



UNIVERSIDAD
DE LA REPÚBLICA
URUGUAY



FACULTAD DE
INGENIERÍA

Generación y valoración de calendarios de evaluaciones en Facultad de Ingeniería

Informe de Proyecto de Grado presentado por

Florencia Carle Vitale, Lucas De Sierra Gandolfo,
Facundo Spira Rivas

en cumplimiento parcial de los requerimientos para la graduación de la carrera
de Ingeniería en Computación de Facultad de Ingeniería de la Universidad de
la República

Supervisores

Libertad Tansini
Carlos Testuri

Usuaría Responsable

Ema Soboredo

Montevideo, 11 de junio de 2025



Generación y valoración de calendarios de evaluaciones en Facultad de Ingeniería por Florencia Carle Vitale, Lucas De Sierra Gandolfo, Facundo Spira Rivas tiene licencia [CC Atribución 4.0](#).

Agradecimientos

Queremos expresar nuestro más sincero agradecimiento a todas las personas e instituciones que de una u otra forma contribuyeron al desarrollo y culminación de este Proyecto de Grado.

En primer lugar, agradecemos a la Facultad de Ingeniería de la Universidad de la República por brindarnos el marco académico y la oportunidad de realizar este trabajo, contribuyendo así a nuestro crecimiento profesional.

Agradecemos especialmente a Ema Soboredo, Asistente Académica de la Facultad, quien actuó como usuaria de nuestra tesis. Su disposición, compromiso y aportes fueron esenciales para la evolución del proyecto.

Nuestro reconocimiento y gratitud a nuestros tutores, Libertad Tansini y Carlos Testuri, por su acompañamiento, orientación y valiosos aportes a lo largo de todo el proceso. Su experiencia y dedicación resultaron claves para alcanzar los objetivos planteados.

Finalmente, cada uno de los integrantes desea agradecer de forma personal.

“Agradezco a mi familia, que fue mi mayor fuente de apoyo y motivación a lo largo de este camino. A mi madre por su amor incondicional, su fuerza y compañía constante. A mi padre por confiar siempre en mí y animarme a seguir adelante. A mi hermano por tantos años de crecimiento compartidos. A mis abuelos, cuya memoria y cariño siguen acompañándome día a día. Y a mis amigos, que hicieron este trayecto más liviano con su presencia y apoyo en los buenos, y por sobre todo, malos momentos.” (Facundo Spira).

“Agradezco a mi familia por el apoyo incondicional a lo largo de la carrera, a mi padre por acompañarme y saber transmitirme su característica tranquilidad, y en especial a mi madre quien fue mi compañía constante y supo compartir mis alegrías pero sobre todo tolerar mis frustraciones. A Santi, por el amor, el apoyo y los incontables consejos a lo largo de mi formación académica. Por último, pero no menos importante, a mis amigos por siempre estar ahí y ser mi fan número 1.” (Florencia Carle).

“Agradezco a mi familia por todo el apoyo en esta etapa. A mi madre por su amor y por acompañarme en los momentos más lindos y en los más difíciles también. A mi padre y mis hermanos por el soporte y motivación constantes. Y a mis amigos por hacer que todo sea más llevadero con su compañía.” (Lucas de Sierra).

A todos ellos, nuestro más profundo agradecimiento.

Resumen

La programación de calendarios de evaluaciones representa un desafío logístico significativo para las instituciones educativas de nivel superior, donde se deben equilibrar múltiples restricciones y objetivos contrapuestos. En la Facultad de Ingeniería de la Universidad de la República, este problema implica asignar a aproximadamente 80 unidades curriculares distribuidas en nueve carreras diferentes un turno específico en un día determinado del período de evaluaciones, considerando el aforo de los salones, las relaciones entre carreras y previaturas de las unidades curriculares, pares de unidades con cuerpo docente compartido y limitaciones de turnos y días según unidades curriculares, y con los objetivos de minimizar la cantidad de estudiantes con coincidencia de evaluaciones en el día o en días consecutivos y el intervalo entre unidades curriculares que no tienen dependencia en previas. Este problema, que pertenece a la clase de problemas NP Difícil, implica una elevada complejidad computacional al buscar soluciones óptimas. Este proyecto aborda esta problemática mediante el desarrollo de un sistema automatizado para la generación de calendarios de evaluaciones utilizando técnicas de programación lineal entera mixta. El trabajo comienza con un estudio exhaustivo de los fundamentos teóricos de la optimización matemática y las herramientas computacionales disponibles para la resolución de este tipo de problemas, así como una revisión sistemática de antecedentes relacionados con la optimización de horarios en contextos educativos. Para resolver el problema, se desarrolla un modelo matemático que incorpora aspectos críticos del mismo. La implementación computacional se realiza utilizando Python y la biblioteca PuLP, empleando Gurobi como solver para determinar soluciones óptimas o factibles con certificado de brecha de optimalidad. Para evaluar la calidad de las soluciones generadas, se diseñan métricas específicas que cuantifican la cantidad promedio de estudiantes afectados, la coincidencia promedio total entre evaluaciones, la coherencia en la distribución de previas y la uniformidad en la trayectoria sugerida. Se realizan experimentos con cuatro períodos de evaluaciones del año 2024, y los resultados obtenidos se comparan con los calendarios oficiales, evidenciando que el sistema propuesto logra, en varios casos, mejorar, de acuerdo con las métricas diseñadas, el calendario oficial. El sistema desarrollado constituye una herramienta de apoyo a la toma de decisiones que permite generar calendarios de calidad en un tiempo menor al requerido por el proceso manual, liberando al responsable para que pueda concentrarse en aspectos cualitativos de la planificación y en ajustes específicos según necesidades particulares de cada evaluación, no incluidas en los requerimientos del sistema.

Palabras clave: Generación de Calendarios, Programación Lineal Entera Mixta, Optimización, Programación Matemática.

Índice general

1. Introducción	1
2. Marco teórico	5
2.1. Optimización matemática	5
2.2. Solvers y lenguajes algebraicos	6
3. Revisión de antecedentes	9
3.1. Proceso de búsqueda de antecedentes	9
3.2. Antecedentes	12
3.2.1. Optimización de horarios	13
3.2.2. Optimización de flujos	17
4. Análisis del problema	19
4.1. Definición del problema	19
4.2. Relevamiento de requisitos	21
4.2.1. Requisitos funcionales	22
5. Definición del modelo matemático	23
5.1. Conjuntos	24
5.2. Parámetros	25

5.3. Variables de decisión	26
5.4. Restricciones	27
5.5. Función objetivo	29
5.5.1. Distribución temporal de evaluaciones	29
5.5.2. Incorporación de previas	30
6. Implementación de la solución	33
6.1. Tecnologías utilizadas	33
6.1.1. Modelador de programación matemática PuLP	33
6.1.2. Otras librerías	34
6.2. Arquitectura de la solución	35
6.2.1. Procesamiento de datos	36
6.2.2. Carga de datos	40
6.2.3. Definición del modelo	41
6.2.4. Generación del calendario	43
7. Métricas de calidad de solución	45
7.1. Métrica de estudiantes afectados	45
7.2. Métrica de coincidencia	46
7.3. Métrica de previas	47
7.4. Métrica de trayectoria sugerida	48
8. Resultados experimentales y análisis	51
8.1. Casos de prueba y hardware utilizado	51
8.2. Selección del solver	53
8.3. Calibración de los parámetros α y β	53
8.4. Experimentaciones de tiempo extendido y análisis de resultados	55

8.4.1. Incorporación de mejoras a partir de observaciones cualitativas	57
8.4.2. Análisis cualitativo del calendario generado	60
9. Conclusiones y trabajo futuro	61
9.1. Principales logros y contribuciones	61
9.2. Limitaciones identificadas	62
9.3. Trabajo futuro	63
A. Categorización de la literatura revisada	71
B. Calendarios generados	73
B.1. Calendario 1s1p	74
B.2. Calendario 1s2p	75
B.3. Calendario 2s1p	76
B.4. Calendario 2s2p	77
B.5. Calendario 1s1p revisado	78

Capítulo 1

Introducción

La programación de calendarios de evaluaciones en instituciones educativas de nivel superior representa un desafío logístico significativo que implica equilibrar múltiples restricciones y objetivos contrapuestos, así como se puede observar en el relevamiento bibliográfico de estudios anteriores que se presenta en detalle en el Capítulo 3. En este contexto, un calendario efectivo debe satisfacer diversas condiciones, como respetar la capacidad física de los salones disponibles, considerar las limitaciones de personal para supervisión, evitar coincidencias temporales para estudiantes que cursan determinadas combinaciones de unidades curriculares (UC) y, fundamentalmente, distribuir las evaluaciones de manera que los estudiantes puedan rendir las evaluaciones de las UC que cursan, evitando superposiciones.

En la Facultad de Ingeniería (FIng) de la Universidad de la República (Ude-laR), la complejidad de esta tarea aumenta debido a la escala del problema: existen aproximadamente 80 UC distribuidas en nueve carreras diferentes, cada una con su propia estructura curricular y sistema de previaturas (**Facultad de Ingeniería, 2025**). Esta dimensión convierte la generación manual de calendarios en una labor compleja, donde la consideración simultánea de todas las restricciones y la optimización de múltiples criterios se vuelve una tarea objetivamente difícil. Esto causa que en cada período de evaluaciones se presenten reclamos por parte de estudiantes afectados por superposiciones o proximidades inadecuadas entre evaluaciones. La resolución de estas situaciones particulares resulta problemática, ya que cualquier modificación puntual puede desencadenar nuevos conflictos en otras áreas del calendario, generando un efecto dominó de complicaciones.

Actualmente, la responsabilidad de confeccionar estos calendarios recae en Ema Soboredo, Asistente Académica de Decanato, quien desarrolla este siguiendo un proceso manual mediante hojas de cálculo. Su metodología consiste en

asignar secuencialmente las evaluaciones por carrera, intentando evitar superposiciones evidentes, pero sin contar con herramientas que le permitan visualizar o ponderar el impacto global de cada decisión. Este enfoque, aunque respaldado por la experiencia y el criterio profesional, depende excesivamente de conocimientos basados en la práctica y la intuición, y carece de mecanismos sistemáticos para optimizar aspectos críticos como la distribución temporal de evaluaciones para beneficio de los estudiantes.

El objetivo general de este proyecto es desarrollar un sistema automatizado de programación de calendarios de evaluaciones basado en técnicas de programación lineal entera mixta (Chvátal, 1999). Este sistema busca generar soluciones que iguallen o superen la calidad de los calendarios elaborados manualmente, pero en menos tiempo que el requerido actualmente. La finalidad no es reemplazar completamente el criterio humano, sino proporcionar una herramienta de apoyo a la decisión que permita concentrarse en los ajustes finos y consideraciones cualitativas del calendario, liberando de la difícil tarea inicial de ubicación de evaluaciones bajo restricciones básicas.

Para alcanzar este objetivo general, el proyecto contempla varios objetivos específicos interrelacionados. Por un lado, se busca construir un marco teórico sólido sobre programación lineal entera mixta y su aplicación a problemas de optimización combinatoria en contextos educativos, estableciendo así los fundamentos matemáticos y conceptuales del trabajo. De forma paralela, se realiza una revisión sistemática de la literatura para identificar enfoques, modelos y técnicas aplicadas en problemas similares de programación de calendarios académicos, identificando antecedentes y soluciones existentes. Sustentado en estas bases, se desarrolla un modelo matemático que representa fielmente, de acuerdo a los criterios de Ema, las restricciones institucionales y objetivos de optimización relevantes para la FIng, el cual se implementa computacionalmente utilizando el lenguaje Python y la biblioteca PuLP para la formulación y resolución del problema. Complementariamente, se diseñan métricas cuantitativas que permiten evaluar objetivamente la calidad de los calendarios generados, facilitando la comparación entre diferentes soluciones y con los calendarios históricos. Finalmente, se realiza una experimentación sistemática para calibrar los parámetros del modelo, evaluar su desempeño bajo diferentes escenarios y validar su efectividad mediante la comparación con calendarios reales.

La relevancia de este proyecto trasciende la mera automatización de un proceso administrativo. Un calendario de evaluaciones bien diseñado impacta directamente en el rendimiento académico de los estudiantes, en la eficiencia operativa de la institución y en la satisfacción general de la comunidad universitaria. La organización y distribución adecuada de las evaluaciones no solo reduce el estrés y la ansiedad estudiantil, sino que también facilita una mejor gestión del tiempo y promueve un ambiente académico más saludable (Misra y Mckean, 2000). Mediante la aplicación de técnicas matemáticas avanzadas a este problema práctico, se busca contribuir a la mejora de procesos institucionales.

Los siguientes capítulos abordan en detalle los distintos aspectos de este trabajo: el marco teórico en el Capítulo 2, la revisión de antecedentes relevantes en el Capítulo 3, el análisis detallado del problema en el Capítulo 4, la definición del modelo matemático propuesto en el Capítulo 5, su implementación computacional en el Capítulo 6, las métricas de calidad de solución en el Capítulo 7, los resultados experimentales y su análisis en el Capítulo 8, y finalmente, las conclusiones y el trabajo futuro en el Capítulo 9.

Capítulo 2

Marco teórico

En este capítulo se presentan los conceptos fundamentales de la optimización matemática, así como las principales herramientas computacionales utilizadas para resolver este tipo de problemas. Se introducen las nociones básicas que permiten formular y analizar modelos de optimización, y se describen los componentes y el funcionamiento general de los solucionadores, también conocidos por su nombre en inglés, solvers, que constituyen la base tecnológica para su resolución. El objetivo es establecer un marco teórico que sirva de sustento para los desarrollos presentados en los capítulos siguientes.

2.1. Optimización matemática

La optimización matemática se define como el proceso de identificar la mejor solución posible dentro de un conjunto de alternativas factibles, en particular de maximizar o minimizar una función objetivo sobre variables cuyo dominio factible de soluciones se establece mediante restricciones algebraicas. (Vasily, 2019).

Los problemas de optimización se pueden clasificar en varias categorías según la naturaleza de la función objetivo y las restricciones, así como el tipo de variables de decisión involucradas. Una distinción importante se da entre la programación lineal (LP) y la programación no lineal, dependiendo de si las especificaciones de las variables son lineales o no. Otra clasificación relevante es entre la optimización continua, donde las variables pueden tomar cualquier valor real dentro de un rango dado, y la optimización entera, donde las variables están restringidas a valores enteros. Dentro de la optimización entera, se encuentra la programación entera (IP), donde todas las variables deben ser números enteros,

y la programación entera mixta (MIP), donde algunas variables son continuas y otras enteras.

