde. Esto supone también la implementación de una función que calcule el balance de cuántas preferencias de cada docente están siendo satisfechas. Asimismo, se pueden agregar más capas de complejidad a ciertos sumandos de la función de penalidad para reflejar aún mejor la realidad. Por ejemplo, la función que penaliza horas inactivas de los docentes podría tener en cuenta no sólo la cantidad, sino también qué horas tienen libres: las horas libres más cercanas al mediodía podrán ser ocupadas en la Etapa II con clases de apoyo o proyectos, por lo que no deberían recibir igual penalización que las que se encuentran alejadas.

Más en general, si bien el orden de prioridades establecido por los directivos de la escuela con la que se trabajó es razonable, también sería interesante encontrar una escala de preferencias para el cumplimiento de las restricciones blandas más homogéneo u objetivo, que pudiese aplicarse a todas las escuelas y que no necesariamente dependa del criterio de cada autoridad. No porque se dude de este último, sino porque en el estado actual del programa, el ajuste de las penalidades no es lo suficientemente intuitivo como para ser manipulado fácilmente.

Con todo esto en mente, el autor de este trabajo se propone seguir trabajando sobre este proyecto con el objetivo de mejorar el programa ya desarrollado, haciéndolo más accesible, fácilmente utilizable y aplicable. Si bien la elaboración de horarios es un problema ínfimo comparado con otros desafíos que se le plantean a la escuela como institución, el objetivo es que futuras iteraciones de este programa sirvan para resolverlo de la mejor manera posible y, de ese modo, aunque sea aportar una solución a la escuela y restar una preocupación a los educadores.

Apéndice I

Modelos de PLE

Modelo de PLE para generación de horarios de cursada admisibles:

$$\text{mín}: \quad \sum_{i\in M}\sum_{h\in H}\sum_{c\in C} x_{ikc} + 20\cdot\sum_{i\in M}\sum_{c\in C}\sum_{d\in\delta} w_{icd} + 25\cdot\sum_{(i,c)\in E_{(\sigma)}} y_{ic}$$

$$\text{s.a.}: \quad \sum_{i\in M}\sum_{h\in H_{TT}} x_{ihc} = 0 \quad \forall c\in C_{TM}$$

$$\sum_{i\in M}\sum_{h\in H_{TM}} x_{ihc} = 0 \quad \forall c\in C_{TT}$$

$$\sum_{i\in M}\sum_{h\in H_d} x_{ihc} \leq 7 \quad \forall c\in C_{TM},\, \forall d\in\Delta$$

$$\sum_{i\in M}\sum_{h\in H_d} x_{ihc} \geq 7 \quad \forall c\in C_{TM},\, \forall d\in\Delta$$

$$\sum_{i\in M}\sum_{h\in H_d} x_{ihc} \leq 8 \quad \forall c\in C_{TT},\, \forall d\in\Delta$$

$$\sum_{i\in M}\sum_{h\in H_d} x_{ihc} \geq 6 \quad \forall c\in C_{TT},\, \forall d\in\Delta$$

$$\sum_{h\in H_d} x_{ihc} \leq 2 + w_{icd} \quad \forall c\in C,\, \forall i\in M,\, \forall d\in\Delta$$

$$\sum_{d\in\Delta} w_{icd} \leq 1 \quad \forall i\in M,\, \forall c\in C$$

$$\sum_{i\in M} x_{ihc} \leq 1 \quad \forall c\in C,\, \forall k\in H$$

$$\sum_{h\in D_{(i,c)}} x_{ihc} = cgs_{i,c} \quad \forall (i,c)\in\{(i,c)\in M\times C\colon cgs_{i,c}>0\}$$

$$\sum_{h\in\{0,16,32,48,64\}} x_{ihc} = 0 \quad \forall i\in M,\, \forall c\in C$$

$$\sum_{i\in M} x_{ihc} = 1 \quad \forall c\in C_{TM},\, \forall h\in\{h\in H_{TM}\colon h\not\equiv 0 \pmod 8 \wedge h\not\equiv 7 \pmod 8\}$$

$$\sum_{i \in M} x_{ihc} = 1 \quad \forall c \in C_{TT}, \forall h \in \{h \in H_{TT} \colon h \not\equiv 0 \pmod 8 \wedge h \not\equiv 7 \pmod 8\}$$

$$\sum_{i \in M} x_{i,15,c} = 0 \quad \forall c \in C_{TT}$$

$$\sum_{(i,c) \in K} x_{ihc} \leq 1 \quad \forall h \in H, \forall K \in CK_{TM}$$

$$\sum_{(i,c) \in K} x_{ihc} \leq 1 \quad \forall h \in H, \forall K \in CK_{TT}$$

$$x_{k_1,h,k_2} + x_{k_3,h+1,k_4} \leq 1 \quad \forall h \in \{h \in H_{TM} \colon\ h \equiv 7 \pmod 8\}, \forall k = (k_1, k_2, k_3, k_4) \in CK_{PH}$$

$$x_{s_1,h,s_k} = x_{s_1,h,s_j} \quad \forall h \in H, \forall s_k,\ s_j \in s_2,\ s_k \neq s_j, \forall s \in S$$

$$u_{s_1,h,s_r} = 0 \quad \forall h \in H \setminus B_{(s_1,s_r)}, \forall s \in S$$

$$\sum_{h \in B_{(s_1,s_r)}} u_{s_1,h,s_r} = \beta_{(s_1,s_r)} \quad \forall s \in S$$

$$u_{s_1,h,s_r} + u_{s_1,h+1,s_r} \leq 1 \quad \forall h \in B_{(s_1,s_r)}, \forall s \in S$$

$$x_{s_1,h,s_r} + x_{s_1,h+1,s_r} \geq 2 - (1 - u_{s_1,k,s_r}) \cdot 30 \quad \forall h \in B_{(s_1,s_r)}, \forall s \in S$$

$$u_{ihc} = 0 \quad \forall h \in H \setminus B_{(i,c)}, \forall (i,c) \in E_{(\sigma)}$$

$$\sum_{h \in B_{(i,c)}} u_{ihc} = \beta_{(i,c)} - y_{(ic)} \quad \forall (i,c) \in E_{(\sigma)}$$

$$y_{ic} \leq \beta_{(i,c)} \quad \forall (i,c) \in E_{(\sigma)}$$

$$u_{ihc} + u_{i,h+1,c} \leq 1 \quad \forall h \in B_{(i,c)}, \forall (i,c) \in E_{(\sigma)}$$

$$x_{ihc} + x_{i,h+1,c} \geq 2 - (1 - u_{ihc}) \cdot 30 \quad \forall h \in B_{(i,c)}, \forall (i,c) \in E_{(\sigma)}$$

$$x_{ihc} \in \{0,1\} \quad \forall i \in M, \forall h \in H, \forall c \in C$$

$$u_{ihc} \in \{0,1\} \quad \forall i \in M, \forall h \in H, \forall c \in C$$

$$w_{icd} \in \{0,1\} \quad \forall i \in M, \forall c \in C, \forall d \in \Delta$$

$$y_{ic} \in \{0, \ldots, \beta_{(i,c)}\} \quad \forall (i,c) \in E_{(\sigma)}$$

Modelo de PLE para asignación de clases de apoyo

$$\text{mín}: \quad \sum_{p\in P}\sum_{i\in M}\sum_{h\in H} x_{pih} + 25\cdot\sum_{i\in M}\sum_{t\in\{0,1\}} y_{it} + 25\cdot\sum_{\substack{i\in M\\ O_{(i,t)}>1}}\sum_{t\in\{0,1\}}\sum_{d\in\Delta} v_{itd} + 20\cdot\sum_{i\in M}\sum_{p\in P} u_{ip} + 15\cdot\sum_{h\in H} w_h$$

$$\text{s.a.}: \quad \sum_{i\in M} x_{pih} \leq 1 \quad \forall p\in P, \forall h\in H$$

$$\sum_{i\in M} x_{p,i,h} + \sum_{i\in M} x_{p,i,h+1} \leq 1 \quad \forall p\in P, \forall i\in M, \forall h\in\{h\in H\colon h\equiv 7 \pmod{16}\}$$

$$x_{p,i,15} = 0 \quad \forall p\in P, \forall i\in M$$

$$\sum_{h\in H_{TT}\cap D_{(p)}} x_{p,c_1,h} = c_3 \quad \forall p\in P, \forall c\in\{c=(c_1,c_2,c_3)\in C_{(p)}\colon c_2=0\}$$

$$\sum_{h\in H_{TM}\cap D_{(p)}} x_{p,c_1,h} = c_3 \quad \forall p\in P, \forall c\in\{c=(c_1,c_2,c_3)\in C_{(p)}\colon c_2=1\}$$

$$\sum_{p\in P}\sum_{h\in K_{TM}} x_{pih} \leq y_{i0} \quad \forall i\in M_{TM}$$

$$\sum_{p\in P}\sum_{h\in K_{TT}} x_{pih} \leq y_{i1} \quad \forall i\in M_{TT}$$

$$\sum_{p\in P}\sum_{h\in H_d^{(TT)}} x_{pih} \leq T_{(i,0)} + v_{i0d} \quad \forall d\in\Delta\,\forall i\in M,\, O_{(i,0)}>1$$

$$\sum_{p\in P}\sum_{h\in H_d^{(TM)}} x_{pih} \leq T_{(i,1)} + v_{i1d} \quad \forall d\in\Delta\,\forall i\in M,\, O_{(i,1)}>1$$

$$\sum_{h\in H\setminus F_{(p)}} x_{pih} = u_{ip} \quad \forall i\in M\,\forall p\in P$$

$$\sum_{p\in P}\sum_{i\in M} x_{pih} \leq lim + w_h \quad \forall h\in H$$

$$x_{pih}\in\{0,1\} \quad \forall i\in M, \forall p\in P, \forall h\in H$$

$$y_{it}\in\mathbb{Z} \quad \forall i\in M, \forall t\in\{0,1\}$$

$$v_{itd}\in\mathbb{Z} \quad \forall i\in M, O_{(i,t)}>1, \forall t\in\{0,1\}, \forall d\in\Delta$$

$$u_{ip}\in Z \quad \forall i\in M, \forall p\in P$$

$$w_h\in\mathbb{Z} \quad \forall h\in H$$

Modelo de PLE para asignación de horas de proyecto

$$\text{mín}: \quad \sum_{p\in P}\sum_{i\in M}\sum_{h\in H} x_{pih} + 10\cdot\sum_{i\in M}\sum_{t\in\{0,1\}}\sum_{h\in H}\rho(h)y_{ith} + 20\cdot\sum_{i\in M}\sum_{p\in P} u_{ip}$$

$$\text{s.a.}: \quad \sum_{i\in M} x_{pih} \leq 1 \quad \forall p\in P,\, \forall h\in H$$

$$\sum_{i\in M} x_{p,i,h} + \sum_{i\in M} x_{p,i,h+1} \leq 1 \quad \forall p\in P,\, \forall i\in M,\, \forall h\in\{h\in H\colon h\equiv 7 \pmod{16}\}$$

$$x_{p,i,15} = 0 \quad \forall p\in P,\, \forall i\in M$$

$$\sum_{h\in D_{(p)}\cap H_{TT}} x_{p,r_1,h} = r_3 \quad \forall p\in P,\, \forall (r_1,0,r_3,r_4)\in R_{(p)}$$

$$\sum_{h\in D_{(p)}\cap H_{TM}} x_{p,r_1,h} = r_3 \quad \forall p\in P,\, \forall (r_1,1,r_3,r_4)\in R_{(p)}$$

$$\sum_{p\in P} x_{pih} \leq y_{i0h} \quad \forall h\in K_{TM} \forall i\in M_{\mathcal{A}}$$

$$\sum_{p\in P} x_{pih} \leq y_{i1h} \quad \forall h\in K_{TT} \forall i\in M_{\mathcal{A}}$$

$$\sum_{h\in H\setminus F_{(p)}} x_{pih} = u_{ip} \quad \forall i\in M\, \forall p\in P$$

$$x_{pih}\in\{0,1\} \quad \forall i\in M,\, \forall p\in P,\, \forall h\in H$$

$$y_{ith}\in\mathbb{Z} \quad \forall i\in M_{\mathcal{A}},\, \forall t\in\{0,1\}, \forall h\in H$$

$$u_{ip}\in\mathbb{Z} \quad \forall i\in M,\, \forall p\in P$$

Apéndice II

Cálculo de $|\mathcal{S}|$ y de $|N(G_{\hat{\eta}})|$

Calcular $|\mathcal{S}|$ implica contar la cantidad de grillas que podrían formarse, sean válidas o inválidas. Como no se requiere que la grilla represente un horario de cursada válido, se pueden contar los horarios de cursada posibles para cada curso separadamente, puesto que son independientes entre sí, y multiplicarlos posteriormente. Para el caso de estudio se estableció que hay 40 horas cátedra semanales para cada curso y se tiene un total de 22 cursos. Sea $M_j = \{m_0, \ldots, m_{k_j}\}$ una numeración de las materias que se dictan en el curso j y sea $cgs(m_i)$, $0 \leq i \leq k_j$ la carga horaria semanal de la materia i-ésima en el curso j, la cantidad de posibles horarios de cursada para este curso se calcula de la siguiente manera:

$$\varrho_j = \prod_{i=0}^{k_j} \binom{40 - \sum_{h<i} cgs(m_h)}{cgs(m_i)}$$

Luego, la cantidad total de posibles grillas para el caso de estudio es:

$$|\mathcal{S}| = \prod_{j=0}^{22} \varrho_j \approx 9{,}91 \times 10^{774}$$

Para calcular $|N(G_{\hat{\eta}})|$ tampoco resulta relevante si la grilla vecina es válida o inválida. Recordar que dos grillas son consideradas vecinas si una se obtiene a partir de la otra intercambiando el lugar de dos horas cátedra de un curso. Por tanto, basta calcular cuántos horarios de cursada se pueden obtener para cada curso intercambiando un par de horas cátedra y posteriormente sumar[1] los resultados de todos los cursos.

Se explicará el procedimiento de conteo de la cantidad de horarios de cursada vecinos del curso j utilizando un ejemplo. Por simplicidad, supongamos que en vez de 40 horas cátedra semanales, hubiesen 16. Dado el curso j representamos su horario de cursada como una sucesión de letras, donde cada una representa a una materia, y ceros, si en esas horas no se dictan materias:

0GGLLPII0GMLLPPP

[1] Puesto que a partir de una grilla se obtiene una vecina realizando un sólo intercambio en un sólo curso, los casos de intercambio en cada curso son disjuntos

Según ese esquema, en la prehora del primer día no se dictan materias, durante la primera y segunda hora se dicta la materia G, etc. Ahora bien, el problema puede replantearse como contar cuántas palabras distintas pueden formarse con un único intercambio de dos letras; es decir, contar cúantas permutaciones simples existen de la palabra.

Sea $\mathcal{I}$ el conjunto de índices de la palabra, se define la función $R\colon \mathcal{I} \to \mathbb{N}_0$ la cual, dado un índice de la palabra, devuelve cuántas veces se repite la letra de ese índice en los índices mayores. Si i es el índice de la última letra, se define $R(i) = 0$. Por ejemplo, $R(2) = 2$, pues en el índice 2 se encuentra la letra G, la cual se repite 2 veces en lo que resta de la palabra.