Matemáticamente podemos definir un problema de programación entera mixta de la siguiente forma (Chvátal, 1999; J. Lee, 2024),

$$\left\{ \begin{array}{l} \text{mín} \quad \sum_{j=1}^n c_j x_j \\ \text{s.a} \quad \sum_{j=1}^n a_{i,j} x_j = b_i, \quad i = 1, 2, \dots, m, \\ x_j \geq 0, \quad j = 1, 2, \dots, n, \\ x_j \in \mathbb{Z}, \quad \text{para algún } j \in \{1, 2, \dots, n\}. \end{array} \right. \quad (2.1)$$

2.2. Solvers y lenguajes algebraicos

Un solver de programación entera mixta (MIP, por Mixed Integer Programming) resuelve un problema siguiendo un proceso que combina técnicas de optimización continua y estrategias discretas para manejar las variables enteras. En general, se comienza tratando de ignorar las restricciones enteras: el solver formula y resuelve una relajación del problema, donde a las variables originalmente enteras se les permite tomar cualquier valor continuo dentro de sus dominios permitidos. Esto transforma el MIP en un problema de programación lineal (LP) más sencillo, que puede ser resuelto rápidamente utilizando métodos como el simplex o algoritmos de punto interior. La solución de esta relajación proporciona dos o tres informaciones cruciales: un valor de referencia, un límite inferior en el caso de minimización (o superior en maximización), y, a veces, una idea de qué tan “cerca” se está de una solución factible para el problema original. Este límite proviene directamente del teorema de la dualidad de la programación lineal, el cual establece que, bajo ciertas condiciones, el valor óptimo del problema primal (el original) coincide con el valor óptimo de su problema dual. Así, la relajación continua no solo entrega una solución candidata, sino también un certificado matemático de la mejor cota posible que se puede llegar a obtener. Si, por casualidad, las variables enteras ya toman valores enteros en esta primera relajación, el problema se resuelve inmediatamente. Sin embargo, en la mayoría de los casos esto no sucede, y entonces el solver debe recurrir a técnicas más complejas para la búsqueda de soluciones en el espacio de soluciones factibles (Gurobi, 2024, 2025).

Una de las metodologías más utilizadas es “branch and bound”; para llevarlo a cabo, el solver divide el problema original en subproblemas más pequeños (ramificación), donde en cada uno se imponen restricciones adicionales para forzar

que ciertas variables enteras se acerquen a su valor deseado. Por ejemplo, si una variable debería ser 0 o 1 y la relajación sugiere 0.6, el solver crea dos subproblemas: uno donde se impone que la variable sea menor o igual a 0 y otro donde sea mayor o igual a 1. Cada uno de estos subproblemas es resuelto, de nuevo, como un problema continuo. Si un subproblema da una solución mejor que cualquier otra conocida y cumple las restricciones de integralidad, se actualiza lo que se conoce como incumbente, que es la mejor solución factible encontrada hasta el momento. A partir de este concepto del incumbente, el solver puede comparar los resultados de nuevos subproblemas: si la solución de un subproblema es peor que el valor del incumbente, o si su relajación resulta infactible, ese subproblema es descartado (poda), ya que no puede conducir a una mejor solución. Este proceso de ramificación y poda genera un árbol de decisiones que puede volverse muy grande, por lo que los solvers modernos implementan estrategias de selección inteligentes para decidir en qué orden explorar los nodos del árbol y cuándo abandonar ramas completas. Además, durante el proceso, se evalúa el gap de optimalidad, que mide la diferencia relativa entre el valor del incumbente y el mejor límite conocido (el de la relajación más prometedora aún abierta). Este gap se calcula típicamente como una proporción entre la diferencia y el valor del incumbente, y permite medir cuán cerca se está de garantizar la optimalidad de la solución actual. Un gap de optimalidad de cero implica que el incumbente encontrado es efectivamente óptimo. Los solvers pueden configurarse para detenerse automáticamente cuando el gap cae por debajo de un umbral deseado, aceptando una solución “suficientemente buena” sin necesidad de explorar todo el árbol (Gurobi, 2025).

Para mejorar la eficiencia, los solvers incorporan técnicas adicionales como la generación de cortes (cutting planes), que son restricciones válidas para el conjunto de soluciones enteras, pero que eliminan porciones de la relajación continua que no contienen soluciones factibles. Estas técnicas, como los cortes de Gomory, los cortes de cobertura o los cortes de flujo, refinan progresivamente la relajación para que se aproxime mejor al conjunto original de soluciones enteras. La combinación de cortes dinámicos con la exploración de subproblemas da lugar a estrategias conocidas como “branch and cut” (M. K. Lee, 2024).

Además, durante la búsqueda, se aplican heurísticas de factibilización, procedimientos diseñados para encontrar rápidamente soluciones factibles, aunque no necesariamente óptimas, que pueden mejorar el incumbente temprano en el proceso. Contar con un buen incumbente desde etapas tempranas acelera significativamente la poda de ramas, al estrechar los criterios para descartar subproblemas (Gurobi, 2025).

Finalmente, el proceso de resolución se detiene cuando se alcanza alguna condición de parada predefinida. Una de ellas es la exploración completa del árbol de búsqueda, lo que permite concluir que se ha encontrado la mejor solución factible o que el problema es infactible o ilimitado. Sin embargo, los solvers también permiten definir criterios de terminación alternativos que buscan equi-

librar precisión y eficiencia computacional. Por ejemplo, se puede establecer un tiempo límite de ejecución, tras el cual el solver interrumpe la búsqueda y reporta la mejor solución encontrada hasta ese momento. Otra condición común es fijar un valor mínimo aceptable para el gap de optimalidad, conocido como *minGap*, de modo que si la diferencia relativa entre el incumbente y el mejor límite conocido cae por debajo de ese umbral, el solver se detiene considerando que la solución actual es suficientemente buena. En todos los casos, el reporte final incluye no solo la solución hallada y su valor objetivo, sino también el gap de optimalidad alcanzado, que representa cuán lejos se encuentra dicha solución del mejor valor que podría obtenerse si se resolviera exactamente el problema original. Este valor ideal está acotado por la relajación continua, y en muchos casos puede no ser alcanzable si no existe una solución factible entera que logre dicho óptimo (Gurobi, 2024).

Así, los diferentes solvers suelen proveer en adición un lenguaje de modelado algebraico (AML), como GLPK, que admite el lenguaje de modelado GNU MathProg, el cual es un subconjunto del lenguaje AMPL. Los AML permiten formular problemas de optimización en una notación cercana a la matemática original, abstrayendo los detalles de implementación y facilitando la comunicación entre especialistas en diferentes áreas. Estos lenguajes de modelado algebraico ofrecen sintaxis declarativas que representan directamente las estructuras matemáticas del problema, como conjuntos, índices, parámetros, variables, restricciones y funciones objetivo. Sin embargo, también existen lenguajes de dominio específico embebidos (DSLs por sus siglas en inglés) que permiten solucionar problemas de optimización dentro de lenguajes de programación de propósito general. Un ejemplo destacado es PuLP, un lenguaje embebido en Python que proporciona una interfaz intuitiva para formular modelos de programación lineal y entera mixta. PuLP permite aprovechar todas las capacidades del ecosistema Python (desde análisis de datos hasta visualización) mientras mantiene la claridad en la expresión de los modelos matemáticos (Free Software Foundation, 2025; Fourer y cols., 1990; Wikipedia contributors, 2025).

Capítulo 3

Revisión de antecedentes

En este capítulo se presenta una revisión exhaustiva y sistemática de los antecedentes relacionados con la optimización de horarios en contextos educativos. Se describe detalladamente el proceso metodológico utilizado para la búsqueda y selección de material bibliográfico relevante, incluyendo la utilización de repositorios institucionales y bases de datos académicas. Los trabajos seleccionados se analizan y clasifican según el problema que abordan y las metodologías empleadas, permitiendo identificar las diferentes aproximaciones y técnicas utilizadas en la literatura para resolver problemas similares.

3.1. Proceso de búsqueda de antecedentes

El proceso de búsqueda de antecedentes se divide en varias etapas. En primer lugar, se realiza una búsqueda en Colibrí, el repositorio institucional de la Universidad de la República, el cual almacena y publica trabajos académicos de estudiantes y docentes. Para la búsqueda, se utiliza particularmente el repositorio de la FIng (*Colibrí: Facultad de Ingeniería*, 2014), que contiene tesis y documentos técnicos relacionados con diversas áreas de estudio dentro de la ingeniería. Esto es una primera aproximación, con el objetivo de comenzar a conocer sobre el tema y analizar el material disponible. De entre los encontrados, se seleccionan tres tesis a utilizar en el análisis de antecedentes.

Posteriormente, se realiza una búsqueda en Timbó, una plataforma que ofrece acceso a recursos científicos y tecnológicos para instituciones en Uruguay (*Buscador Timbó Foco*, 2009). El proceso de búsqueda toma algunos elementos de la búsqueda sistemática, que es una metodología rigurosa y auditable diseñada para evaluar e interpretar toda la investigación disponible sobre una

pregunta o tema específico. Este tipo de búsqueda sigue un protocolo definido, empleando estrategias de búsqueda exhaustivas y criterios explícitos de inclusión y exclusión para garantizar que la revisión sea completa y objetiva (Kitchenham, 2004). Utilizando principalmente las palabras clave y los resúmenes de las tesis seleccionadas en Colibrí, se arma un *string* de búsqueda para utilizar en Timbó. El *string* de búsqueda surge de un proceso iterativo de modificaciones del *string* y limitadores/filtros dentro de la plataforma hasta que se obtiene un número razonable de materiales para analizar. La versión final del *string* de búsqueda se separa en dos partes; por un lado *búsqueda* representa el *string* de búsqueda en sí, mientras que *limitadores* son los filtros que se aplican dentro de la plataforma para reducir los resultados que se obtienen. El uso de estos filtros permite acotar la búsqueda por temáticas, así como también por lenguajes o rangos de fechas. Finalmente, se obtiene el siguiente *string*:

Búsqueda

```
integer programming university timetable timetabling [AB] OR
integer programming university timetable timetabling optimization [TX]
NOT genetic [AB]
```

Limitadores

- **Lenguaje:** “english”, “spanish”
- **Fecha:** 2010 - 2024
- **Materia:** “mathematical optimization”, “scheduling”, “timetabling”, “mixed integer linear programming”, “algorithms”, “mathematical models”, “linear programming”, “integer programming”

El string de búsqueda utiliza una sintaxis específica que permite delimitar los términos a ciertas partes de los documentos dentro de la plataforma Timbó. En particular, las etiquetas [AB] y [TX] tienen un significado específico. [AB] indica que los términos precedidos por esta etiqueta se buscarán únicamente en el campo de resumen (abstract) de los documentos, lo que permite priorizar resultados cuyo tema principal esté relacionado con los términos de búsqueda. Por otro lado, [TX] indica que los términos precedidos por esta etiqueta se buscarán en el texto completo (full text) de los documentos, incluyendo el cuerpo principal, tablas, figuras y otros elementos. Esto permite ampliar el alcance de la búsqueda para incluir documentos que mencionen los términos en cualquier parte del contenido. Por ejemplo, en el string presentado, la búsqueda de los términos *integer programming university timetable timetabling* en el campo [AB] se enfoca en documentos cuyo resumen mencione explícitamente estos conceptos, mientras que la búsqueda en [TX] amplía el alcance para incluir documentos que los mencionen en cualquier parte del texto. Además, el operador NOT excluye

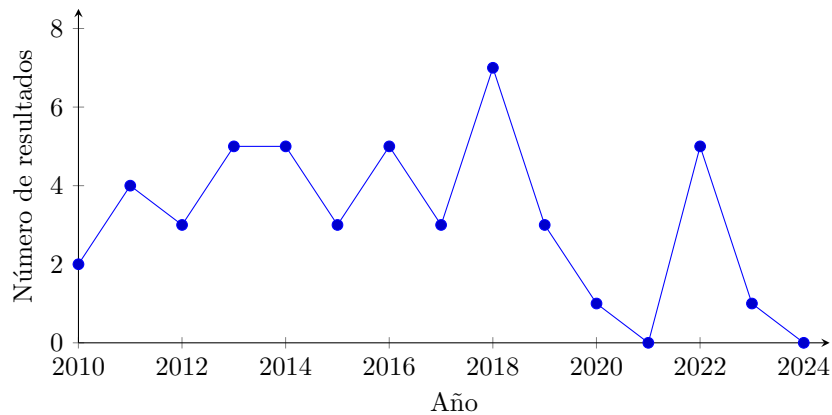


Figura 3.1: Distribución de resultados por año.

documentos cuyo resumen contenga el término *genetic*, lo que permite filtrar resultados no relevantes relacionados con algoritmos genéticos.

Utilizando dicho *string*, la plataforma arroja 48 resultados, los cuales se distribuyen, según año de publicación, como se puede ver en la Figura 3.1. A partir del gráfico se observa una distribución irregular de resultados a lo largo de los años. Se muestra un patrón interesante de altibajos, sin una tendencia clara de crecimiento o disminución sostenida. Se aprecia un pico máximo de siete artículos en 2018, seguido por una caída significativa en los años posteriores. Es particularmente interesante notar la ausencia de publicaciones en 2021 y el primer semestre de 2024, lo que podría indicar cambios en el interés de investigación o en las prioridades académicas durante esos períodos.

Una vez obtenidos los resultados iniciales, se procede a dividir los artículos entre los miembros del equipo para llevar a cabo una evaluación más detallada. Se examina tanto el resumen como parte del contenido de los textos asignados, con el objetivo de determinar su relevancia para abordar el problema en cuestión. Para determinar si un texto es o no relevante, se plantean las siguientes preguntas:

- ¿Aplica el texto técnicas de modelado matemático en la definición y optimización de calendarios académicos?
- ¿Incluye el texto metodologías innovadoras o ideas que puedan adaptarse al contexto específico de la investigación?
- ¿Es el contexto educativo o institucional descrito en el texto similar al del estudio actual?

Siguiendo estos criterios, cada miembro del equipo realiza un análisis de los

textos asignados, decidiendo si son o no relevantes en función de las respuestas a las preguntas planteadas. Esta evaluación individual permite identificar, de manera preliminar, qué artículos podrían contribuir al estudio. Posteriormente, se lleva a cabo una reunión para volver a evaluar, esta vez de forma colectiva, los artículos preseleccionados. Durante esta revisión conjunta, se discuten los textos más prometedores, tomando en cuenta las observaciones de cada miembro del equipo, hasta llegar a un consenso sobre cuáles son los más adecuados.

Tras esta reunión, se seleccionan nueve artículos que se consideran más adecuados para las necesidades y objetivos de la investigación. Para tomar esta decisión, se puntúa cada documento del uno al tres en base a las preguntas planteadas anteriormente, donde uno es “poco relevante” y tres es “muy relevante”. Esta técnica permite fácilmente descartar material que la mayoría considera de menor relevancia. Para los casos de empate, se discute en equipo para entender qué documento es más valioso para el estudio. Luego de este proceso, el material final comprende estos nueve artículos seleccionados y dos tesis de la FIng, que se eligieron inicialmente de Colibrí, los cuales se leen en profundidad para evaluar los antecedentes. En última instancia, se agrega un texto adicional a la búsqueda, ya que muchos de los otros textos revisados mencionaban este texto específico, dentro de sus referencias, o bien especificaban que ciertas partes se basaban en dicho estudio previo, lo que sugiere su relevancia para la investigación.

3.2. Antecedentes

Luego de la lectura de los doce textos seleccionados, se procede a clasificarlos por problema abordado y metodología utilizada. En cuanto al problema abordado, los textos se organizan en dos categorías:

- **Optimización de horarios:** enfocada en la planificación y distribución eficiente de tareas o eventos.
- **Optimización de flujos:** orientada hacia la mejora en la distribución y circulación de recursos o información.

Respecto a la metodología utilizada, los textos se clasifican en base a las diferentes técnicas que utilizan:

- **Programación matemática:** se utilizan modelos matemáticos para resolver los problemas.
- **Programación entera mixta:** se combinan variables enteras y continuas en los modelos para resolver el problema.

- **Enfoque híbrido:** se combinan varios enfoques metodológicos para abordar los problemas de manera más flexible.
- **Heurístico/Metaheurístico:** se emplean heurísticas y metaheurísticas para encontrar soluciones aproximadas.
- **Múltiples objetivos:** se consideran varios objetivos simultáneos, buscando un balance entre ellos para encontrar soluciones óptimas.

Notar que para el caso de “problema abordado” un texto se encuentra en únicamente una categoría, mientras que para el caso de “metodología utilizada” un texto puede aplicar varias de las técnicas. Esta clasificación puede verse en el Anexo A. En base a esta distinción, se presenta un breve resumen de los textos estudiados, separados por el problema que abordan.

3.2.1. Optimización de horarios

La optimización de horarios representa la categoría más extensa, abarcando nueve de los doce trabajos analizados. Estos estudios abordan diversos contextos educativos y culturales, desde la programación de exámenes hasta la organización de eventos artísticos.

En el ámbito de la programación de exámenes, destacan dos trabajos que, aunque comparten el objetivo común de optimizar la calendarización de evaluaciones, emplean metodologías similares, pero con algunas diferencias. [Neri y Peirano \(2024\)](#), abordan la calendarización de parciales en la FIng. El trabajo implementa un enfoque híbrido que combina técnicas de programación matemática con heurísticas específicas del dominio. De esta forma, se busca crear un calendario de parciales que optimice la distribución de las evaluaciones, considerando restricciones como la diferencia de días entre UC, el uso de una matriz de coincidencia de estudiantes para evitar la congestión de evaluaciones para los estudiantes, y respetando las capacidades de los salones disponibles. El modelo base se formula utilizando programación matemática, capturando así las restricciones fundamentales del problema. Luego, se aplican heurísticas especializadas para manejar aspectos como la distribución de exámenes a lo largo del período de evaluación y la consideración de preferencias de los distintos departamentos. Para la evaluación del modelo, se presenta una fórmula que integra la diferencia de días entre evaluaciones, la diferencia de turnos y la coincidencia de estudiantes compartidos entre UC.

Por su parte, [Al-Yakoob y cols. \(2010\)](#), se enfocan en la programación de exámenes y asignación de supervisores en la Universidad de Kuwait, utilizando un enfoque híbrido que combina programación entera mixta con heurísticas específicas para problemas de programación de exámenes. Mientras que el modelo matemático proporciona una formulación precisa del problema, las heurísticas

especializadas son utilizadas para generar soluciones iniciales y mejorar la eficiencia de la resolución. El modelo matemático desarrollado incluye tanto variables enteras como continuas. Las variables enteras se utilizan para representar decisiones como la asignación de exámenes a períodos específicos o la selección de aulas, mientras que las variables continuas modelan aspectos como el tiempo entre exámenes. Las restricciones del modelo capturan requisitos como la capacidad de las aulas, la disponibilidad de supervisores y la distribución equitativa de exámenes para los estudiantes. La función objetivo busca minimizar el número total de períodos de examen y maximizar el tiempo entre exámenes para los estudiantes. Utilizan CPLEX para resolver el modelo de programación entera mixta y aplican un procedimiento heurístico iterativo para mejorar la asignación de supervisores, dada la complejidad computacional.

En cuanto a la programación de horarios escolares, se observa una diversidad de enfoques metodológicos. **Post y cols. (2012)** proponen un enfoque diferente para la programación de horarios escolares, buscando asignar horarios, aulas y profesores a clases, considerando diversas restricciones duras (como evitar conflictos) y blandas (como las preferencias de los profesores por días libres). Este estudio introduce el concepto de transferencias cíclicas, un método heurístico que mejora iterativamente las soluciones de programación de horarios, permitiendo explorar eficientemente el espacio de soluciones y encontrar mejoras significativas en horarios ya existentes. El método de transferencias cíclicas se basa en la construcción de un grafo de mejora donde ciertos ciclos negativos corresponden a vecinos de la solución actual. Estos ciclos se exploran mediante un método recursivo, lo que permite identificar posibles mejoras en la asignación de horarios. Para optimizar el proceso, se emplean técnicas de descenso de ganancia parcial y primer descenso, que buscan reducir el costo de la solución mediante ajustes locales en los horarios. Finalmente, se evalúa la calidad de las soluciones en función del “costo total”, medido como la suma ponderada de violaciones de restricciones blandas. El algoritmo también mide el porcentaje de mejora respecto a la solución inicial y el tiempo de ejecución, lo que permite comparar el método propuesto con otros ya existentes.

Por otro lado, **da Fonseca y cols. (2016)**, presentan un *solver* para la programación de horarios en escuelas secundarias, buscando crear horarios escolares optimizados que satisfagan las necesidades de todas las partes involucradas. Para ello se desarrolla un algoritmo híbrido que combina técnicas de búsqueda local con otros métodos metaheurísticos. El proceso comienza utilizando el *Kingston High School Timetabling Engine* para generar una solución inicial rápidamente. Aunque esta solución puede no ser factible para instancias más complejas, proporciona un punto de partida adecuado para las fases siguientes. Posteriormente, se aplica *Simulated Annealing* para mejorar la solución inicial. Esta técnica simula un proceso físico de enfriamiento gradual, donde se generan soluciones vecinas aleatorias y se aceptan movimientos que mejoran la solución o, en algunos casos, aquellos que empeoran la solución con una cierta probabilidad. Esto permite al algoritmo escapar de óptimos locales y explorar nuevas regiones del

espacio de soluciones. El recalentamiento se utiliza como mecanismo adicional para evitar estancamientos en óptimos locales. Finalmente, el algoritmo emplea *Iterated Local Search*, que perfecciona la solución obtenida mediante la aplicación de perturbaciones y la posterior búsqueda local. Este enfoque iterativo continúa hasta que se cumplen los criterios de parada, como un tiempo límite o cuando se dejan de ver mejoras significativas. El *GOAL solver* equilibra la intensificación y la diversificación de la búsqueda, abordando restricciones clave como la asignación de horarios y recursos, la evasión de conflictos, y el cumplimiento de límites de carga de trabajo. Además, las soluciones son evaluadas según el número de violaciones de restricciones duras (obligatorias) y blandas (deseables), asignando penalizaciones según la gravedad de las infracciones.

A diferencia de los anteriores, [Andrade y cols. \(2019\)](#), abordan la programación de horarios escolares combinando un modelo matemático con procedimientos de búsqueda local y búsqueda local iterativa. De esta forma, el enfoque permite una exploración eficiente del espacio de soluciones y la mejora iterativa de los horarios. Consideran tanto restricciones duras como blandas. Las restricciones duras incluyen que un profesor no puede impartir clases a más de un grupo a la vez y que los horarios semanales deben estar completamente ocupados. Las restricciones blandas consideran las preferencias de los profesores sobre días y clases consecutivas. La evaluación se basa en el valor de la función objetivo, el número de días adicionales de clase, y la satisfacción de las preferencias, además del tiempo de procesamiento computacional. Estas técnicas se aplicaron a 14 casos de estudio en escuelas públicas, comparando los resultados con soluciones óptimas obtenidas por métodos exactos.

Por su parte, [Canseco-González y cols. \(2016\)](#), abordan la optimización de horarios escolares en una institución de nivel medio superior en México, utilizando un modelo de programación matemática resuelto mediante el algoritmo de ramificación y acotamiento (*Branch and Bound*) y el software LINGO. El estudio busca asignar cursos a grupos en periodos específicos, cumpliendo con restricciones de disponibilidad de profesores, aulas y preferencias horarias. La solución generada por el modelo maximiza el uso de recursos y mejora la satisfacción de profesores y estudiantes, respetando las restricciones institucionales y pedagógicas impuestas, logrando una asignación más eficiente que los métodos manuales tradicionales. Los tiempos de resolución del modelo fueron muy cortos (cuatro segundos), cumpliendo con todas las restricciones impuestas, demostrando así la eficacia del enfoque. El modelo considera restricciones como la distribución de horas semanales por curso, el cumplimiento de que ciertos cursos deben de impartirse en días y horarios específicos, y la evasión de conflictos de horario entre grupos. Además, se limita la programación a horarios matutinos, y se respeta un máximo de tres grupos por curso.

Otro tema que se ha tratado es la organización de horarios para presentaciones escolares. Esto es lo que tratan [Akkan y cols. \(2016\)](#), donde buscan agrupar estudiantes en equipos según criterios de diversidad/homogeneidad y asignarlos

a horarios para presentar los resultados de sus proyectos, llegando a una solución que es tanto óptima como robusta. Para esto, se utiliza un enfoque híbrido y de múltiples objetivos. El estudio se enfrenta a los problemas usuales de organizar eventos en un contexto académico, como profesores que no pueden asistir, estudiantes que faltan y presentaciones que duran más (o menos) de lo esperado. Para lograrlo, se separa el problema en dos etapas: primero buscan soluciones óptimas, y luego, de entre esas, seleccionan la más robusta utilizando un método heurístico basado en vecindarios. Para la primera etapa, se utiliza programación entera mixta, capturando las restricciones y objetivos principales del problema. El objetivo es buscar un equilibrio entre la minimización de conflictos de horarios y la maximización de la asistencia de los estudiantes. Posteriormente, en la segunda etapa, se aplican técnicas de optimización robusta para generar soluciones que sean resistentes a la incertidumbre y a cambios inesperados. De esta forma, se crean horarios que no solo son óptimos en condiciones ideales, sino que también mantienen un buen rendimiento bajo diferentes escenarios de perturbación.

Por su parte, [De Werra \(1985\)](#) da una introducción teórica al problema de asignación de horarios para cursos, presentando distintos modelos y explicando como estos se podrían insertar en problemas más complejos. A lo largo del artículo, se presentan varios modelos que utilizan principalmente enfoques teóricos basados en grafos y redes, destacando las dificultades y soluciones que surgen a medida que se introducen restricciones y se los escenarios se hacen más complejos. Se comienza presentando problemas básicos como la asignación diaria de profesores a cursos, teniendo en cuenta restricciones de horarios. Para estos casos, el modelo es representado a través de un grafo bipartito, donde los nodos representan profesores y cursos, y las aristas, las interacciones entre ellos. A través de la coloración de grafos, se busca una solución factible, aunque no necesariamente óptima. En ausencia de restricciones, se emplea la coloración de aristas, pero una vez que se incluyen restricciones, el problema se convierte en NP-difícil, lo que lleva a una transición hacia la coloración de nodos y el desarrollo de heurísticas específicas para cada caso. Luego, se presenta el problema de asignación de cursos a horarios, con el objetivo de evitar que dos cursos tomados por los mismos estudiantes coincidan en el tiempo. En este caso, el modelo también se representa mediante un grafo, donde los nodos son cursos y las aristas indican conflictos de horario entre cursos que deben ser evitados. El problema se aborda inicialmente utilizando coloración de nodos, y se proponen diversas heurísticas para encontrar una solución que minimice las colisiones. A partir de esto, más adelante, se introduce un modelo basado en programación matemática para resolver el problema que considere las restricciones planteadas. Por último, se introduce un modelo más general que aborda la asignación de cursos cuando los estudiantes pueden elegir múltiples cursos, lo que aumenta considerablemente la complejidad. El autor utiliza la coloración de nodos para modelar el problema y propone técnicas avanzadas, como el uso de flujos en redes, para encontrar soluciones eficientes.

Finalmente, [Machado y Baráibar \(2021\)](#), tratan una temática con un contexto totalmente diferente, pero que guarda similitudes en como se resuelve el problema. A lo largo del texto, se busca aplicar técnicas de optimización y programación matemática al Carnaval uruguayo, particularmente el diseño del calendario para el Carnaval de las Promesas, buscando equilibrar la concurrencia del público en las distintas etapas del evento. El modelo presentado incluye variables para representar la asignación de grupos artísticos a fechas y escenarios. Se consideran restricciones que capturan requisitos como la rotación de jurados, la equidad en la exposición de los participantes y algunas otras restricciones logísticas del evento, con el objetivo de maximizar tanto la calidad del espectáculo como la justicia en la competencia. Además, se consideran factores como las preferencias del público, la rotación de escenarios y la distribución de las categorías de espectáculos. A lo largo del trabajo, desarrollan una serie de modelos con diferentes restricciones o particularidades. Resolvieron los modelos utilizando dos *solvers* diferentes: CPLEX y Choco, para posteriormente compararlos.

3.2.2. Optimización de flujos

En el área de optimización de flujos, los tres trabajos analizados se centran en diferentes aspectos del movimiento y transferencia eficiente de personas y recursos.

Por un lado [Vermuyten y cols. \(2016\)](#), proponen un enfoque que no solo optimiza la asignación de recursos académicos, sino que también minimiza los flujos de estudiantes entre clases para evitar congestiones en pasillos y áreas comunes. El estudio se enfoca en reducir los tiempos de desplazamiento entre aulas y evitar colapsos en las áreas comunes durante los cambios de clase, generando horarios eficientes en cuanto a recursos y movilidad. El modelo utiliza un enfoque en dos etapas: primero, asigna clases a franjas horarias maximizando las preferencias y respetando restricciones; luego, reasigna las aulas para minimizar los flujos estudiantiles. Las restricciones incluyen evitar clases en horarios consecutivos conflictivos para los profesores y garantizar que los estudiantes no tengan largos períodos libres entre clases. Los resultados del modelo destacan la reducción de congestiones y optimización del tiempo de viaje de los estudiantes, manteniendo la factibilidad del horario y respetando las restricciones impuestas.

Por otro lado, [X. Feng y cols. \(2014\)](#), se enfocan en la optimización de la eficiencia de transferencia de pasajeros en sistemas de transporte. Se utilizan simulaciones avanzadas para modelar y mejorar los flujos de personas en entornos complejos como aeropuertos o estaciones de tren, lo cual permite evaluar el impacto de diferentes estrategias en escenarios realistas. El estudio considera múltiples factores y restricciones en un enfoque multiobjetivo. Por un lado, analiza los tiempos de espera, la congestión en puntos de transferencia y la

sincronización entre diferentes modos de transporte. Por otro lado, tiene en cuenta la capacidad y frecuencia de los trenes, así como las limitaciones operativas de la infraestructura ferroviaria. El objetivo es optimizar simultáneamente la eficiencia operativa, la satisfacción del pasajero y la utilización de recursos, equilibrando estas variables dentro de las restricciones del sistema. Para ello, se desarrolla un modelo de optimización integrado que busca racionalizar los horarios de llegada y salida de los trenes, minimizando así el tiempo promedio de espera de las transferencias. Para la evaluación del algoritmo, se consideraron un conjunto de métricas como el tiempo promedio de espera de transferencia para un pasajero y el porcentaje de disminución del tiempo promedio de espera tras la implementación de la racionalización de los horarios.