Veamos cómo resolver el problema. Tomamos un índice i_0 y consideramos todas las posibles permutaciones simples con las letras de lo que resta de la palabra. A continuación, se restan las repeticiones de la letra correspondiente a ese índice en lo que queda de la palabra, es decir, se resta $R(i_0)$. Se repite este procedimiento para todos los índices y se suman los resultados. Más formalmente, este procedimiento da lugar a la definición de la siguiente función $f\colon \mathcal{I} \to \mathbb{N}_0$, siendo L la longitud de la palabra:

$$f(i) = L - i - R(i)$$

La solución del problema es entonces:

$$\kappa_j = \sum_{i \in \mathcal{I}_|} f(i) = \sum_{i \in \mathcal{I}_|} (L - i - R(i)) = L^2 - \frac{L(L+1)}{2} - \sum_{i \in \mathcal{I}_|} R(i) = \frac{L(L-1)}{2} - \sum_{i \in \mathcal{I}_|} R(i)$$

En el ejemplo se tiene que $L = 16$ y que:

$$\begin{array}{cccccccc} R(1) = 1 & R(2) = 2 & R(3) = 1 & R(4) = 3 & R(5) = 2 & R(6) = 3 & R(7) = 1 & R(8) = 0 \\ R(9) = 0 & R(10) = 0 & R(11) = 0 & R(12) = 1 & R(13) = 0 & R(14) = 2 & R(15) = 1 & R(16) = 0 \end{array}$$

$$\Rightarrow \kappa_j = \frac{16 \cdot 15}{2} - \sum_{i=1}^{16} R(i) = 103$$

De vuelta al caso de estudio, se calcula κ_j para $0 \leq j \leq 22$, recordando que en total hay 40 horas cátedra semanales disponibles para asignar materias, y se suman para obtener la cantidad de vecinos de una grilla:

$$|N(G_{\hat{\eta}})| = \sum_{j=0}^{22} \kappa_j = 15926$$

Apéndice III

Pseudocódigos

Algoritmo 3 *Hill-climbing* simple

Input: G grilla, *datos* información, *iter_max* cantidad máxima de iteraciones permitida

```
1: procedure SIMPLEHILLCLIMBING(G, datos, iter_max)
2:     inicializar F, D, P                                   ▷ aquí se utiliza datos
3:     mejor ← Ψ̂(G)
4:     local ←FALSE
5:     bloq_flag ← FALSE
6:     iter ← 0
7:     while iter ≤ iter_max and local = FALSE do
8:         exit ← FALSE
9:         for curso ∈ C do
10:            if exit = TRUE then
11:                break
12:            end if
13:            inicializar swap_validos
14:            B ← Conjunto de horas que encabezan bloques
15:            for swap ∈ swaps_validos do                   ▷ swap = (h1, h2)
16:                if negarCambio(G, h1, h2) = TRUE then     ▷ aquí se utiliza datos
17:                    continue
18:                end if
19:                profs ← profesores afectados por el swap
20:                bloq_flag ← bloqFlag(G, curso, B, h1, h2)
21:                efectuarSwap(G, bloq_flag, swap)
22:                modificarFichas(F, profs, bloq_flag)
23:                valido ← validar(G, F)                    ▷ aquí se utiliza datos
24:                if valido = FALSE then
25:                    revertir cambios de G, F
26:                else
27:                    actualizarD, P                        ▷ aquí se utiliza datos
28:                    if Ψ̂(G) < mejor then
29:                        mejor ← Ψ̂(G)
30:                        iter ← iter + 1
31:                        exit ← TRUE
32:                        break
33:                    else
34:                        revertir cambios de G, F, D, P    ▷ aquí se utiliza datos
35:                    end if
36:                end if
37:            end for
38:        end for
39:        if exit = FALSE then
40:            local ← TRUE
41:        end if
42:    end while
43: end procedure
```

Algoritmo 4: *Hill-climbing* con movimientos laterales y búsqueda tabú

Input: G grilla, *datos* información, $iter\_max$, k

procedure smHillClimbing($G, datos, iter\_max, k$)
- inicializar F, D, P ▷ aquí se utiliza *datos*
- inicializar T ▷ T es la lista tabú
- $mejor \leftarrow \hat{\Psi}(G)$
- $local \leftarrow$FALSE
- $bloq\_flag \leftarrow$ FALSE
- $iter \leftarrow 0$
- $pasos \leftarrow 0$
- **while** $iter \leq iter\_max$ **and** $local =$ FALSE **and** $pasos \leq k$ **do**
  - $exit \leftarrow$ FALSE
  - **for** $curso \in \mathcal{C}$ **do**
    - **if** $exit =$ TRUE **then**
      - **break**
    - **end if**
    - inicializar $swap\_validos$
    - $B \leftarrow$ Conjunto de horas que encabezan bloques
    - **for** $swap \in swaps\_validos$ **do** ▷ $swap = (h_1, h_2)$
      - **if** $negarCambio(G, h_1, h_2) =$ TRUE **then** ▷ aquí se utiliza *datos*
        - **continue**
      - **end if**
      - $profs \leftarrow$ profesores afectados por el swap
      - $bloq\_flag \leftarrow bloqFlag(G, curso, B, h_1, h_2)$
      - $efectuarSwap(G, bloq\_flag, swap)$
      - $modificarFichas(F, profs, bloq\_flag)$
      - $valido \leftarrow validar(G, F)$ ▷ aquí se utiliza *datos*
      - **if** $valido =$ FALSE **then**
        - revertir cambios de G, F
      - **else**
        - actualizarD, P ▷ aquí se utiliza *datos*
        - $nueva\_pen \leftarrow \hat{\Psi}(G)$
        - **if** $nueva\_pen \leq mejor$ **then**
          - **if** $nueva\_pen = mejor$ **then**
            - $pasos \leftarrow pasos + 1$
            - agregar G a T; si T tiene 51 elementos, eliminar el primero
          - **else**
            - $pasos \leftarrow 0$
            - vaciar T
            - $iter \leftarrow iter + 1$
          - **end if**

```
40:                                   mejor ← Ψ̂(G)
41:                                   exit ← TRUE
42:                                   break
43:                               else
44:                                   revertir cambios de G, F, D, P          ▷ aquí se utiliza datos
45:                               end if
46:                           end if
47:                       end for
48:                   end for
49:                   if exit = FALSE then
50:                       local ← TRUE
51:                   end if
52:               end while
53:           end procedure
```

Algoritmo 5: *Steepest-ascent hill-climbing*

Input: G grilla, *datos* información, *iter_max*, *pasos_max*

procedure STEEPESTHILLCLIMBING($G, datos, iter\_max, pasos\_max$)
  inicializar F, D, P ▷ aquí se utiliza *datos*
  inicializar T ▷ T es la lista tabú
  $mejor \leftarrow \hat{\Psi}(G)$
  $local \leftarrow$FALSE
  $bloq\_flag \leftarrow$ FALSE
  $iter \leftarrow 0$
  $pasos \leftarrow 0$
  $flag\_mejora \leftarrow$ FALSE
  while $iter \leq iter\_max$ **and** $local =$ FALSE **and** $pasos \leq pasos\_max$ **do**
    $mejor\_ant \leftarrow mejor$
    for $curso \in \mathcal{C}$ **do**
      inicializar *swap_validos*
      $B \leftarrow$ Conjunto de horas que encabezan bloques
      for $swap \in swaps\_validos$ **do** ▷ $swap = (h_1, h_2)$
        if $negarCambio(G, h_1, h_2) =$ TRUE **then** ▷ aquí se utiliza *datos*
          continue
        end if
        $profs \leftarrow$ profesores afectados por el swap
        $bloq\_flag \leftarrow bloqFlag(G, curso, B, h_1, h_2)$
        $efectuarSwap(G, bloq\_flag, swap)$
        $modificarFichas(F, profs, bloq\_flag)$
        $valido \leftarrow validar(G, F)$ ▷ aquí se utiliza *datos*
        if $valido =$ FALSE **then**
          revertir cambios de G, F
        else
          actualizar D, P ▷ aquí se utiliza *datos*
          $nueva\_pen \leftarrow \hat{\Psi}(G)$
          if $nueva\_pen \leq mejor$ **then**
            $mejor \leftarrow \hat{\Psi}(G)$
            $mejor\_swap \leftarrow swap$
            $mejor\_curso \leftarrow curso$
            $flag\_mejora \leftarrow$ TRUE
            revertir cambios de G, F, D, P ▷ aquí se utiliza *datos*
          else
            revertir cambios de G, F, D, P ▷ aquí se utiliza *datos*
          end if
        end if
      end for
    end for
    if $flag\_mejora =$ TRUE **then**
      $efectuarSwap(G, bloq\_flag, mejor\_swap)$ en $mejor\_curso$
      $modificarFichas(F, profs, bloq\_flag)$
      actualizar D, P ▷ aquí se utiliza *datos*
      $iter \leftarrow iter + 1$

```
46:         if mejor = mejor_ant then
47:             pasos ← pasos + 1
48:             agregar G a T; si T tiene 51 elementos, eliminar el primero
49:         else
50:             pasos ← 0
51:             vaciar T
52:         end if
53:     else
54:         local ← TRUE
55:     end if
56: end while
57: end procedure
```

Algoritmo 6: *Hill-climbing* híbrido

Input: G grilla, *datos* información, $iter\_max$, $pasos\_max$

procedure HYBRIDHILLCLIMBING($G, datos, iter\_max, pasos\_max$)
- inicializar F, D, P ▷ aquí se utiliza *datos*
- inicializar T ▷ T es la lista tabú
- $mejor \leftarrow \hat{\Psi}(G)$
- $local \leftarrow$FALSE
- $bloq\_flag \leftarrow$ FALSE
- $iter \leftarrow 0$
- $pasos \leftarrow 0$
- $flag\_mejora \leftarrow$ FALSE
- inicializar $marcadores\_mejora$ como lista de FALSE
- **while** $iter \leq iter\_max$ **and** $local =$ FALSE **and** $pasos \leq pasos\_max$ **do**
  - **for** $curso \in \mathcal{C}$ **do**
    - $mejor\_ant \leftarrow mejor$
    - asignar FALSE a la coordenada c de $marcadores\_mejora$
    - inicializar $swap\_validos$
    - $B \leftarrow$ Conjunto de horas que encabezan bloques
    - **for** $swap \in swaps\_validos$ **do** ▷ $swap = (h_1, h_2)$
      - **if** $negarCambio(G, h_1, h_2) =$ TRUE **then** ▷ aquí se utiliza *datos*
        - **continue**
      - **end if**
      - $profs \leftarrow$ profesores afectados por el swap
      - $bloq\_flag \leftarrow bloqFlag(G, curso, B, h_1, h_2)$
      - $efectuarSwap(G, bloq\_flag, swap)$
      - $modificarFichas(F, profs, bloq\_flag)$
      - $valido \leftarrow validar(G, F)$ ▷ aquí se utiliza *datos*
      - **if** $valido =$ FALSE **then**
        - revertir cambios de G, F
      - **else**
        - actualizar D, P ▷ aquí se utiliza *datos*
        - $nueva\_pen \leftarrow \hat{\Psi}(G)$
        - **if** $nueva\_pen \leq mejor$ **then**
          - $mejor \leftarrow \hat{\Psi}(G)$
          - $mejor\_swap \leftarrow swap$
          - $flag\_mejora \leftarrow$ TRUE
          - revertir cambios de G, F, D, P ▷ aquí se utiliza *datos*
        - **else**
          - revertir cambios de G, F, D, P ▷ aquí se utiliza *datos*
        - **end if**
      - **end if**
    - **end for**
    - asignar $flag\_mejora$ a la coordenada c de $marcadores\_mejora$

```
42:         if flag_mejora = TRUE then
43:             efectuarSwap(G, bloq_flag, mejor_swap)
44:             modificarFichas(F, profs, bloq_flag)
45:             actualizar D, P                                  ▷ aquí se utiliza datos
46:             iter ← iter + 1
47:             if mejor = mejor_ant then
48:                 pasos ← pasos + 1
49:                 agregar G a T; si T tiene 111 elementos, eliminar el primero
50:             else
51:                 pasos ← 0
52:                 vaciar T
53:             end if
54:         end if
55:     end for
56:     if todas las coordenadas de marcadores_mejora son FALSE then
57:         local ← TRUE
58:     end if
59: end while
60: end procedure
```

Algoritmo 7 *Hill-climbing* estocástico

Input: G grilla, *datos* información, *iter_max*

```
1: procedure stochasticHillClimbing(G, datos, iter_max)
2:     inicializar F, D, P                                   ▷ aquí se utiliza datos
3:     mejor ← Ψ̂(G)
4:     bloq_flag ← FALSE
5:     iter ← 0
6:     mejor_grilla ← G
7:     minima_penalidad ← Ψ̂(G)
8:     while iter ≤ iter_max do
9:         for curso ∈ C do
10:            inicializar swap_validos
11:            B ← Conjunto de horas que encabezan bloques
12:            for swap ∈ swaps_validos do                   ▷ swap = (h1, h2)
13:                if negarCambio(G, h1, h2) = TRUE then     ▷ aquí se utiliza datos
14:                    continue
15:                end if
16:                profs ← profesores afectados por el swap
17:                bloq_flag ← bloqFlag(G, curso, B, h1, h2)
18:                efectuarSwap(G, bloq_flag, swap)
19:                modificarFichas(F, profs, bloq_flag)
20:                valido ← validar(G, F)                    ▷ aquí se utiliza datos
21:                if valido = FALSE then
22:                    revertir cambios de G, F
23:                else
24:                    actualizar D, P                       ▷ aquí se utiliza datos
25:                    nueva_pen ← Ψ̂(G)
26:                    calcular p
27:                    if Be(p) then                         ▷ Se "tira una moneda" con probabilidad p
28:                        mejor ← nueva_pem
29:                        if nueva_pen < minima_penalidad then
30:                            mejor_grilla ← G
31:                            minima_penalidad ← Ψ̂(G)
32:                        end if
33:                        break
34:                    else
35:                        revertir cambios de G, F, D, P    ▷ aquí se utiliza datos
36:                    end if
37:                end if
38:            end for
39:        end for
40:        iter ← iter + 1
41:    end while
42: end procedure
```

Apéndice IV

Comparación de horarios de cursada

DIA	HORA	CURSO 0	CURSO 1	CURSO 2	CURSO 3	CURSO 4	CURSO 5	CURSO 6	CURSO 7	CURSO 8	CURSO 9	CURSO 10	CURSO 11
LUNES	PH												
	1º	M1	M0	M2	M22	M24	M2	M12	M22	M14	M0	M9	M29
	2º	M1	M0	M2	M22	M24	M2	M12	M22	M14	M0	M9	M29
	3º	M11	M2	M6	M0	M22	M22	M26	M13	M28	M4	M6	M0
	4º	M11	M2	M6	M1	M22	M22	M26	M13	M28	M4	M6	M0
	5º	M0	M1	M11	M2	M2	M6	M22	M0	M4	M29	M28	M18
	6º	M3	M1	M11	M10	M2	M6	M22	M12	M4	M14	M28	M18
	7º	M3	M11	M22	M10	M23	M1	M20	M12	M1	M14	M1	M28
MARTES	PH												
	1º	M10	M6	M24	M0	M0	M22	M12	M15	M0	M4	M18	M1
	2º	M10	M6	M24	M0	M0	M22	M12	M15	M0	M4	M18	M1
	3º	M0	M24	M1	M2	M8	M2	M0	M12	M4	M1	M17	M9
	4º	M0	M24	M23	M2	M8	M2	M0	M12	M4	M1	M17	M9
	5º	M24	M2	M22	M3	M2	M1	M26	M20	M19	M28	M28	M0
	6º	M22	M1	M2	M3	M2	M24	M26	M20	M19	M28	M29	M17
	7º	M22	M0	M2	M23	M1	M24	M22	M13	M1	M29	M29	M17
MIERCOLES	PH												
	1º	M2	M0	M7	M6	M22	M25	M25	M11	M27	M16	M17	M5
	2º	M2	M0	M7	M6	M22	M25	M25	M11	M27	M16	M17	M5
	3º	M0	M11	M25	M11	M6	M11	M7	M7	M0	M27	M28	M5
	4º	M0	M11	M25	M11	M6	M0	M7	M7	M29	M27	M0	M28
	5º	M22	M25	M11	M2	M11	M0	M11	M25	M29	M0	M5	M28
	6º	M11	M25	M0	M0	M11	M7	M11	M25	M28	M21	M5	M6
	7º	M11	M2	M0	M0	M0	M7	M1	M0	M16	M21	M5	M6
JUEVES	PH												
	1º	M6	M10	M7	M22	M0	M0	M11	M6	M7	M7	M18	M28
	2º	M6	M10	M7	M22	M0	M0	M11	M6	M7	M7	M18	M28
	3º	M3	M8	M22	M3	M8	M11	M0	M22	M16	M16	M28	M18
	4º	M3	M8	M22	M3	M8	M11	M0	M1	M16	M29	M28	M18
	5º	M2	M22	M10	M11	M24	M7	M20	M1	M28	M29	M7	M7
	6º	M2	M22	M10	M25	M11	M7	M20	M20	M29	M28	M29	M17
	7º	M23	M23	M0	M25	M1	M24	M7	M7	M29	M28	M29	M17
VIERNES	PH												
	1º	M24	M8	M11	M1	M11	M10	M6	M0	M14	M19	M5	M29
	2º	M24	M8	M11	M1	M11	M10	M6	M0	M14	M19	M5	M29
	3º	M25	M24	M1	M11	M10	M11	M1	M2	M21	M14	M0	M5
	4º	M25	M11	M24	M11	M10	M11	M15	M2	M21	M14	M0	M5
	5º	M22	M11	M0	M24	M25	M1	M15	M11	M7	M7	M1	M28
	6º	M1	M22	M0	M24	M25	M23	M2	M11	M28	M28	M7	M7
	7º	M11	M22	M1	M24	M1	M0	M2	M13	M28	M28	M7	M7