Finalmente [Z. Feng y cols. \(2018\)](#), abordan la optimización del enrutamiento de trenes en estaciones de alta velocidad buscando determinar las rutas óptimas para los trenes a través de los cuellos de botella (entradas/salidas) y las vías de llegada/salida de la estación, considerando trenes heterogéneos y contando con la estructura detallada de la estación. Se aplica un algoritmo de búsqueda tabú para resolver un problema de optimización multiobjetivo que considera aspectos como la eficiencia operativa, la satisfacción del pasajero y la utilización de la infraestructura ferroviaria. El modelo desarrollado busca equilibrar objetivos que resultan conflictivos, como minimizar los tiempos de espera de los trenes, maximizar la puntualidad y optimizar el uso de las plataformas de la estación. La aplicación de la búsqueda tabú permite explorar eficientemente el espacio de soluciones y encontrar soluciones de alta calidad. Aunque no es directamente un problema educativo, las técnicas son aplicables a la programación de horarios en general. Se valida el algoritmo mediante un conjunto de métricas de evaluación, que incluyen la utilización equilibrada de las vías de llegada/salida, el tiempo total de viaje de los trenes en la estación, las desviaciones totales del tiempo de salida de los trenes respecto al horario planificado. También se toma como métrica el tiempo de ejecución del algoritmo, y para casos pequeños se compara la calidad de la solución con el óptimo.

Capítulo 4

Análisis del problema

En este capítulo se presenta una definición del problema, con sus componentes y restricciones. Posteriormente, se realiza un relevamiento de requisitos, que sirve como base para el desarrollo del sistema que dará solución al problema.

4.1. Definición del problema

El problema a resolver es la definición de un calendario de evaluaciones de la FIng, pudiendo ser estas instancias de exámenes o parciales. Este proceso consiste en asignar a cada UC un turno específico en un día dentro del período establecido, el cual corresponde a un rango de días que se tienen disponibles para la programación de evaluaciones. En la práctica, las evaluaciones se suelen programar de lunes a viernes en tres turnos (matutino, vespertino y nocturno), y los sábados en dos turnos (matutino y vespertino). Además, los períodos suelen tener entre 10 y 18 días.

Cada turno en un día dado tiene una capacidad asociada, la cual se determina por la cantidad de salones disponibles para tomar las evaluaciones en dicho turno. Aunque la cantidad de salones es fija, no todos los salones que se utilizan pertenecen a la FIng, sino que algunos pertenecen a la Facultad de Ciencias Económicas y de Administración y otros a la Facultad de Arquitectura, Diseño y Urbanismo, lo que implica que su disponibilidad está sujeta a las necesidades de estas facultades, complicando así el proceso de asignación. Actualmente, el proceso de generación del calendario se realiza en dos etapas: primero se genera un calendario considerando la capacidad total disponible, y luego Bedelías es responsable de asignar los salones específicos. En base a esto, la solución se enfoca en resolver la primera parte del proceso, es decir, la generación del calendario

teniendo en cuenta la capacidad total, mientras que la asignación de los salones continúa siendo responsabilidad de Bedelías.

Un factor adicional que se considera es la interacción con otras facultades, en particular con la Facultad de Química, dado que algunos estudiantes de la FIng comparten UC con esta facultad. La Facultad de Química fija sus calendarios de evaluaciones a principios de año, y una vez establecidos, estos son inamovibles. Por lo tanto, es necesario que la solución sea capaz de generar un calendario que se adapte a estas fechas preestablecidas.

Además, es crucial tener en cuenta las UC que suelen ser cursadas en simultáneo por los estudiantes. Un claro ejemplo de esto está en las UC que se encuentran en el mismo semestre dentro de las currículas sugeridas de las diferentes carreras, las cuales definen qué UC el estudiante idealmente debiera cursar en un mismo semestre, lo cual implica que la mayoría de los estudiantes de esa carrera estarán cursando las UC sugeridas allí en simultáneo. Además de las currículas, se toma como insumo, provisto por la Unidad de Enseñanza de Facultad de Ingeniería (UEFI), un promedio de la cantidad de estudiantes que cursan dos UC en simultáneo en los últimos años (se nombra a este valor “coincidencia” de dos UC).

También es importante considerar el sistema de previas de las UC, ya que éste influye en la combinación de UC que un estudiante puede tomar en un semestre dado. Para explicarlo mejor, se da un ejemplo: dadas la UC x y la UC y , si x es previa de y , se precisa haber aprobado x para cursar y , por lo que un estudiante no puede cursar la UC x e y en simultáneo. El sistema toma este dato como un incentivo para que las evaluaciones de UC que son previas sean en el mismo día, o en días cercanos.

El objetivo del sistema es definir un calendario factible, pero que a su vez minimice las coincidencias de evaluaciones que puedan afectar a aquellos estudiantes que deban rendir evaluaciones de diferentes UC, ya sea siguiendo la currícula sugerida, avanzando según las previas que hayan cumplido, o en casos donde históricamente se han identificado altas coincidencias entre evaluaciones. Hoy en día, los estudiantes se ven afectados a veces por coincidencias, no pudiendo rendir la evaluación de ambas UC porque coinciden en fecha, o a causa de la cercanía de las fechas no pueden dedicar todo el tiempo necesario para cada UC, causando un rendimiento menor al que lograrían. De esta forma, buscamos crear un calendario que facilite la participación de los estudiantes en todas sus evaluaciones programadas.

Semestre	Sábado 27 Día 1	Lunes 29 Día 2	Martes 30 Día 3	Jueves 2 Día 4	Viernes 3 Día 5	Sábado 4 Día 6	Lunes 6 Día 7	Martes 7 Día 8	Miércoles 8 Día 9	Sábado 11 Día 10
Química y Alimentos	Tercero	Estadística		Física 3		Q. orgánica I/II		Cálculo Vectorial		
	Cuarto									AGPI
	Quinto	Probabilidad y Estadística / Química Inorgánica	Fenómenos de Transporte	Bioquímica		Química Analítica	Termodinámica aplicada		Fisicoquímica 103	
	Séptimo	Tecn. y Serv. Industriales 1	Ing. reacciones químicas 1		Electrotécnica 1	Análisis de alimentos		Transferencia de calor y masa 2		
	Noveno		Ingeniería Bioquímica	Dinámica y Control de Procesos		Tratamiento biológico de Efluentes	Instrumentación Industrial	Diseño de procesos químicos	Gestión de Calidad	
Eléctrico	Primero	GAL I/ GAL 2	CDIV/ CDIVV		Física 1	/ Física 2			Programación 1	
	Tercero		Probabilidad y Estadística	Física 3		Mecánica Newtoniana	Principios de Química General	Cálculo Vectorial		
	Quinto		Electrotécnica	Señales y Sistemas						Funciones de Variable Compleja
	Séptimo	Antenas y Propagación	Electrotécnica	Intro. Invest. De Operaciones	Redes de datos 2 / Electrónica de potencia		Medidas eléctricas		Redes eléctricas / Electrónica Avanzada 2	AGPI
	Noveno	Imágenes médicas	Electrotécnica	Subestaciones de Media Tensión				Tecnologías de redes y servicios de telecomunicaciones		
Temas de Comunicación	Primero	GAL I/ GAL 2	Matemática inicial	CDIV/ CDIVV	Física 1	/ Física 2			Programación 1	
	Tercero		Probabilidad y Estadística	Física 3			Matemática Discreta 1	Cálculo Vectorial		
	Quinto		Antenas y Propagación	Señales y Sistemas		Lógica			Programación 2	Funciones de Variable Compleja
	Séptimo		Teoría de Lenguajes		Intro. Invest. De Operaciones	Redes de datos 2		Sistemas Operativos		AGPI

Figura 4.1: Parte de la hoja de cálculo usada para el armado del calendario.

4.2. Relevamiento de requisitos

Hoy en día, Ema Soboredo, Asistente Académica de Decanato, confecciona los calendarios manualmente, mediante planillas de cálculo. Se puede ver una parte de este documento en la Figura 4.1. Este método, si bien funciona y es lo que se hace desde hace ya tiempo, es bastante tedioso. Ema debe manejar manualmente una gran cantidad de datos, incluyendo información sobre UC, capacidad de salones, restricciones de horarios, currículas sugeridas y sistema de previas. Además, tiene que considerar restricciones específicas de ciertas UC, como aquellas que requieren un turno completo o las que no pueden programarse en ciertos días. Sumado a esto, debe realizarlo dentro de un plazo ajustado para que el calendario esté listo a tiempo para el período de evaluaciones. Por lo tanto, al ser un proceso mayormente manual y depender de la experiencia y criterio de Ema, es propenso a errores y le consume una cantidad significativa de tiempo.

Se pone especial énfasis en comprender cómo Ema lleva a cabo el trabajo, el espacio físico utilizado, las herramientas empleadas y los procesos seguidos. A partir de esto, se modela el problema y se busca generar posibles soluciones de calendarios de evaluaciones factibles para que Ema evalúe, simplificando así su trabajo.

Al analizar los requisitos se separan en dos categorías: del usuario y del sistema. Esta división ayuda a tener una visión más completa de lo que se necesita construir. Esta forma de organizar los requisitos permite asegurar que no solo se enfoque en lo que el sistema debe hacer, sino también en cómo debe hacerlo para que realmente sea útil y eficaz en el contexto de la facultad.

4.2.1. Requisitos funcionales

Los requisitos funcionales representan las capacidades específicas que el sistema debe proporcionar para resolver el problema de la generación del calendario de evaluaciones. Estos requisitos definen las acciones concretas que los usuarios podrán realizar y las funcionalidades que el sistema debe implementar para resolver el problema.

Requisitos de usuario

1. Indicar los días disponibles para tomar evaluaciones, así como los turnos disponibles en cada uno de estos días.
2. Asignar algunas UC a días y turnos específicos. Esta asignación no puede ser modificada. Esto es muy importante para poder resolver casos como el de la Facultad de Química, que fija de antemano sus evaluaciones y estas no pueden ser movidas.
3. Indicar un rango de días en el que debe asignarse una UC.
4. Seleccionar las UC que deben ser asignadas en el período.
5. Indicar la cantidad de alumnos promedio inscriptos a una determinada UC.
6. Elegir un valor de holgura para la capacidad total de los salones. Por ejemplo, si se tiene un turno con capacidad para 2000 estudiantes, se requiere poder definir que se quiere usar un cierto porcentaje de esa capacidad.

Requisitos de sistema

1. Imposibilitar que en un mismo día se asignen las evaluaciones de dos UC que están sugeridos para un mismo semestre.
2. Considerar la cercanía de las UC en la currícula sugerida para realizar la asignación. Se debe fomentar que exista la mayor separación de días posible entre UC que están sugeridos para un mismo semestre.
3. Considerar la cercanía de las UC que poseen una coincidencia elevada. Se debe fomentar que si muchos estudiantes coinciden en dos UC, entonces la evaluación de estas UC esté lo más separada posible.

Capítulo 5

Definición del modelo matemático

En este capítulo se propone un modelo matemático para resolver el problema planteado. Se busca generar un calendario que satisfaga las restricciones institucionales, mientras optimiza la distribución temporal de las evaluaciones para beneficiar a los estudiantes.

La formulación matemática se basa en programación lineal entera mixta, donde las variables principales representan la asignación de evaluaciones a días y turnos específicos. El modelo incorpora diversos aspectos críticos del problema, incluyendo:

- La capacidad de los salones disponibles en cada turno para cada día.
- Las preferencias y restricciones preestablecidas para algunas UC.
- Las relaciones entre UC basadas en las currículas sugeridas.
- Los patrones históricos de inscripción simultánea de estudiantes.
- La dependencia de previas entre UC.

A continuación, se detallan los componentes del modelo, comenzando por la definición de los conjuntos que establecen el marco estructural, seguidos por los parámetros que capturan los datos específicos del problema, y las variables de decisión que representan las asignaciones a realizar. Posteriormente, se presentan las restricciones que determinan las soluciones factibles. Finalmente, se describe la función objetivo que guía la optimización hacia soluciones deseables.

5.1. Conjuntos

- D : días del calendario. Conjunto ordenado de días que componen el período.
- C : unidades curriculares. Por ejemplo “Cálculo Diferencial e Integral en varias variables”, “Geometría y álgebra lineal 1”, “Redes de computadoras”, etc.
- T : turnos posibles $T = \{\text{matutino, vespertino, nocturno}\}$.
- $T_d \subseteq T$: turnos disponibles para el día $d \in D$. Por lo general los días de semanas cuentan con los tres turnos, mientras que los sábados se cuenta con dos turnos.
- K : carreras. Por ejemplo “Ingeniería en computación”, “Ingeniería eléctrica”, etc. Para los casos de carreras con más de un perfil, para simplificar el modelado, cada perfil se considera como una carrera diferente.
- S : ordinal de semestres de las carreras. Representan los diez semestres que se encuentran en la currícula sugerida de cada carrera de la FIng. Se utiliza para identificar en qué semestre se sugiere cursar cada UC. $S = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$.
- $SUG \subseteq C \times S \times K$: tripletas (c, s, k) donde la UC c se sugiere en el semestre s para la carrera k . Hay únicamente una tripleta por par (c, k) ya que se sugiere un único semestre para una UC en una carrera.
- $PA_c \subseteq D \times T$: parejas de día d y turno t_d , (d, t_d) , en que debe tener lugar la evaluación de la UC c (pre-asignados).
- $P \subseteq C \times C$: pares (c_1, c_2) donde la UC c_1 es previa de la UC c_2 .
- DS : a partir del conjunto D de días y T de turnos, este conjunto calculado tiene todas las posibles distancias que pueden existir entre pares de días/turnos.
- $COP \subseteq C \times C$: indica pares de UC, que por una cuestión de asignación de profesores, no es viable que se asignen en el mismo turno. Por ejemplo, no sería viable que la evaluación de “Probabilidad y Estadística” sea en el mismo turno que la de “Geometría y Álgebra Lineal 1”, ya que ambas UC son masivas y los profesores que se necesitan para supervisar la evaluación van a ser muy probablemente los mismos.
- $CC = \{(c_1, c_2) \in C \times C, c_1 < c_2\}$: conjunto auxiliar con los pares de UC para evitar tener pares simétricos $(c_1, c_2), (c_2, c_1)$. De esta manera, definiendo las variables y restricciones sobre este conjunto, se reduce el tamaño del problema, mejorando su eficiencia.

5.2. Parámetros

- $cp_{d,t}$: capacidad total de estudiantes para la asignación de evaluaciones en el día $d \in D$ y turno $t \in T_d$. Depende de los salones disponibles en ese día y turno específico
- fac_cp : proporción de la capacidad total que se decide usar como máximo, expresada como un número entre 0 y 1. Es decir, la capacidad efectiva.
- $alta_co$: valor de coincidencia de estudiantes entre dos UC que se considera alto, lo que indicaría que dos UC con un valor de coincidencia igual o mayor a este deberían estar separadas en el calendario de evaluaciones.
- max_ev : representa la máxima cantidad de evaluaciones que pueden ser asignadas por turno.
- ins_c : promedio de inscriptos a la UC $c \in C$.
- co_{c_1, c_2} : promedio histórico de estudiantes inscriptos simultáneamente en las UC c_1 y c_2 , $c_1, c_2 \in C$. Este promedio se calculó a partir de los valores de coincidencia registrados en diferentes semestres entre 2020 y 2023.
- $M = \max DS$, representa la mayor distancia posible entre dos UC.

A su vez, se definen otros parámetros, los cuales son deducidos a partir de los datos.