Figura IV.1: Horario de cursada del turno mañana elaborado manualmente y utilizado para el año 2017

DIA	HORA	CURSO 0	CURSO 1	CURSO 2	CURSO 3	CURSO 4	CURSO 5	CURSO 6	CURSO 7	CURSO 8	CURSO 9
LUNES	PH			M22	M0	M2	M13	M4		M29	
	1º	M22	M1	M6	M0	M2	M13	M4	M1	M29	M28
	2º	M22	M1	M6	M6	M20	M12	M21	M1	M0	M28
	3º	M0	M0	M1	M6	M26	M12	M21	M0	M28	M29
	4º	M0	M11	M1	M24	M26	M0	M16	M0	M28	M29
	5º	M11	M2	M24	M22	M12	M11	M29	M21	M1	M5
	6º	M3	M2	M0	M23	M12	M11	M29	M21	M1	M5
	7º										
MARTES	PH			M23	M2	M26			M4		
	1º	M10	M6	M2	M2	M1	M0	M1	M28	M28	M17
	2º	M10	M6	M2	M0	M1	M0	M1	M28	M28	M17
	3º	M1	M0	M3	M1	M6	M1	M29	M14	M18	M9
	4º	M1	M0	M3	M1	M6	M1	M29	M14	M18	M9
	5º	M2	M1	M0	M0	M11	M22	M28	M29	M17	M5
	6º	M2	M22	M0	M8	M11	M6	M28	M29	M17	M5
	7º		M22		M8		M6	M14			M5
MIERCOLES	PH	M3	M25	M22		M15		M3	M4		
	1º	M11	M25	M2	M8	M15	M0	M3	M4	M5	M28
	2º	M11	M0	M2	M8	M22	M15	M0	M16	M5	M28
	3º	M0	M0	M11	M1	M22	M15	M14	M16	M28	M18
	4º	M0	M11	M11	M0	M11	M22	M14	M7	M28	M18
	5º	M23	M11	M1	M22	M11	M22	M27	M7	M0	M17
	6º	M25	M22	M0	M11	M0	M13	M27	M14	M3	M1
	7º	M25	M22	M0	M11	M0	M13		M14	M3	M1
JUEVES	PH	M22	M23		M11	M26					M17
	1º	M22	M10	M11	M24	M7	M7	M16	M28	M0	M0
	2º	M3	M10	M11	M24	M7	M7	M16	M28	M18	M0
	3º	M3	M2	M24	M10	M12	M11	M28	M4	M18	M7
	4º	M2	M2	M24	M10	M12	M11	M28	M7	M29	M18
	5º	M24	M8	M3	M25	M25	M20	M3	M29	M29	M18
	6º	M24	M8	M3	M25	M25	M12	M4	M29	M6	M29
	7º	M24					M12	M4	M16	M6	M29
VIERNES	PH			M11					M0	M5	M6
	1º	M2	M24	M22	M11	M7	M7	M0	M27	M5	M6
	2º	M1	M24	M22	M11	M20	M2	M0	M27	M5	M7
	3º	M0	M24	M25	M2	M20	M2	M28	M19	M17	M7
	4º	M11	M8	M25	M2	M22	M20	M28	M19	M17	M0
	5º	M11	M8	M10	M22	M0	M20	M14	M28	M3	M28
	6º	M6	M11	M10	M22	M0	M25	M19	M28	M9	M28
	7º	M6	M11				M25	M19		M9	

Figura IV.2: Horario de cursada del turno tarde elaborado manualmente y utilizado para el año 2017

DIA	HORA	CURSO 0	CURSO 1	CURSO 2	CURSO 3	CURSO 4	CURSO 5	CURSO 6	CURSO 7	CURSO 8	CURSO 9	CURSO 10	CURSO 11
LUNES	PH												
	1º	M11	M2	M2	M22	M24	M0	M20	M2	M14	M4	M0	M28
	2º	M11	M2	M2	M22	M24	M0	M20	M2	M14	M4	M0	M28
	3º	M6	M6	M22	M6	M22	M25	M26	M1	M16	M19	M29	M5
	4º	M6	M6	M22	M6	M22	M25	M26	M1	M28	M19	M29	M5
	5º	M23	M11	M0	M2	M1	M22	M12	M15	M4	M14	M9	M0
	6º	M3	M22	M11	M2	M1	M22	M12	M15	M4	M14	M9	M17
	7º	M3	M22	M11	M23	M2	M1	M22	M12	M1	M0	M28	M29
MARTES	PH												
	1º	M0	M1	M22	M10	M8	M2	M0	M11	M19	M27	M28	M0
	2º	M0	M1	M24	M10	M8	M2	M0	M11	M19	M27	M28	M0
	3º	M10	M2	M6	M24	M0	M7	M12	M0	M29	M4	M18	M17
	4º	M10	M2	M6	M24	M0	M7	M12	M0	M29	M4	M18	M17
	5º	M1	M0	M7	M22	M10	M24	M20	M12	M4	M1	M0	M5
	6º	M24	M0	M7	M22	M10	M1	M26	M13	M4	M28	M6	M9
	7º	M22	M24	M23	M0	M2	M1	M26	M13	M1	M28	M6	M9
MIERCOLES	PH												
	1º	M2	M0	M25	M11	M6	M11	M15	M22	M27	M0	M17	M18
	2º	M2	M0	M25	M11	M6	M11	M15	M22	M27	M0	M17	M18
	3º	M1	M25	M11	M2	M11	M7	M0	M0	M28	M21	M18	M5
	4º	M1	M25	M11	M2	M11	M7	M0	M0	M16	M21	M29	M5
	5º	M0	M11	M1	M3	M1	M0	M7	M7	M16	M29	M29	M6
	6º	M0	M11	M0	M3	M0	M10	M7	M7	M0	M16	M5	M6
	7º	M11	M1	M0	M0	M0	M10	M11	M20	M0	M16	M5	M28
JUEVES	PH												
	1º	M2	M22	M10	M3	M8	M6	M25	M25	M0	M16	M17	M29
	2º	M2	M22	M10	M3	M8	M6	M25	M25	M7	M7	M17	M1
	3º	M0	M10	M2	M25	M22	M22	M11	M6	M28	M29	M18	M1
	4º	M22	M10	M2	M25	M22	M22	M11	M6	M28	M29	M7	M7
	5º	M22	M23	M22	M1	M11	M24	M7	M7	M29	M1	M28	M18
	6º	M3	M8	M7	M0	M0	M24	M22	M20	M21	M28	M5	M18
	7º	M3	M8	M7	M11	M24	M11	M22	M20	M21	M28	M5	M28
VIERNES	PH												
	1º	M25	M11	M24	M1	M2	M0	M1	M12	M28	M14	M7	M7
	2º	M25	M11	M24	M1	M2	M0	M1	M12	M28	M14	M7	M7
	3º	M11	M24	M1	M11	M25	M2	M6	M11	M7	M7	M28	M29
	4º	M11	M24	M1	M11	M25	M2	M6	M11	M7	M7	M28	M29
	5º	M24	M8	M0	M0	M23	M23	M11	M13	M14	M29	M5	M28
	6º	M24	M8	M0	M0	M11	M11	M2	M13	M14	M28	M1	M28
	7º	M22	M0	M11	M24	M11	M11	M2	M22	M29	M28	M1	M17