- $dist_{d_1, d_2}$: distancia en días entre los días d_1 y d_2 del conjunto D de días del calendario,

$$dist_{d_1, d_2} = |d_1 - d_2|, \quad d_1, d_2 \in D. \quad (5.1)$$

- $dist_sem_{c_1, c_2}$: calcula la mínima distancia expresada en semestres de la currícula sugerida entre dos UC c_1 y $c_2 \in C$, si ambas están sugeridas para una misma carrera $k \in K$. En caso contrario se le asigna $|S|$, dado que este valor es la máxima distancia posible que podemos tener. Matemáticamente se formula como,

$$dist_sem_{c_1, c_2} = \begin{cases} \min_{k \in K} \left| \sum_{(c_1, s, k) \in SUG} s - \sum_{(c_2, s, k) \in SUG} s \right| & \text{si } \exists k \in K : (c_1, s_1, k), (c_2, s_2, k) \in SUG, c_1, c_2 \in C \\ |S|, \text{ en otro caso.} & \end{cases} \quad (5.2)$$

Por ejemplo, si se considera las UC “Cálculo Diferencial e Integral en una variable” (UC1) y “Programación 2” (UC2), la distancia semestre calcula

para todas las currículas, cuál es la distancia entre los semestres en los que estas UC se sugieren. Luego, de estas distancias, se toma la mínima. Para el caso en que el par de UC no se encuentre en ninguna trayectoria sugerida de alguna carrera, el parámetro toma el valor $|S|$. Si se supone que la UC1 se sugiere para el semestre 2 en la carrera k_1 , y para el semestre 4 en la carrera k_2 , y la UC2 se sugiere para el semestre 4 en la carrera k_1 , y para el semestre 5 en la carrera k_2 , entonces la distancia semestre es $\min\{|2 - 4|, |4 - 5|\} = 1$.

- $dist_peso_{ds}$: valor que asocia el peso asociado a que dos UC se asignen con una separación de ds . Por construcción, se busca que sea un número entre 0 y 1, y que sea alto para valores de ds chicos y que luego decrezca rápidamente a medida que ds sube,

$$dist_peso_{ds} = e^{-0.2 \cdot ds}, \quad ds \in DS. \quad (5.3)$$

5.3. Variables de decisión

- $x_{c,d,t}$: vale 1 si la UC $c \in C$ se asigna al día $d \in D$ en el turno $t \in T_d$, y 0 en otro caso.
- $w_{c_1,c_2,ds}$: vale 1 si la distancia entre las UC $(c_1, c_2) \in CC$ es ds y 0 en otro caso.
- $z_{c_1,c_2} \in \mathbb{R}$: representa el valor absoluto de la diferencia en días/turnos de asignación entre las UC $(c_1, c_2) \in CC$.

Dado que el problema se busca resolver con un modelo de programación lineal, no es posible representar directamente un valor absoluto, ya que la función valor absoluto no es una función lineal. Una técnica que se utiliza para resolver estos tipos de problemas, y que se explica en mayor detalle en el artículo de [Shanno y Weil \(1971\)](#), es definir dos variables que controlen la parte positiva y negativa del valor absoluto. Para que esto funcione, es importante que solo una de las variables tome un valor distinto de 0, y por la naturaleza del problema planteado, es necesario definir otra variable binaria que controle dicha restricción.

- $zp_{c_1,c_2} \geq 0$: representa la parte positiva de la diferencia, tiene valor cuando el día en el que $c_1 \in C$ es asignado es mayor al día en que $c_2 \in C$ es asignado.
- $zm_{c_1,c_2} \geq 0$: representa la parte negativa de la diferencia, tiene valor cuando el día en el que c_2 es asignado es mayor al día en que c_1 es asignado.

- y_{c_1, c_2} : controla cual parte (positiva o negativa) es la que tiene valor. Vale 1 si la parte positiva es la que tiene valor, y 0 en caso contrario.

Por ejemplo, si la UC c_1 se asigna al día 4, y la UC c_2 se asigna al día 1, entonces $zp_{c_1, c_2} = 3$, $zm_{c_1, c_2} = 0$ y $y_{c_1, c_2} = 1$. En cambio, si la UC c_1 se asigna al día 1, y la UC c_2 se asigna al día 4, entonces $zp_{c_1, c_2} = 0$, $zm_{c_1, c_2} = 3$ y $y_{c_1, c_2} = 0$.

5.4. Restricciones

- La evaluación de una UC se asigna a un único día y turno,

$$\sum_{d \in D} \sum_{t \in T_d} x_{c, d, t} = 1, \quad c \in C. \quad (5.4)$$

- Si dos UC están sugeridas en el mismo semestre para la misma carrera, se asignan a días distintos,

$$\sum_{t \in T_d} (x_{c_1, d, t} + x_{c_2, d, t}) \leq 1, \quad (5.5)$$

$$d \in D, \quad (c_1, s, k), (c_2, s, k) \in SUG : c_1 \neq c_2.$$

- No se puede superar la capacidad disponible de los salones, teniendo en cuenta el factor de capacidad, para cada día y turno,

$$\sum_{c \in C} ins_c x_{c, d, t} \leq fac_cp \ cp_{d, t}, \quad d \in D, t \in T_d. \quad (5.6)$$

- En un turno dado, solo pueden asignarse hasta un máximo de max_ev evaluaciones,

$$\sum_{c \in C} x_{c, d, t} \leq max_ev, \quad d \in D, t \in T_d. \quad (5.7)$$

- Las evaluaciones que se pre-asignan a algún conjunto de posibles días y turnos, se asigna en alguno de ellos,

$$\sum_{(d, t) \in PA_c} x_{c, d, t} = 1, \quad c \in C. \quad (5.8)$$

- Las UC que tengan profesores coincidentes no pueden ser asignadas en el mismo turno,

$$x_{c_1, d, t} + x_{c_2, d, t} \leq 1, \quad d \in D, t \in T_d, (c_1, c_2) \in COP. \quad (5.9)$$

- Dos UC solo pueden tener una distancia asignada,

$$\sum_{ds \in DS} w_{c_1, c_2, ds} = 1, \quad (c_1, c_2) \in CC. \quad (5.10)$$

- La distancia asignada a dos UC coincide con el valor de z para esas UC,

$$\sum_{ds \in DS} ds \, w_{c_1, c_2, ds} = z_{c_1, c_2}, \quad (c_1, c_2) \in CC. \quad (5.11)$$

- Si dos UC tienen una alta coincidencia de estudiantes, sus evaluaciones deben estar separadas al menos dos días. Se considera una alta coincidencia cuando el promedio histórico de coincidencias es mayor o igual a *alta_co*,

$$z_{c_1, c_2} \geq 2, \quad (c_1, c_2) \in CC, co_{c_1, c_2} \geq alta_co. \quad (5.12)$$

- Asegurar que z tome el valor absoluto de la distancia entre las evaluaciones de dos UC.

- La diferencia entre los días se puede expresar como la diferencia entre una parte positiva y una negativa,

$$zp_{c_1, c_2} - zm_{c_1, c_2} = \sum_{d \in D} \sum_{t \in T_d} d \, x_{c_1, d, t} - \sum_{d \in D} \sum_{t \in T_d} d \, x_{c_2, d, t} \quad (5.13)$$

$$(c_1, c_2) \in CC.$$

- El valor absoluto, es decir el valor de z , es la suma de ambas partes,

$$z_{c_1, c_2} = zp_{c_1, c_2} + zm_{c_1, c_2} \quad (c_1, c_2) \in CC. \quad (5.14)$$

- Control de zp a través de y ,

$$zp_{c_1, c_2} \leq M \, y_{c_1, c_2} \quad (c_1, c_2) \in CC. \quad (5.15)$$

- Control de zm a través de y ,

$$zm_{c_1, c_2} \leq M \, (1 - y_{c_1, c_2}) \quad (c_1, c_2) \in CC. \quad (5.16)$$

Con estas restricciones se logra que solo una de las partes tenga valor. Cuando el día en que se asigna c_1 es mayor al día en que se asigna c_2 , en base a las restricciones (5.13) y (5.14), zp_{c_1, c_2} tomará como valor un número mayor o igual a la diferencia de días donde se toman las evaluaciones de c_1 y c_2 , la cual es positiva. En base a la restricción (5.15), esto solo puede ocurrir cuando $y_{c_1, c_2} = 1$, ya que de lo contrario ($y_{c_1, c_2} = 0$) quedaría que $zp_{c_1, c_2} \leq 0$, con lo cual, al ser una variable positiva, se tendría $zp_{c_1, c_2} = 0$ que no puede suceder. Luego, como $y_{c_1, c_2} = 1$, en base a la restricción (5.16) tenemos que $zm_{c_1, c_2} \leq 0$, con lo cual, al ser una variable

positiva nos queda que $zm_{c_1,c_2} = 0$, por lo que finalmente debe pasar que zp_{c_1,c_2} tomará como valor exactamente la diferencia de días.

De forma totalmente análoga, sucede que si el día en que se asigna la evaluación de la UC c_1 es menor que el día en que se asigna la de la UC c_2 , entonces la diferencia entre c_1 y c_2 es negativa, por lo que zm_{c_1,c_2} deberá tomar un valor positivo. Para que esto suceda tiene que pasar que $y_{c_1,c_2} = 0$, que a su vez hace que $zp_{c_1,c_2} = 0$, por lo que finalmente zm_{c_1,c_2} debe ser exactamente igual a la diferencia de días de las evaluaciones.

Finalmente, si la evaluación de la UC c_1 y la de la UC c_2 son en el mismo día, luego zp_{c_1,c_2} y zm_{c_1,c_2} deben ambos tomar el mismo valor, y este debe de ser 0, dado que si alguno no lo fuera, ambos deberían tomar el mismo valor positivo, lo cual es imposible que suceda ya que debido a las restricciones (5.15) y (5.16), si $y_{c_1,c_2} = 0$, luego $zp_{c_1,c_2} = 0$ y si $y_{c_1,c_2} = 1$, luego $zm_{c_1,c_2} = 0$.

5.5. Función objetivo

En esta sección se describe la función objetivo utilizada para optimizar la distribución de evaluaciones en el calendario de evaluaciones. El objetivo principal es minimizar conflictos y mejorar la organización, considerando tanto la coincidencia de estudiantes como las dependencias entre unidades curriculares (UC). Para ello, se plantean dos componentes principales: la distribución temporal de evaluaciones y la incorporación de relaciones de previa entre UC.

5.5.1. Distribución temporal de evaluaciones

En primer lugar, se busca distribuir las evaluaciones de modo que pocos estudiantes rindan en el mismo día o en días consecutivos. Para lograr esto, se aumenta la distancia entre la evaluación de las UC que son sugeridas en el mismo semestre para una carrera y/o que tienen muchos estudiantes que las cursan en simultáneo. Para esto, se utiliza un valor normalizado de la coincidencia ($\overline{co_{c_1,c_2}}$) y de la distancia en semestres ($\overline{dist_sem_{c_1,c_2}}$).

Para normalizar la coincidencia se divide el valor para las UC dadas por el máximo valor de coincidencia para dos UC cualquiera y diferentes, obteniendo así un valor entre 0 y 1; cuanto menor sea la coincidencia, más cercano a 0 será este valor. Entonces,

$$\overline{co_{c_1,c_2}} = \frac{co_{c_1,c_2}}{1 + \max_{(c_1,c_2) \in CC} co_{c_1,c_2}}.$$

Se suma 1 al denominador para evitar una división por 0 en el caso excepcional en que no haya coincidencias para ningún par de UC, causando que el máximo sea 0.

Por otro lado, el inverso de la distancia en semestres normalizada se define mediante la siguiente ecuación,

$$\overline{dist_sem}_{c_1, c_2} = \frac{1}{\overline{dist_sem}_{c_1, c_2} + 1}.$$

Con esta normalización se obtiene también un valor entre 0 y 1 tal que cuanto más alejados en semestres estén las UC, más cercano a 0 será el valor. Se le suma 1 al denominador para evitar la división entre 0.

Con estos valores calculamos el concepto de “riesgo”, el cual se define como la suma de la coincidencia y la distancia en semestres normalizadas, ponderando la distancia de semestres normalizadas con un parámetro alfa: $\overline{co}_{c_1, c_2} + \alpha \cdot \overline{dist_sem}_{c_1, c_2}$.

Por último, se quiere que el riesgo penalice de forma distinta, en base a la distancia entre las evaluaciones de dos UC, buscando que la penalización sea mayor cuanto más cercanas sean las evaluaciones. Para esto, se multiplica la expresión previa por el valor $dist_peso_{ds}$, que indica el peso asignado a una distancia ds . De esta forma, la expresión resultante es,

$$dist_peso_{ds} \left(\overline{co}_{c_1, c_2} + \alpha \cdot \overline{dist_sem}_{c_1, c_2} \right).$$

Por último, queremos que esta solo se aplique si las UC se encuentran a esa distancia dada. Para esto se utiliza la variable $w_{c_1, c_2, ds}$ la cual solo vale 1 cuando la distancia entre las evaluaciones de c_1 y c_2 es efectivamente ds . Además, queremos sumar estas penalizaciones para todos los pares de UC, por lo que la expresión resulta en,

$$\sum_{(c_1, c_2) \in CC} \sum_{ds \in DS} dist_peso_{ds} \left(\overline{co}_{c_1, c_2} + \alpha \cdot \overline{dist_sem}_{c_1, c_2} \right) w_{c_1, c_2, ds}. \quad (5.17)$$

5.5.2. Incorporación de previas

Como segundo término en la función objetivo se busca beneficiar que las UC que tienen dependencia de previa sean evaluadas en días cercanos.

Para lograr esto, en la expresión planteada anteriormente (5.17), se suma un término adicional que penalice si UC que son previas son asignadas lejos en el calendario. Dadas dos UC, cuyas evaluaciones se encuentran a una distancia ds , se considera el valor $\min\{1, \frac{ds}{5}\}$. De esta forma, si dos UC que son previas son

asignadas en el mismo día ($ds = 0$), la penalización es 0, y esta va aumentando a medida que más lejos están sus evaluaciones, con un máximo valor de 1, para mantener una escala consistente con las demás partes de la función objetivo. De esta manera, al ser el modelo uno de minimización, se favorece naturalmente que las UC con relación de previa estén cercanas en el calendario. Como queremos que solo se aplique para pares que estén a distancia ds , multiplicamos por la variable de decisión $w_{c_1, c_2, ds}$, y de forma análoga al caso anterior, sumamos las penalizaciones para todos los pares de UC que son previas, llegando así la siguiente expresión,

$$\sum_{(c_1, c_2) \in P} \sum_{ds \in DS} \min \left\{ 1, \frac{ds}{5} \right\} w_{c_1, c_2, ds}. \quad (5.18)$$

Agrupando las expresiones (5.17) y (5.18) se obtiene la siguiente expresión,

$$\begin{aligned} & \min \sum_{(c_1, c_2) \in CC} \sum_{ds \in DS} dist_peso_{ds} (\overline{co_{c_1, c_2}} + \alpha \cdot \overline{dist_sem_{c_1, c_2}}) w_{c_1, c_2, ds} \\ & + \sum_{(c_1, c_2) \in P} \sum_{ds \in DS} \min \left\{ 1, \frac{ds}{5} \right\} w_{c_1, c_2, ds}. \end{aligned} \quad (5.19)$$

Finalmente, para controlar la influencia relativa de las dependencias de previas en el modelo, se introduce un parámetro de ponderación $\beta \in [0, 1]$. Este parámetro multiplica al componente de la función objetivo relacionado con las dependencias de previas entre UC (5.18), permitiendo ajustar su importancia frente al resto de los términos de la función objetivo. De esta forma, la función objetivo resultante es,

$$\begin{aligned} & \min \sum_{(c_1, c_2) \in CC} \sum_{ds \in DS} dist_peso_{ds} (\overline{co_{c_1, c_2}} + \alpha \cdot \overline{dist_sem_{c_1, c_2}}) w_{c_1, c_2, ds} \\ & + \beta \sum_{(c_1, c_2) \in P} \sum_{ds \in DS} \min \left\{ 1, \frac{ds}{5} \right\} w_{c_1, c_2, ds}. \end{aligned} \quad (5.20)$$

Capítulo 6

Implementación de la solución

En este capítulo, se detalla el proceso de implementación de la solución propuesta para la generación automática del calendario de evaluaciones. La implementación combina técnicas de programación matemática con herramientas especializadas, asegurando que la solución sea funcional, eficiente y escalable.

6.1. Tecnologías utilizadas

Para la implementación del modelo de optimización, se emplean diversas librerías de código abierto que facilitan su formulación, resolución y análisis. Estas herramientas permiten gestionar datos de entrada, definir restricciones y variables de manera eficiente, resolver el problema con distintos solvers y analizar los resultados obtenidos. El desarrollo se realiza en Python debido a su amplia adopción en la comunidad científica, su versatilidad y la disponibilidad de librerías especializadas para optimización. A continuación, se describen las principales tecnologías utilizadas y su rol dentro del flujo de trabajo. El repositorio con todo el código asociado al proyecto de grado se encuentra disponible en [Carle, de Sierra, y Spira \(2025\)](#).

6.1.1. Modelador de programación matemática PuLP

Para la resolución del modelo matemático, se opta por el uso de la librería PuLP en Python, una librería de código abierto desarrollada por COIN-OR, una

“[...] fundación científica y educativa sin fines de lucro, cuya misión es crear y difundir conocimientos relacionados con todos los aspectos de la investigación de operaciones computacionales” (*COIN-OR: Computational Infrastructure for Operations Research*, 2016). De acuerdo a su documentación, “PuLP es un modelador de programación entera lineal y mixta escrito en Python” (Mitchell y cols., 2011).

Una de sus mayores ventajas es que utilizando PuLP es posible definir modelos de optimización de manera independiente al solver a utilizar, usando una única sintaxis definida por la propia librería y extendida por las instrucciones del lenguaje Python. De esta forma, no es necesario estudiar la sintaxis o detalles específicos de cada solver, sino que se escribe el modelo una vez y luego puede resolverse con diferentes solvers cambiando únicamente una línea de código. PuLP soporta el uso de varios solvers, tanto de uso libre como comerciales. Entre los de código libre se destaca el de la propia fundación, COIN-OR Forrest y cols. (2024). Dentro de los comerciales, teniendo la licencia apropiada, se puede usar CPLEX (*IBM ILOG CPLEX Optimization Studio*, 2022), Gurobi (*Gurobi Optimization, LLC*, 2024), entre otros.

Otra ventaja de PuLP es que se trata de una librería de código abierto, por lo que cuenta con un amplio soporte por parte de la comunidad académica, ofrece una extensa documentación, y tiene por detrás una comunidad activa que contribuye a su desarrollo. Adicionalmente, el hecho de que PuLP sea una librería de Python permite desarrollar una solución integral donde todas las etapas del proceso, desde la ingesta y preprocesamiento de datos, pasando por el modelado del problema y la optimización, hasta la generación del calendario y el análisis de resultados, pueden implementarse en un mismo entorno de desarrollo. Esta integración no solo facilita el mantenimiento y la extensibilidad del código, sino que también permite aprovechar el amplio ecosistema de librerías científicas y de análisis de datos disponibles en Python, como pandas para el manejo de datos o matplotlib para visualizaciones. Esto resulta particularmente ventajoso en el contexto de un proyecto de Ingeniería en Computación, ya que se alinea con las mejores prácticas de desarrollo de software, donde se busca minimizar las fricciones entre diferentes componentes del sistema y mantener un flujo de trabajo coherente y eficiente.

6.1.2. Otras librerías

Además de PuLP, se emplean varias librerías adicionales para diferentes tareas dentro del flujo de trabajo:

- *itertools* (*itertools — Functions creating iterators for efficient looping*, 2025) es una biblioteca estándar de Python que proporciona herramientas eficientes para trabajar con iteradores. En este proyecto, se utiliza pa-

ra generar de manera eficiente todas las combinaciones y permutaciones posibles necesarias para implementar las restricciones y la definición de variables del modelo matemático, optimizando así el proceso de generación de restricciones, y la cantidad de variables que se definen. Por ejemplo, en múltiples restricciones y cálculos, el modelo itera sobre el conjunto $\{(c_1, c_2) \in C \times C, c_1 < c_2\}$. Para evitar el constante re-cálculo del conjunto, se puede pre-calcular, y definir como un nuevo conjunto *PARES_UC*. Para definirlo se puede utilizar la función `combinations` de `itertools`, para generar todas las posibles combinaciones de dos UC. Esto se realiza de la siguiente forma.

```
from itertools import combinations
PARES_UC = combinations(C, 2)
```

- pandas ([pandas development team, 2024](#)) es una biblioteca de análisis de datos que ofrece estructuras de datos flexibles y herramientas para manipular datos tabulares. En este proyecto, se emplea para leer, procesar y transformar los datos de entrada del problema, como los horarios disponibles, la capacidad de los salones y otros parámetros necesarios, facilitando su posterior uso en el modelo de optimización.

6.2. Arquitectura de la solución

El sistema desarrollado para la generación de calendarios de evaluaciones se estructura en varios módulos interconectados, cada uno responsable de una etapa específica dentro del proceso. La Figura 6.1 presenta, mediante un diagrama de actividad UML, la estructura general del sistema, ilustrando el proceso desde la obtención de los datos iniciales hasta la generación del calendario final.

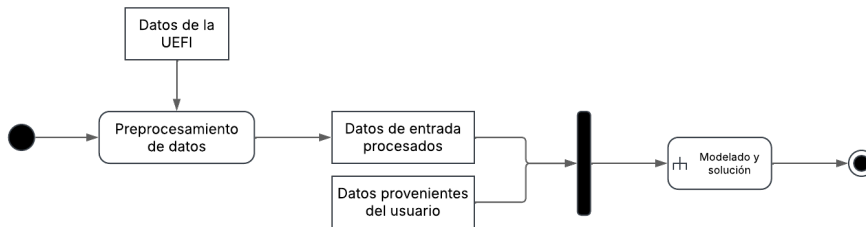


Figura 6.1: Diagrama de actividad general del sistema

El proceso comienza con el procesamiento de datos provenientes de la UEFI, para llevarlos a un formato más estandarizado dentro del modelo. Estos datos, junto con otros datos de entrada provenientes del usuario, son utilizados en la

etapa central del sistema: el modelado y la solución del problema de optimización. Esta etapa crucial se detalla en la Figura 6.2.

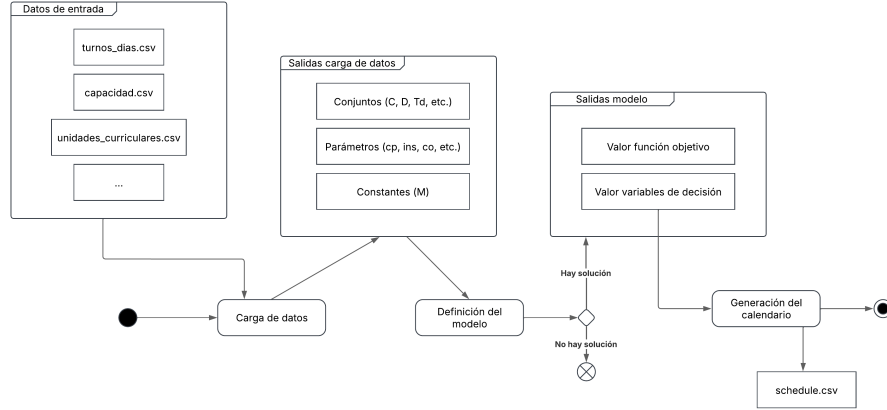


Figura 6.2: Diagrama de actividad de la sub-actividad “Modelado y solución”

Como se observa en el diagrama, el proceso de modelado y solución comprende tres fases principales: la carga de datos desde archivos CSV, la definición del modelo matemático con sus respectivos parámetros y conjuntos, y finalmente, la generación del calendario basada en la solución óptima encontrada. A continuación, se describen en detalle los módulos que conforman esta arquitectura.

6.2.1. Procesamiento de datos

El sistema cuenta con dos módulos de procesamiento: `process_uri_data.py` y `convert_previas.py`, cada uno responsable de transformar datos de diferentes fuentes a un formato estandarizado para su posterior uso en el modelo.

Procesamiento de datos de la UEFI

El módulo `process_uri_data.py` se encarga de procesar dos conjuntos de datos provenientes de la UEFI: la matriz de coincidencia entre UC y el historial de inscripciones.

Para la matriz de coincidencia, se lee un archivo provisto por la UEFI, el cual contiene una lista con pares de UC por período, que indica la cantidad de estudiantes que en ese período se encontraban cursando ambas UC en simultáneo. Un ejemplo con algunas filas de este archivo puede verse en la Tabla 6.1.

mat1	cod1	mat2	cod2	nom_periodo	cantidad
CALCULO DIF. E INTEGRAL EN UNA VARIABLE	1061	CALCULO DIF. E INTEGRAL EN VARIAS VARIABLES	1062	202101	1
CALCULO DIF. E INTEGRAL EN UNA VARIABLE	1061	GEOMETRIA Y ALGEBRA LINEAL 1	1030	202101	1534
CALCULO DIF. E INTEGRAL EN UNA VARIABLE	1061	PROGRAMACION 1	1373	202302	427
CALCULO DIF. E INTEGRAL EN UNA VARIABLE	1061	PROGRAMACION 1	1373	202401	414

Tabla 6.1: Ejemplo del CSV indicando la coincidencia de estudiantes entre UC por período

Como primer paso, se extrae el semestre desde el período académico. Los datos originales incluyen el semestre como el último dígito del campo `nom_periodo`. Para lograr esto se utiliza el método `str[-1]` que accede al último carácter de cada string, y luego se convierte a un número entero utilizando `astype(int)`

```
# Extraer el semestre directamente del último dígito del período
df['semestre'] = df['nom_periodo'].str[-1].astype(int)
```

Posteriormente, se calcula el promedio de estudiantes que cursan simultáneamente cada par de UC, agrupando por los códigos de las unidades curriculares y el semestre:

```
# Calcular cantidad promedio para cada par cod1-cod2 por semestre
avg_coincidencia = df_coincidencia.groupby(
    ["cod1", "cod2", "semestre"]
)[ "cantidad" ].mean().reset_index()
```

La función `groupby()` agrupa los datos por las columnas especificadas, `mean()` calcula el promedio de la columna cantidad para cada grupo, y finalmente `reset_index()` convierte el resultado en un data frame regular.

Por último, se separa la información por semestres, y se renombran las columnas para dejar la información en el formato necesario.

```
# Crear DataFrames separados para cada semestre de
# coincidencia con formato coincidencia.csv
avg_conc_sem1 = (
    avg_coincidencia[avg_coincidencia["semestre"] == 1]
    .rename(columns={
        "cod1": "uc_1",
        "cod2": "uc_2",
        "cantidad": "coincidencia"
    })
    .drop("semestre", axis=1)
)
```

La función `rename()` cambia los nombres de las columnas al formato requerido y `drop()` elimina la columna de semestre que ya no es necesaria.

El procesamiento de inscripciones se realiza de forma similar, con la diferencia de que la información provista por la UEFI cuenta con un campo `fecha_inicio`. Por tanto, primero se debe procesar este campo para deducir el semestre que corresponde.

```
# Convertir fecha_inicio a tipo datetime
df_ins['fecha_inicio'] = pd.to_datetime(df_ins['fecha_inicio'])

# Agregar columna de semestre (1 o 2)
df_ins['semestre'] = df_ins['fecha_inicio'].dt.month.map(
    lambda x: 1 if x <= 6 else 2
)
```

La función `pd.to_datetime()` convierte cadenas de texto con fechas, a objetos del tipo `datetime`, permitiendo acceder a componentes como el mes mediante `dt.month`. Luego, usando `map()` con una función lambda, se asigna el semestre 1 a los meses de enero a junio, y el semestre 2 al resto.

El cálculo del promedio de inscripciones sigue un proceso similar al de coincidencia:

```
# Calcular promedio por código y semestre
avg_ins = df_ins.groupby(
    ['codenservicio_mat', 'semestre']
)['cantidad'].mean().reset_index()
```

Finalmente, los datos se separan por semestre y se formatean según los requerimientos del sistema. Esto implica renombrar columnas y redondear valores numéricos:

```
# Crear DataFrames separados para cada semestre de
# inscripciones con formato inscriptos.csv
avg_ins_sem1 = (
    avg_ins[avg_ins["semestre"] == 1]
    .rename(columns={
        "codenservicio_mat": "uc",
        "cantidad": "inscriptos"
    })
    .drop("semestre", axis=1)
)

# Redondear valores de inscriptos a enteros
avg_ins_sem1['inscriptos'] =
    avg_ins_sem1['inscriptos'].round().astype(int)
```

Procesamiento de previas

El módulo `convert_previas.py` se encarga de transformar la información de previaturas almacenada en formato YAML, proveniente del repositorio Mi-Carrera (Cedarcode, 2019), a un formato CSV estructurado.

El archivo YAML de entrada utiliza una estructura que permite expresar requisitos mediante operaciones lógicas. Esto hace posible representar situaciones donde una UC requiere haber aprobado varias materias simultáneamente (operador AND) o al menos una de un conjunto de materias (operador OR). Por ejemplo, una UC podría requerir haber aprobado tanto Cálculo I como Álgebra Lineal, o podría requerir haber aprobado al menos una de varias materias optativas.