Figura IV.3: Horario de cursada del turno mañana para el año 2017 provisto por el programa desarrollado en este trabajo

DIA	HORA	CURSO 0	CURSO 1	CURSO 2	CURSO 3	CURSO 4	CURSO 5	CURSO 6	CURSO 7	CURSO 8	CURSO 9
LUNES	PH		M1		M0		M13	M4		M5	M18
	1º	M22	M1	M23	M22	M12	M13	M0	M0	M3	M5
	2º	M22	M0	M1	M22	M12	M20	M29	M0	M3	M5
	3º	M1	M11	M2	M24	M22	M0	M16	M29	M29	M1
	4º	M1	M11	M24	M2	M26	M0	M16	M29	M29	M1
	5º	M24	M2	M0	M10	M0	M12	M21	M21	M18	M29
	6º	M11	M2	M0	M10	M0	M12	M21	M21	M18	M29
	7º										
MARTES	PH		M10	M2	M11	M15					
	1º	M23	M10	M11	M0	M15	M11	M28	M1	M28	M17
	2º	M10	M25	M11	M0	M2	M1	M28	M1	M28	M17
	3º	M10	M25	M3	M1	M2	M1	M27	M0	M0	M28
	4º	M2	M6	M3	M1	M6	M22	M27	M14	M0	M28
	5º	M2	M6	M0	M23	M6	M22	M29	M28	M17	M0
	6º	M0	M22	M1	M2	M11	M6	M14	M28	M17	M29
	7º	M0	M22	M1	M2	M11	M6	M14	M29	M18	M29
MIERCOLES	PH	M3		M2	M11				M7	M29	
	1º	M3	M11	M2	M11	M11	M2	M0	M7	M5	M17
	2º	M1	M11	M11	M6	M11	M2	M0	M29	M5	M17
	3º	M25	M0	M25	M6	M26	M25	M14	M16	M6	M0
	4º	M25	M0	M25	M1	M26	M25	M14	M16	M6	M0
	5º	M11	M1	M0	M8	M1	M22	M1	M4	M28	M18
	6º	M11	M22	M22	M8	M1	M13	M1	M4	M28	M5
	7º		M22	M22			M13		M14		M28
JUEVES	PH	M22	M23			M26		M29		M0	
	1º	M22	M0	M6	M24	M0	M0	M29	M28	M1	M7
	2º	M3	M0	M6	M24	M0	M0	M16	M28	M1	M7
	3º	M3	M8	M24	M8	M25	M11	M3	M4	M18	M28
	4º	M2	M8	M24	M8	M25	M11	M3	M4	M9	M5
	5º	M24	M2	M3	M25	M12	M15	M28	M7	M9	M5
	6º	M24	M2	M3	M25	M20	M15	M28	M19	M3	M18
	7º	M0				M7	M7	M4	M19	M29	M18
VIERNES	PH	M2					M11				M6
	1º	M6	M24	M22	M11	M7	M7	M4	M27	M5	M6
	2º	M6	M24	M22	M11	M7	M7	M4	M27	M5	M9
	3º	M0	M24	M11	M0	M22	M20	M19	M16	M28	M9
	4º	M0	M11	M11	M0	M22	M20	M19	M28	M28	M7
	5º	M11	M8	M10	M2	M20	M12	M3	M28	M17	M28
	6º	M11	M8	M10	M22	M20	M12	M28	M14	M17	M28
	7º			M0	M22	M12		M28	M14		

Figura IV.4: Horario de cursada del turno tarde para el año 2017 provisto por el programa desarrollado en este trabajo

DIA	HORA	CURSO 0	CURSO 1	CURSO 2	CURSO 3	CURSO 4	CURSO 5	CURSO 6	CURSO 7	CURSO 8	CURSO 9	CURSO 10	CURSO 11
LUNES	PH												
	1º	M2	M1	M4	M23	M26	M4	M15	M23	M27	M1	M10	M31
	2º	M2	M1	M4	M23	M26	M4	M15	M23	M27	M1	M10	M31
	3º	M13	M4	M7	M1	M23	M23	M29	M16	M7	M21	M7	M1
	4º	M13	M4	M7	M2	M23	M23	M29	M16	M7	M21	M7	M1
	5º	M1	M2	M13	M4	M4	M7	M23	M1	M28	M0	M30	M20
	6º	M5	M2	M13	M5	M4	M7	M23	M15	M2	M4	M30	M20
	7º	M5	M13	M23	M5	M24	M2	M22	M15	M2	M4	M2	M30
MARTES	PH												
	1º	M12	M7	M26	M1	M1	M23	M15	M17	M1	M17	M20	M2
	2º	M12	M7	M26	M1	M1	M23	M15	M17	M1	M17	M20	M2
	3º	M1	M26	M23	M4	M9	M4	M1	M15	M28	M2	M1	M10
	4º	M1	M26	M23	M4	M9	M4	M1	M15	M28	M2	M2	M10
	5º	M26	M4	M2	M12	M4	M24	M29	M22	M21	M27	M30	M1
	6º	M23	M2	M4	M12	M4	M26	M29	M22	M21	M27	M31	M30
	7º	M23	M1	M4	M24	M2	M26	M23	M16	M22	M21	M31	M30
MIERCOLES	PH												
	1º	M1	M13	M8	M7	M23	M27	M27	M13	M22	M18	M19	M6
	2º	M1	M13	M8	M7	M23	M27	M27	M13	M22	M18	M19	M6
	3º	M4	M1	M27	M13	M7	M13	M8	M8	M18	M11	M1	M6
	4º	M4	M1	M27	M13	M7	M1	M8	M8	M18	M11	M1	M19
	5º	M23	M27	M13	M4	M13	M1	M13	M27	M17	M1	M6	M19
	6º	M13	M27	M1	M1	M13	M8	M13	M27	M17	M1	M6	M7
	7º	M13	M4	M1	M1	M1	M8	M2	M1	M3	M14	M6	M7
JUEVES	PH												
	1º	M7	M12	M8	M23	M1	M1	M13	M7	M8	M8	M20	M30
	2º	M7	M12	M8	M23	M1	M1	M13	M7	M8	M8	M20	M30
	3º	M4	M9	M23	M5	M9	M13	M1	M23	M4	M18	M30	M20
	4º	M4	M9	M1	M5	M9	M13	M1	M2	M4	M18	M30	M20
	5º	M5	M23	M12	M13	M26	M8	M22	M2	M18	M14	M8	M8
	6º	M5	M23	M12	M27	M13	M8	M22	M22	M18	M14	M31	M19
	7º	M23	M24	M24	M27	M2	M26	M8	M8	M21	M25	M31	M19
VIERNES	PH												
	1º	M26	M9	M13	M2	M13	M12	M7	M1	M11	M25	M6	M31
	2º	M26	M9	M13	M2	M13	M12	M7	M1	M11	M25	M6	M31
	3º	M27	M26	M2	M13	M12	M13	M2	M4	M1	M7	M30	M30
	4º	M27	M13	M26	M13	M12	M13	M17	M4	M1	M7	M19	M6
	5º	M24	M13	M1	M26	M27	M2	M17	M13	M8	M8	M19	M6
	6º	M2	M23	M1	M26	M27	M2	M4	M13	M3	M0	M8	M8
	7º	M13	M23	M2	M26	M2	M1	M4	M16	M3	M0	M8	M8

Figura IV.5: Horario de cursada del turno mañana elaborado manualmente y utilizado para el año 2018