Para procesar esta estructura jerárquica, el módulo implementa un enfoque recursivo mediante la función `extract_subject_codes`. Esta función navega a través del árbol de operaciones lógicas, identificando y extrayendo las relaciones directas entre UC. Cuando encuentra una hoja del árbol (un operando de tipo “subject”), registra la relación entre la UC principal y la UC requerida. En el caso de encontrar un nodo interno (un operador lógico), la función se llama recursivamente sobre sus operandos, acumulando todas las relaciones encontradas.

```
def extract_subject_codes(operands, code):
    results = []
    if not isinstance(operands, list):
        operands = [operands]

    for operand in operands:
        if isinstance(operand, dict):
            if operand.get('type') == 'subject':
                results.append(
                    (code, operand.get('subject_needed_code'))
                )
            elif operand.get('type') == 'logical':
                if 'operands' in operand:
                    results.extend(
                        extract_subject_codes(operand['operands'], code)
                    )
    return results
```

Esta función se invoca para cada elemento del archivo YML leído, y se acumulan las relaciones de previa en una variable `relationships`. Finalmente, este resultado acumulado es utilizado para escribir la información de previas en un archivo CSV estandarizado para el modelo, *previas.csv*.

```
relationships = []
for item in data:
```

```

if (
    isinstance(item, dict)
    and "subject_code" in item
    and "operands" in item
):
    relationships.extend(
        extract_subject_codes(item["operands"], item["subject_code"])
    )

# Escribir a CSV
with open("previas.csv", "w", newline="", encoding="utf-8") as f:
    writer = csv.writer(f)
    writer.writerow(["uc", "uc_requerida"]) # Títulos
    for rel in relationships:
        if rel[0] and rel[1]: # Solo escribir si ambos códigos existen
            writer.writerow(rel)

```

6.2.2. Carga de datos

El módulo `csv_data_to_model_data.py` se encarga de la lectura y procesamiento de los datos de entrada desde archivos CSV. Este módulo implementa la función `load_calendar_data` que toma como parámetro la ruta al directorio que contiene los archivos CSV y retorna un diccionario con todos los conjuntos y parámetros necesarios para el modelo. La función realiza la carga de múltiples archivos CSV utilizando `pandas`.

```

def load_csv(name):
    file_path = os.path.join(dir_name, f"{name}.csv")
    if not os.path.exists(file_path):
        raise FileNotFoundError(f"No se encontró {file_path}")
    return pd.read_csv(file_path)

```

Para cada tipo de dato, se realiza una transformación específica. Por ejemplo, para cargar los días del calendario y las UC se hace lo siguiente:

```

# Días del calendario
D = list(load_csv("dias")["id"])

# Unidades curriculares
C = list(load_csv("unidades_curriculares")["codigo"])

```

El módulo también se encarga de generar conjuntos derivados. Por ejemplo, `PARES_UC` que se define como las combinaciones de dos cursos, utilizando `itertools`. O también el conjunto `UC_MISMO_SEMESTRE` el cual contiene el conjunto de las UC que son sugeridos en el mismo semestre en una misma carrera.

```

# Pares de UC
PARES_UC = list(combinations(C, 2))

# Cursos mismo semestre
UC_MISMO_SEMESTRE = {
    (c1, c2): True
    for (c1, s1, k1) in SUG
    for (c2, s2, k2) in SUG
    if c1 != c2 and s1 == s2 and k1 == k2
}

```

6.2.3. Definición del modelo

El módulo `solve.py` implementa el modelo matemático utilizando PuLP. Esto se divide en una serie de pasos: configuración del problema, definición de las variables de decisión, definición de la función objetivo, definición de las restricciones y, finalmente, resolución del modelo. A continuación, se describen estos pasos en mayor detalle.

Configuración del problema

Para definir un problema se instancia un nuevo elemento de la clase `LpProblem`, indicando si es un problema de minimización o de maximización.

```

problem = pl.LpProblem("Optimizacion_Calendario", pl.LpMinimize)

```

Variables de decisión

Posteriormente se definen todas las variables del problema, utilizando la clase `LpVariable`. Por ejemplo, en el modelo, se define la variable $x_{c,d,t} \in \{0,1\}$, $c \in C, d \in D, t \in T_d$. Esto se puede representar en el código de la siguiente manera:

```

x = {}
for c in C:
    for d in D:
        for t in Td[d]:
            x[(c, d, t)] = pl.LpVariable(
                f"x_{c}_{d}_{t}",
                cat=pl.LpBinary
            )

```

Se inicializa un diccionario vacío $\mathbf{x} = \{\}$ que almacenará todas las variables de decisión $\mathbf{x}$. Luego, se utilizan tres bucles anidados para iterar sobre todas las UC ($\mathbf{c} \in \mathbf{C}$), todos los días disponibles ($\mathbf{d} \in \mathbf{D}$) y todos los turnos disponibles para cada día ($\mathbf{t} \in \mathbf{Td}[\mathbf{d}]$). Para cada combinación se crea una variable binaria usando la clase `LpVariable`. A cada variable se le asigna además un nombre único `f"x_{c}_{d}_{t}"` y una categoría `cat=pl.LpBinary`, la cual indica que la variable solo puede tomar como valores 0 o 1.

Función objetivo

Para definir la función objetivo, se agrega la expresión de la misma al modelo definido anteriormente en la instancia `problem`. Se representan las sumatorias utilizando la función `lpSum`, e indicando sobre qué conjuntos en particular se realiza la sumatoria. Por ejemplo, se puede representar la primera parte de la función objetivo definida en (5.20) de la siguiente forma:

```
problem += pl.lpSum(
    pl.lpSum(
        dist_peso[ds]
        * (co[c1, c2] / max_co + 1 / (dist_sem[c1, c2] + 1))
        * w[c1, c2, ds]
        for ds in DS
    )
    for c1, c2 in PARES_UC
)
```

En este ejemplo en concreto, la expresión utiliza dos sumatorias anidadas: una sobre todos los pares de cursos (`PARES_UC`) y otra sobre todas las distancias posibles (`DS`).

Restricciones

Para definir las restricciones del problema, de forma similar a como se define la función objetivo, se deben agregar las expresiones de las mismas al modelo definido en la instancia `problem`.

Por ejemplo, en el modelo se tiene como restricción que “la evaluación de una UC se asigna únicamente a un día y turno” (5.4). En código se puede expresar esta restricción de la siguiente manera:

```
for c in C:
    problem += (
        pl.lpSum(x[c, d, t] for d in D for t in Td[d]) == 1,
```

```
        f"AsignacionUnica_{c}",
    )
```

Se itera sobre todas las UC (c in C) y para cada una se agrega una restricción al modelo definido en PuLP utilizando el operador `+=`. En este caso, la restricción se construye utilizando la función `lpSum` que suma todas las variables de decisión $x[c,d,t]$ para todos los días disponibles (d in D) y todos los turnos disponibles en cada día (t in $Td[d]$). Esta suma se iguala a 1 (`== 1`), lo cual asegura que cada UC sea asignada exactamente a un único día y turno. Finalmente, a cada restricción se le asigna un nombre único para facilitar la identificación de las restricciones.

Las demás restricciones del modelo se definen de manera análoga. Los operadores de “para todo” se implementan como bucles, mientras que las sumatorias se representan mediante la función `lpSum` de PuLP.

Solución del modelo

Una vez definido el modelo, se puede resolver utilizando PuLP. Para hacer esto, se tiene que primero instanciar un solver, para lo cual PuLP provee APIs para cada uno de los solvers que soporta. Posteriormente se utiliza el método `solve` de la instancia del problema, pasando como parámetro el solver que se quiere utilizar. Por ejemplo, si se quiere resolver el problema con CPLEX, se puede indicar de la siguiente manera:

```
solver = pl.CPLEX_CMD()
problem.solve(solver)
```

En el momento de instanciar el solver, es posible indicar por parámetro diferentes opciones de configuración, como por ejemplo el tiempo límite, el gap relativo, la cantidad máxima de nodos a explorar, entre otros. Una vez que se resuelve el problema, se puede acceder a las variables resultantes, así como al valor final de la función objetivo.

```
objective = pl.value(problem.objective)
variables = problem.variables()
```

6.2.4. Generación del calendario

Una vez obtenida la solución del modelo matemático, es necesario transformar las variables de decisión resultantes en un formato más útil y comprensible para los usuarios finales. Esta tarea la realiza el módulo `generate_schedule.py` mediante la función `create_schedule_csv`.

El proceso comienza con el filtrado de las variables de decisión. El modelo matemático genera variables binarias $\mathbf{x}_{\{c,d,t\}}$ que indican si una UC se asigna a un día y turno específico. De estas variables, se seleccionan aquellas que tienen valor 1, ya que son las que representan asignaciones efectivas de evaluaciones.

Luego, se estructuran los datos en un formato más natural para representar un calendario. Se crea una estructura jerárquica que organiza las asignaciones por día y turno, procesando los nombres de las variables que contienen esta información en el formato "x_curso.dia_turno". Por ejemplo, si se tienen evaluaciones asignadas a tres días con dos turnos cada uno, la estructura tendría esta forma:

```

schedule = {
  1: { # Día 1
    1: ["Cálculo 1", "Física 1"],      # Turno 1
    2: ["Programación 1"]             # Turno 2
  },
  2: { # Día 2
    1: ["Álgebra Lineal"],             # Turno 1
    2: ["Matemática Discreta"]         # Turno 2
  },
  3: { # Día 3
    1: ["Probabilidad y Estadística"], # Turno 1
    2: ["Lógica"]                     # Turno 2
  }
}

```

Finalmente, se genera un archivo CSV que contiene el calendario de evaluaciones, donde las filas representan los días, las columnas representan los turnos, y cada celda contiene las UC asignadas a ese momento específico. Este formato fue elegido por su simplicidad y porque puede ser fácilmente abierto con cualquier programa de hojas de cálculo, permitiendo su uso por parte de cualquier usuario interesado.

Capítulo 7

Métricas de calidad de solución

Para determinar la calidad de las soluciones, es necesario establecer métricas con las que se pueda determinar si una solución cumple con las características deseadas. A su vez, las métricas permiten comparar distintos calendarios y así ayudar a decidir cuál es mejor.

7.1. Métrica de estudiantes afectados

Un estudiante afectado se define como quien tenga dos evaluaciones el mismo día o en días consecutivos. El objetivo es evaluar en base a la coincidencia de estudiantes de dos UC dadas, y la fecha en que se posiciona cada evaluación, la cantidad de estudiantes promedio que resultan afectados con la asignación. Luego, se calcula el total de estudiantes afectados de la siguiente manera,

$$m_{estudiantes} = \sum_{(c_1, c_2) \in CC} \left(co_{c_1, c_2} \sum_{dist \in \{0, 1\}} w_{c_1, c_2, dist} \right).$$

Por cada par de UC se suma la coincidencia de estas, solo si la distancia que las separa es de cero o un día. Para lograr esto, se realiza la multiplicación de la coincidencia por el valor de w para esas dos UC solo en los casos que $dist$ es 0 o 1, causando que solo se agregue el valor de la coincidencia para las UC que están a la distancia esperada. Un valor más alto indica que se impacta negativamente a una mayor cantidad de estudiantes.

7.2. Métrica de coincidencia

Con el objetivo de visualizar si el modelo logra un espaciamiento adecuado entre UC, se define una métrica que lo represente.

En primer lugar, se calcula el promedio de separación en días, ponderando por la cantidad de estudiantes que cursan dos UC en simultáneo. Esto se logra dando mayor peso a la separación de UC con mayor coincidencia. El cálculo se realiza de la siguiente forma,

$$\bar{z}_{co} = \frac{\sum_{(c_1, c_2) \in CC} co_{c_1, c_2} z_{c_1, c_2}}{\sum_{(c_1, c_2) \in CC} co_{c_1, c_2}}.$$

En el numerador, se suma por cada par de UC la distancia en días entre las evaluaciones multiplicada por la cantidad de estudiantes inscriptos simultáneamente. En el denominador, se calcula el total de coincidencias entre todas las parejas de UC, para normalizar el resultado.

Por otro lado, se calcula la desviación estándar para cuantificar la dispersión de las distancias entre evaluaciones,

$$\sigma_{co} = \sqrt{\frac{\sum_{(c_1, c_2) \in CC} co_{c_1, c_2} (z_{c_1, c_2} - \bar{z}_{co})^2}{\sum_{(c_1, c_2) \in CC} co_{c_1, c_2}}}.$$

Si la desviación estándar es alta, significa que algunas evaluaciones están muy separadas, mientras que otras están muy próximas, lo que indica una distribución inconsistente. En cambio, un valor bajo refleja una mayor uniformidad en la separación de evaluaciones.

Finalmente, para obtener el valor de la métrica considerando la media de espaciamiento y la consistencia en distribución, se define la siguiente expresión,

$$m_{coincidencia} = \bar{z}_{co} \left(1 - \alpha \frac{\sigma_{co}}{\bar{z}_{co}} \right)$$

donde:

- $\frac{\sigma_{co}}{\bar{z}_{co}}$ es el coeficiente de variación, que mide la dispersión relativa de las distancias.
- α es un parámetro de ajuste que controla cuánto influye la dispersión en la métrica.

Como se quiere que con una dispersión alta, la penalización sea significativa (cuando $\sigma_{co} = \bar{z}_{co}$, el valor debería disminuir un 50 %), se procede a fijar: $\alpha = 0.5$. Esto implica que:

- Si la dispersión relativa es del 100 % ($\sigma_{co} = \bar{z}_{co}$), la métrica disminuirá en un 50 %

$$m_{coincidencia} = 0.5\bar{z}_{co}.$$

- Si $\sigma = 0$ (todas las distancias son iguales), la métrica permanecerá en

$$m_{coincidencia} = \bar{z}_{co}.$$

Esta métrica penaliza soluciones con alta dispersión en las distancias entre materias con alta coincidencia, garantizando que la separación entre evaluaciones sea más uniforme. Si la variabilidad es baja ($\sigma \ll \bar{z}$), la métrica se mantiene cercana a $\bar{z}_{co}$, mientras que si hay una gran dispersión, el término de penalización se incrementa.

Un valor alto de $m_{coincidencia}$ indica que las evaluaciones de UC con mucha coincidencia entre sí están programadas en días lejanos y con una separación uniforme, mientras que un valor bajo refleja una mayor dispersión y menos coherencia en la asignación de exámenes ponderados por coincidencia.

Finalmente, con $\alpha = 0.5$ la métrica queda definida como

$$m_{coincidencia} = \bar{z}_{co} - 0.5\sigma_{co}.$$

7.3. Métrica de previas

De forma similar a la métrica anterior, se busca generar una métrica que evalúe el esparcimiento de las UC, pero esta vez teniendo en cuenta la relación de previas entre ellas.

En primer lugar, se calcula el promedio de separación en días de las evaluaciones de las UC, teniendo en cuenta únicamente las UC con dependencia de previas,

$$\bar{z}_{previas} = \frac{\sum_{(c_1, c_2) \in P} z_{c_1, c_2}}{|P|}.$$

Luego, se calcula la desviación estándar, la cual permite cuantificar la dispersión de las distancias entre evaluaciones,

$$\sigma_{previas} = \sqrt{\frac{1}{|P|} \sum_{(c_1, c_2) \in P} (z_{c_1, c_2} - \bar{z}_{previas})^2}.$$

Si la desviación estándar es alta, significa que algunas evaluaciones están muy separadas, mientras que otras están muy próximas, lo que indica una distribución inconsistente. En cambio, un valor bajo refleja una mayor uniformidad en la separación de evaluaciones.

Finalmente, para obtener una única métrica que combine la proximidad de las previas y la consistencia en su distribución, de forma análoga a la métrica anterior, con la única diferencia de que en este caso debemos sumar la desviación en lugar de restarla, ya que en esta métrica se penalizan valores altos, se define la siguiente métrica unificada,

$$m_{previas} = \bar{z}_{previas} \left(1 + \alpha \frac{\sigma_{previas}}{\bar{z}_{previas}} \right).$$

Siguiendo un razonamiento similar, se decide utilizar $\alpha = 0.5$. Este valor de α garantiza que soluciones con alta dispersión reciban una penalización considerable, incentivando una distribución más homogénea de las evaluaciones. De esta forma, la métrica resultante queda definida como,

$$m_{previas} = \bar{z}_{previas} + 0.5\sigma_{previas}.$$

7.4. Métrica de trayectoria sugerida

Nuevamente, se busca evaluar el esparcimiento de UC, pero en esta ocasión teniendo en cuenta únicamente las UC que son sugeridas en un semestre para una carrera,

$$\bar{z}_{curricula} = \frac{\sum_{\substack{(c_1, s, k), (c_2, s, k) \in SUG \\ c_1 \neq c_2}} z_{c_1, c_2}}{|\{(c_1, c_2) \mid (c_1, s, k), (c_2, s, k) \in SUG, c_1 \neq c_2\}|}.$$

En el numerador, se calcula la suma de las distancias z_{c_1, c_2} entre las evaluaciones de todas las parejas (c_1, c_2) sugeridas en el mismo semestre s para la misma carrera k . En el denominador, se considera el número total de pares de UC sugeridas en el mismo semestre s y carrera k , obteniendo así el promedio de distancias.