DIA	HORA	CURSO 0	CURSO 1	CURSO 2	CURSO 3	CURSO 4	CURSO 5	CURSO 6	CURSO 7	CURSO 8	CURSO 9
LUNES	PH			M23	M1	M4	M16			M31	
	1º	M23	M2	M7	M1	M4	M16	M4	M2	M31	M30
	2º	M23	M2	M7	M7	M27	M15	M4	M2	M1	M1
	3º	M2	M1	M2	M7	M29	M15	M18	M1	M2	M31
	4º	M1	M13	M2	M26	M29	M1	M18	M1	M2	M31
	5º	M13	M4	M26	M23	M15	M13	M17	M7	M30	M6
	6º	M24	M4	M1	M24	M15	M13	M17	M7	M30	M6
	7º										
MARTES	PH		M2		M9	M29		M7	M0		
	1º	M2	M1	M4	M9	M2	M1	M7	M0	M30	M19
	2º	M2	M7	M4	M1	M2	M1	M3	M18	M30	M19
	3º	M12	M7	M5	M1	M7	M2	M28	M18	M20	M10
	4º	M12	M1	M5	M2	M7	M2	M11	M14	M20	M10
	5º	M4	M23	M1	M2	M13	M7	M11	M14	M19	M6
	6º	M4	M23	M1	M4	M13	M7	M2	M27	M19	M6
	7º				M4		M23	M2	M27		M6
MIERCOLES	PH	M5	M27	M23		M17		M5			
	1º	M5	M27	M4	M9	M17	M1	M5	M18	M6	M30
	2º	M13	M1	M4	M9	M23	M17	M5	M18	M6	M30
	3º	M13	M1	M13	M2	M23	M17	M28	M8	M30	M20
	4º	M1	M13	M13	M1	M13	M23	M28	M8	M30	M20
	5º	M1	M13	M2	M23	M13	M23	M22	M25	M1	M19
	6º	M27	M23	M1	M13	M1	M16	M22	M25	M5	M2
	7º	M27	M23	M1	M13	M1	M16	M22	M14	M5	M2
JUEVES	PH	M23	M12		M13			M27	M4		M19
	1º	M23	M12	M13	M26	M29	M8	M18	M4	M1	M1
	2º	M5	M24	M13	M26	M27	M8	M18	M21	M20	M1
	3º	M5	M4	M26	M12	M15	M13	M1	M25	M20	M8
	4º	M4	M4	M26	M12	M15	M13	M1	M8	M31	M20
	5º	M26	M9	M5	M27	M5	M22	M21	M17	M31	M20
	6º	M26	M9	M5	M27	M5	M15	M27	M17	M7	M31
	7º	M26				M5	M15			M7	M31
VIERNES	PH			M13					M1	M6	
	1º	M4	M26	M23	M13	M22	M8	M1	M1	M6	M7
	2º	M13	M26	M23	M13	M22	M4	M1	M21	M6	M7
	3º	M1	M26	M27	M4	M22	M4	M3	M21	M19	M8
	4º	M1	M13	M27	M4	M23	M22	M3	M0	M19	M8
	5º	M4	M13	M12	M23	M1	M22	M21	M11	M5	M30
	6º	M7	M9	M12	M23	M1	M27	M21	M11	M10	M30
	7º	M7	M9	M24			M27	M5		M10	M30

Figura IV.6: Horario de cursada del turno tarde elaborado manualmente y utilizado para el año 2018

DIA	HORA	CURSO 0	CURSO 1	CURSO 2	CURSO 3	CURSO 4	CURSO 5	CURSO 6	CURSO 7	CURSO 8	CURSO 9	CURSO 10	CURSO 11
LUNES	PH												
	1º	M1	M4	M13	M12	M4	M23	M29	M23	M4	M21	M6	M1
	2º	M1	M4	M13	M12	M4	M23	M29	M23	M4	M21	M6	M1
	3º	M13	M12	M7	M4	M26	M12	M27	M15	M7	M27	M7	M19
	4º	M13	M12	M7	M4	M26	M12	M27	M1	M7	M27	M7	M19
	5º	M23	M7	M24	M23	M23	M26	M15	M1	M28	M7	M19	M6
	6º	M2	M7	M1	M24	M23	M2	M23	M16	M2	M7	M19	M30
	7º	M2	M13	M4	M5	M2	M4	M23	M16	M2	M25	M20	M30
MARTES	PH												
	1º	M1	M2	M23	M26	M7	M8	M1	M15	M22	M17	M31	M20
	2º	M1	M2	M23	M26	M7	M8	M1	M15	M22	M17	M31	M20
	3º	M12	M26	M8	M2	M1	M23	M15	M4	M28	M2	M20	M19
	4º	M12	M26	M8	M2	M1	M23	M15	M4	M28	M2	M20	M1
	5º	M26	M1	M4	M23	M9	M4	M22	M22	M21	M1	M6	M31
	6º	M26	M27	M1	M1	M9	M4	M22	M27	M21	M4	M30	M6
	7º	M24	M27	M26	M1	M24	M1	M23	M27	M1	M4	M30	M6
MIERCOLES	PH												
	1º	M23	M23	M2	M7	M9	M1	M29	M7	M18	M18	M19	M31
	2º	M23	M23	M2	M7	M9	M1	M29	M7	M18	M18	M1	M31
	3º	M13	M4	M1	M5	M13	M13	M8	M8	M17	M11	M20	M10
	4º	M13	M4	M1	M5	M13	M13	M13	M1	M17	M11	M6	M10
	5º	M4	M13	M8	M4	M1	M2	M13	M13	M22	M1	M6	M19
	6º	M4	M13	M8	M4	M12	M2	M17	M22	M3	M14	M30	M7
	7º	M1	M2	M13	M13	M12	M8	M17	M22	M3	M14	M30	M7
JUEVES	PH												
	1º	M2	M1	M23	M13	M1	M24	M13	M17	M1	M14	M10	M6
	2º	M7	M1	M23	M13	M1	M13	M13	M17	M8	M8	M10	M6
	3º	M7	M9	M2	M23	M2	M26	M8	M8	M27	M18	M1	M20
	4º	M4	M9	M4	M23	M2	M26	M8	M8	M27	M18	M1	M20
	5º	M4	M23	M4	M1	M26	M8	M22	M13	M21	M0	M8	M8
	6º	M5	M23	M12	M27	M13	M7	M1	M13	M18	M1	M30	M30
	7º	M5	M9	M12	M27	M13	M7	M1	M15	M18	M1	M30	M30
VIERNES	PH												
	1º	M26	M1	M13	M2	M4	M27	M7	M13	M1	M25	M19	M31
	2º	M23	M1	M13	M26	M4	M27	M7	M2	M1	M25	M8	M8
	3º	M27	M26	M27	M5	M13	M13	M4	M2	M11	M0	M8	M8
	4º	M27	M9	M27	M13	M27	M13	M4	M1	M11	M0	M31	M2
	5º	M5	M13	M26	M13	M27	M1	M15	M23	M8	M8	M31	M2
	6º	M5	M13	M26	M1	M23	M1	M2	M16	M8	M8	M2	M30
	7º	M13	M24	M1	M1	M23	M4	M2	M16	M3	M21	M2	M30

Figura IV.7: Horario de cursada del turno mañana para el año 2018 provisto por el programa desarrollado en este trabajo

DIA	HORA	CURSO 0	CURSO 1	CURSO 2	CURSO 3	CURSO 4	CURSO 5	CURSO 6	CURSO 7	CURSO 8	CURSO 9
LUNES	PH	M26		M7	M1	M17					
	1º	M26	M2	M7	M7	M17	M1	M27	M27	M30	M2
	2º	M2	M1	M26	M7	M15	M1	M27	M27	M30	M2
	3º	M23	M13	M1	M26	M1	M17	M2	M4	M5	M20
	4º	M23	M13	M1	M23	M1	M17	M2	M4	M5	M20
	5º	M13	M4	M23	M4	M27	M4	M18	M18	M1	M6
	6º	M24	M4	M23	M4	M27	M4	M18	M18	M31	M6
	7º										
MARTES	PH	M7				M29	M15	M5			
	1º	M4	M1	M4	M9	M7	M15	M7	M1	M6	M19
	2º	M4	M1	M4	M9	M7	M13	M7	M1	M31	M19
	3º	M12	M12	M5	M1	M4	M7	M22	M2	M30	M10
	4º	M12	M12	M5	M1	M4	M7	M22	M2	M30	M10
	5º	M5	M23	M1	M2	M13	M16	M1	M0	M19	M20
	6º	M5	M23	M2	M4	M13	M16	M17	M0	M19	M20
	7º			M2	M4		M23	M17	M14		M30
MIERCOLES	PH			M2	M13	M5	M27			M6	M1
	1º	M27	M27	M4	M13	M29	M1	M3	M1	M7	M30
	2º	M27	M27	M4	M1	M29	M1	M1	M1	M7	M30
	3º	M1	M2	M1	M1	M13	M23	M11	M8	M20	M31
	4º	M1	M2	M1	M23	M13	M23	M11	M8	M20	M19
	5º	M13	M23	M13	M24	M23	M16	M22	M18	M30	M19
	6º	M13	M23	M13	M2	M23	M16	M28	M18	M30	M6
	7º		M13	M24	M2		M13	M28	M25		
JUEVES	PH	M23	M4	M26				M18	M8	M2	
	1º	M23	M4	M26	M9	M15	M8	M18	M0	M2	M30
	2º	M26	M24	M13	M9	M15	M8	M28	M11	M6	M30
	3º	M4	M1	M27	M13	M22	M13	M21	M11	M20	M31
	4º	M4	M1	M27	M13	M22	M13	M21	M25	M20	M8
	5º	M5	M9	M5	M26	M2	M22	M1	M25	M31	M6
	6º	M5	M9	M5	M26	M2	M22	M1	M21	M31	M6
	7º	M1				M29			M21		
VIERNES	PH		M26			M23		M5		M6	M7
	1º	M1	M26	M12	M23	M22	M2	M5	M7	M6	M7
	2º	M1	M26	M12	M23	M5	M2	M21	M7	M10	M31
	3º	M13	M7	M23	M13	M5	M22	M3	M17	M19	M31
	4º	M4	M7	M23	M27	M1	M15	M3	M17	M19	M8
	5º	M2	M13	M13	M27	M1	M15	M4	M14	M1	M8
	6º	M2	M13	M13	M12	M15	M8	M4	M14	M1	M1
	7º		M9		M12				M21	M5	M1

Figura IV.8: Horario de cursada del turno tarde para el año 2018 provisto por el programa desarrollado en este trabajo

Bibliografía

[1] Csima J. "Investigations on a Time-Table Problem". Tesis doct. Institute of Computer Science, University of Toronto, 1965.

[2] Bondy J.A. and Murty U.S.R. *Graph theory with applications*. North-Holland, 1976.

[3] de Werra, D. "Graphs, hypergraphs and timetabling". En: *Methods of Operations Research* 49 (1985), págs. 201-213.

[4] Carter, M. "A lagrangian relaxation approach to theclassroom assignment problem". En: *INFOR* 27 (1986), págs. 230-246.

[5] Deris, B., Omatu, S., Ohta, H., Samat, D. "University timetabling by constraint-based reasoning: A case study". En: *Journal of Operational Research Society* 12.48 (1997), págs. 1178-1190.

[6] Nonobe, K., Ibaraki, T. "A tabu search approach to the constraint satisfaction problem as a general problem solver". En: *European Journal of Operational Research* 106 (1998), págs. 599-623.

[7] Abramson, D., Krishnamoorthy, M., Dang, H., "Simulated annealing cooling schedules for the school timetabling problem". En: *Asia-Pacific Journal of Operational Research* 16 (1999).

[8] Schaerf, A. "A survey of automated timetabling". En: *Artificial Intelligence Review* 13 (1999), págs. 87-127.

[9] Di Gaspero, L., Schaerf, A. "Tabu search techniques for examination timetabling". En: *Selected Papers from the 3rd International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science*. Vol. 2079. Springer, 2001, págs. 104-117.

[10] Zervoudakis K. "A Generic Object-Oriented Constraint Based Model for University Course Timetabling". En: *Proc. of the 3rd International Conference on the Practice and Theory of Automated Timetabling PATAT 2000*. Vol. 2079. Springer-Verlag, 2001, págs. 28-47.

[11] Cheang, B., Li, H., Lim, A., Rodrigues, B. "Nurse rostering problems: A bibliographic survey". En: *European Journal of Operational Research* 151 (2003), págs. 447-460.

[12] Stuart Russell y Peter Norvig. *Artificial Intelligence: a Modern Approach*. 2.ª ed. Pearson Education, 2003. ISBN: 0-13-790395-2.

[13] Daskalaki, S., Birbas, T., Housos, E. "An integer programming formulation for a case study in university timetabling". En: *European Journal of Operations Research* 1.153 (2004), págs. 117-135.

[14] Easton, K., Nemhauser, G., Trick, M. "Sports scheduling". En: *Handbook of Scheduling: Algorithms,Models, and Performance Analysis*. Ed. por J. Leung. CRC Press, 2004. Cap. 52.

[15] Kwan, R. "Bus and train driver scheduling". En: *Handbook of Scheduling: Algorithms,Models, and Performance Analysis*. Ed. por J. Leung. CRC Press, 2004. Cap. 51.

[16] Petrovic, S., Burke, E.K., "University timetabling". En: *Handbook of Scheduling: Algorithms,Models, and Performance Analysis*. Ed. por J. Leung. CRC Press, 2004. Cap. 45.

[17] Zbigniew Michalewicz y David B. Fogel. *How to Solve It: Modern Heuristics*. 2.ª ed. Springer, 2004. ISBN: 3-540-22494-7.

[18] Côte , P., Wong, T., Sabourin, R. "Application of a hybrid multi-objective evolutionary algorithm to the uncapacitated exam proximity problem". En: *Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science*. Vol. 3616. Springer, 2005, págs. 294-312.

[19] McCollum, B. "A Perspective on Bridging the Gap Between Theory and Practice in University Timetabling". En: *PATAT 2006, Lecture Notes in Computer Science*. Ed. por H. Rudová E.K. Burke. Vol. 3867. 2006, págs. 3-23.

[20] Murray, K., Müller, T., Rudová, H. "Modeling and Solution of a Complex University Course Timetabling Problem". En: *Lecture Notes in Computer Science* 3867 (2006), págs. 189-209.

[21] Burke E. K., McCollum B., Miesels A., Petrovic S., Qu R. "A graph-based hyper-heuristic for educational timetabling problems". En: *European Journal of Operational Research* 176 (2007), págs. 177-192.

[22] Durán G., Guajardo M., Miranda J., Sauré D., Souyris S., Weintraub A., Wolf R. "Scheduling the Chilean Soccer League by Integer Programming". En: *INTERFACES* 37 (2007), págs. 539-552.

[23] Chorbev I., Loskovska S., Dimitrovski I., Mihajlov D. "Solving the High School Scheduling Problem Modelled with Constraints Satisfaction Using Hybrid Heuristic Algorithms". En: *Greedy Algorithms*. Ed. por Witold Bednorz. InTech, 2008. Cap. 28.

[24] McCollum, B., Schaerf, A., Paechter, B., McMullan, P., Lewis, R., Parkes, A., Di Gaspero, L., Qu, R., Burke, E. "Setting the Research Agenda in Automated Timetabling: The Second International Timetabling Competition". En: *INFORMS Journal on Computing* 1.22 (2010), págs. 120-130.

[25] Valouxis C., Gogos C., Alefragis P., Housos H. "Decomposing the High School Timetable Problem". En: *Practice and Theory of Automated Timetabling* (Agosto de 2012), págs. 29-31.

[26] van der Kooy N. J. "The High School Scheduling Problem: Improving Local Search Fairness Evaluation". Tesis de mtría. Mathematical Institute, University of Leiden, 2017.

[27] Gobierno de la Ciudad de Buenos Aires. *La Escuela que Queremos, Profundización de la NES*. `http://www.buenosaires.gob.ar/sites/gcaba/files/la_escuela_que_queremos.pdf`. [Online; accedido el 22 de Mayo de 2018].

[28] Gobierno de la Ciudad de Buenos Aires. *Nueva Escuela Secundaria (NES)*. `http://www.buenosaires.gob.ar/educacion/escuelas/nueva-escuela-secundaria`. [Online; accedido el 22 de Mayo de 2018].