Luego, se calcula la desviación estándar,

$$\sigma_{curricula} = \sqrt{\frac{1}{|P_{curricula}|} \sum_{\substack{(c_1, s, k), (c_2, s, k) \in SUG \\ c_1 \neq c_2}} (z_{c_1, c_2} - \bar{z}_{curricula})^2}$$

donde $P_{curricula}$ representa el conjunto de todos los pares de UC sugeridas en el mismo semestre y carrera.

Esta desviación estándar mide la dispersión de las distancias entre evaluaciones, permitiendo detectar si hay un desbalance en las asignaciones. Un valor alto indica que algunas UC están muy próximas, mientras que otras están demasiado separadas, lo que puede reflejar una distribución inconsistente.

Finalmente, para obtener el valor de la métrica que combine la proximidad y la uniformidad de las evaluaciones de UC sugeridas en el mismo semestre, se define el siguiente índice,

$$m_{curricula} = \bar{z}_{curricula} \left(1 - \alpha \frac{\sigma_{curricula}}{\bar{z}_{curricula}} \right).$$

Se utiliza el mismo valor de α que en la métrica de coherencia de previas, con el mismo objetivo.

Este índice penaliza soluciones con alta dispersión en la distribución de evaluaciones de UC sugeridas en el mismo semestre, incentivando una separación más uniforme. Si la variabilidad es baja ($\sigma_{curricula} \ll \bar{z}_{curricula}$), la métrica se mantiene cercana a $\bar{z}_{curricula}$, mientras que si hay grandes diferencias en la separación, la penalización aumenta.

Un valor alto de $m_{curricula}$ indica que las evaluaciones de UC sugeridas en el mismo semestre están programadas en días lejanos y de manera más uniforme y consistente, mientras que un valor bajo refleja una mayor dispersión en su distribución.

Finalmente, con el valor de $\alpha = 0.5$, la métrica queda definida como

$$m_{curricula} = \bar{z}_{curricula} - 0.5\sigma_{curricula}.$$

Capítulo 8

Resultados experimentales y análisis

Para evaluar el modelo planteado se realizan diferentes pruebas con el objetivo de analizar los resultados que es capaz de obtener. Los calendarios obtenidos se evalúan con las métricas presentadas anteriormente y se comparan con los calendarios oficiales realizados por Ema.

8.1. Casos de prueba y hardware utilizado

Para realizar pruebas se hace uso de los cuatro períodos de parciales del 2024, a los cuales se les asignan diferentes nombres:

- Primer semestre 2024, primeros parciales (1s1p).
- Primer semestre 2024, segundos parciales (1s2p).
- Segundo semestre 2024, primeros parciales (2s1p).
- Segundo semestre 2024, segundos parciales (2s2p).

El armado de un caso de prueba implica la creación de un conjunto de archivos CSV que luego son usados, de acuerdo al flujo definido en la Sección 6.2, como entrada para resolver el modelo matemático. El conjunto se conforma por los siguientes archivos:

- *dias.csv*, *turnos.csv* y *turnos.dias.csv*: estos contienen respectivamente la lista los identificadores de los días disponibles para asignar, los identificadores de turnos y los turnos disponibles para cada día.
- *datos.csv*: almacena parámetros específicos que son utilizados posteriormente en el modelo matemático. De momento contiene el valor del parámetro *fac_cp* (él cuál indica el porcentaje de la capacidad total de salones que se quiere usar como máximo), *alta_co* (él cuál indica el valor de coincidencia de estudiantes entre dos UC que se considera alto, indicando que dos UC con coincidencia mayor o igual a este valor deberían estar separadas) y *max_ev* (él cuál indica el número máximo de evaluaciones que se pueden tener por turno).
- *unidades_curriculares.csv*: contiene los códigos y nombres completos de las UC que se quieren considerar en el período.
- *carreras.csv*: contiene las nueve carreras de ingeniería consideradas en el modelo (Computación, Sistemas de Comunicación, Civil, Eléctrica, Industrial Mecánica, Alimentos, Naval, Producción y Química).
- *capacidad.csv*: indica la capacidad total, sumando la capacidad de todos los salones disponibles, en cada día y turno.
- *inscriptos.csv*: detalla la cantidad de estudiantes inscriptos en cada UC, siendo crucial para determinar la demanda de espacios.
- *semestres.csv*: enumera los identificadores de los 10 semestres académicos para una carrera de Ingeniería, correspondiendo a los cinco años de carrera.
- *trayectoria_sugerida.csv*: establece la ubicación sugerida de cada UC dentro del plan de estudios de cada carrera, indicando el semestre recomendado para cursarla.
- *previas.csv*: incluye una lista con todos los pares de UC que son previas, como se detalla en la Sección 4.1.
- *coincidencia.csv*: contiene para cada par de UC, el promedio histórico de coincidencia de estudiantes que cursaron a la vez esas dos UC en un mismo semestre.
- *preasignaciones.csv*: registra asignaciones previamente establecidas de UC a días y turnos específicos. Notar que puede indicarse un conjunto de posibilidades, para que luego el modelo elija entre una de esas posibilidades. Esto, por ejemplo, resulta útil si se quiere que una UC se asigne “sobre el final del período de evaluaciones”.
- *profesores.csv* contiene pares de UC que comparten profesores, al ser del mismo instituto y que por tanto sus evaluaciones no pueden darse en el mismo turno por falta de cuidadores.

Varios de estos datos pueden ser reutilizados para cualquier caso de prueba, ya que son independientes del semestre o período de la evaluación, pero otros datos, como por ejemplo las UC a considerar, capacidad de salones, días a utilizar, número de inscriptos y preasignaciones, deben armarse de forma específica para cada período de evaluación.

Para la ejecución de los experimentos se utiliza un equipo con chip Apple M1, 16GB de RAM y sistema operativo MacOS, específicamente una MacBook Air (2020) con procesador de 8 núcleos (4 de rendimiento y 4 de eficiencia), GPU integrada de 7 núcleos y almacenamiento SSD de alta velocidad.

8.2. Selección del solver

Con el objetivo de seleccionar un solver para llevar a cabo los experimentos, se seleccionaron tres reconocidos: CPLEX, CBC y Gurobi. Cada solver fue configurado con parámetros equivalentes y se estableció un límite de tiempo de una hora para la obtención de soluciones factibles.

Los resultados de esta evaluación comparativa fueron concluyentes: únicamente Gurobi logró encontrar una solución dentro del tiempo establecido, mientras que CPLEX clasificó el problema como “sin solución” y CBC no consiguió generar ninguna solución factible en el plazo fijado. Esta disparidad en el rendimiento puede atribuirse a las características específicas de nuestro modelo, particularmente a su elevada dimensionalidad y a la compleja estructura de sus restricciones.

En consecuencia, seleccionamos Gurobi como el solver para la ejecución de todos los experimentos subsiguientes.

8.3. Calibración de los parámetros α y β

En el modelo matemático, la función objetivo definida en la expresión (5.20) presenta dos parámetros α y β , los cuales indican respectivamente el peso que se le asigna a la separación de UC que están sugeridas para semestres cercanos en una carrera, y a la distancia entre evaluaciones de UC que son previas entre sí.

De acuerdo a lo observado, los tiempos de ejecución que requiere el solver para encontrar soluciones factibles de calidad son del orden de horas, lo cual hace inviable realizar múltiples ejecuciones extendidas para distintos valores paramétricos. Luego, debido a esta limitación, es necesario determinar un valor específico de α y β previo a la realización de experimentos extendidos. Para

lograr esto, se realizan ejecuciones breves para diferentes combinaciones de α y β , utilizando como referencia el caso de prueba 1s1p (primeros parciales del primer semestre de 2024).

Para la prueba se ejecuta el modelo para todas las combinaciones posibles de los valores de $\alpha \in \{0.5, 1\}$ y $\beta \in \{0, 0.25, 0.5, 0.75, 1\}$. La selección de estos valores se fundamenta en el rol que desempeña cada uno de los términos en la función objetivo. En particular, se considera que el término asociado a α (la separación entre UC cercanas en la currícula sugerida) constituye un criterio fundamental de calidad, cuyo impacto en la solución no se desea atenuar significativamente. Por esta razón, se opta por valores relativamente altos de α , que no reduzcan su importancia a menos de la mitad respecto al resto de los componentes de la función objetivo. En cambio, el término ponderado por β , que promueve la agrupación de evaluaciones entre UC con relación de previa, se considera un criterio más flexible y de naturaleza secundaria. Si bien puede contribuir a una mejor organización del calendario, no resulta estrictamente necesario para garantizar soluciones de buena calidad, y su efecto puede encontrarse parcialmente reflejado en la coincidencia de estudiantes entre UC, lo cual ya está considerado en el primer término de la función objetivo. Por esta razón, se optó por explorar un rango más amplio de valores de β , permitiendo capturar distintos niveles de importancia relativa para este aspecto.

Luego, se ejecuta el modelo para cada combinación paramétrica, con un tiempo límite de 15 minutos, y para cada solución se evalúa con las métricas previamente definidas. Los resultados obtenidos de evaluar cada calendario obtenido con las métricas se pueden observar en la Tabla 8.1.

α	β	m.curricula	m.coincidencia	m.estudiantes	m.previas
0.50	0.00	4.82	4.41	906.98	6.34
0.50	0.25	4.52	3.81	772.50	6.08
0.50	0.50	3.86	4.38	686.23	5.45
0.50	0.75	4.37	4.61	791.43	3.87
0.50	1.00	4.42	4.40	885.67	5.74
1.00	0.00	4.96	4.24	773.83	6.98
1.00	0.25	4.11	4.06	903.02	6.22
1.00	0.50	4.50	4.02	855.68	5.83
1.00	0.75	4.32	3.72	757.25	4.02
1.00	1.00	4.09	4.47	823.28	5.30

Tabla 8.1: Resultados para diferentes valores de los parámetros α y β

A simple vista, no resulta evidente cuál es la mejor combinación de α y β , ya que depende de la métrica observada cuál combinación se posiciona como la más favorable. Esto es consistente con la naturaleza del problema, dado que la función objetivo está definida como una suma ponderada de distintos términos, por lo que el problema se comporta de manera similar a un problema de optimización multiobjetivo. En este contexto, optimizar una métrica puede no implicar necesariamente la optimización de las demás.

Sin embargo, dado el objetivo principal del problema, se considera que la métrica prioritaria es la de estudiantes afectados ($m_{\text{estudiantes}}$). Las demás métricas, relativas a la separación de UC por currícula ($m_{\text{currícula}}$), por coincidencia ($m_{\text{coincidencia}}$) y por relaciones de previa (m_{previas}), si bien son relevantes, tienen como fin asegurar una distribución razonable de las UC en el calendario, por lo que se considera suficiente que sus valores sean adecuados, sin requerir que sean óptimos.

Al analizar los resultados de la Tabla 8.1, se observa que el menor valor de estudiantes afectados se alcanza para la combinación $(\alpha, \beta) = (0.5, 0.5)$, con un valor de 686.23. Además, el resto de las métricas asociadas a esta combinación presentan valores satisfactorios: el promedio de separación por currícula (3.86) y por coincidencia (4.38) superan los tres días, lo que se considera aceptable para la correcta dispersión de evaluaciones. A su vez, el valor de la métrica m_{previas} (5.45) indica una adecuada separación entre UC relacionadas por relaciones de previa, que si bien no es óptima, se mantiene en niveles comparables a otras configuraciones paramétricas.

Luego, considerando tanto la prioridad de minimizar la cantidad de estudiantes afectados como la necesidad de mantener una buena calidad general en la distribución del calendario, se selecciona $(\alpha, \beta) = (0.5, 0.5)$ como la combinación más adecuada para utilizar en la generación del calendario final.

8.4. Experimentaciones de tiempo extendido y análisis de resultados

Definida la combinación de parámetros $(\alpha, \beta) = (0.5, 0.5)$ como la más adecuada según los resultados preliminares, se lleva a cabo una fase de experimentación de tiempo extendido. El objetivo de esta etapa es evaluar la calidad de las soluciones obtenidas bajo condiciones de mayor exploración del espacio de búsqueda, así como realizar un análisis comparativo respecto a los calendarios oficiales vigentes. Para ello, se fija un tiempo de cómputo de ocho horas por instancia y se consideran los cuatro casos de prueba definidos. Cada ejecución se hace empleando la misma configuración utilizada en las pruebas anteriores. Las soluciones generadas son evaluadas mediante las métricas ya presentadas,

1s1p	m_curricula	m_coincidencia	m_estudiantes	m_previas
Publicado	3.82	3.59	560.00	5.19
Generado	3.85	4.38	686.23	5.45

Tabla 8.2: Comparación de métricas entre el calendario generado y el calendario oficial publicado para 1s1p.

1s2p	m_curricula	m_coincidencia	m_estudiantes	m_previas
Publicado	3.85	3.71	1044.22	4.01
Generado	3.93	3.69	1112.67	5.02

Tabla 8.3: Comparación de métricas entre el calendario generado y el calendario oficial publicado para 1s2p.

y se comparan con los valores correspondientes calculados sobre los calendarios oficiales. Los resultados de las comparaciones se presentan en las Tablas 8.2 a 8.5. Los calendarios generados pueden verse en el Anexo B. En general, en todos los casos, se observa que luego de la ejecución el MIP Gap resultante es de aproximadamente 65 %. Cabe señalar que, si bien este valor es relativamente elevado, esto no implica que las soluciones obtenidas sean de baja calidad. Por el contrario, los resultados muestran que el modelo es capaz de generar calendarios competitivos, que en varios casos superan a los oficiales en las métricas definidas. Esto sugiere que, a pesar de no haberse alcanzado la optimalidad matemática, las soluciones halladas son suficientemente buenas y relevantes desde el punto de vista práctico, cumpliendo adecuadamente con los objetivos planteados y aportando mejoras concretas en aspectos clave de la planificación académica.

En el caso correspondiente a 1s1p (Tabla 8.2), se observa que la métrica de estudiantes afectados resulta empeorada respecto al calendario publicado, así como también la métrica de previas. Sin embargo, en la métrica m_curricula se puede observar una pequeña mejora, mientras que en la métrica m_coincidencia se observa una mejora sustancial, lo que indica una mejor separación entre las UC.

Para 1s2p (Tabla 8.3), los valores obtenidos por el modelo son similares a los del calendario publicado, sin mejoras significativas en ninguna de las métricas estudiadas. Se observa que para el caso de m_estudiantes, se obtiene un valor levemente más alto que el calendario oficial, indicando un leve empeoramiento. De todas formas, si bien no se evidencian beneficios claros, los resultados se mantienen estables respecto a la solución oficial publicada.

2s1p	m_curricula	m_coincidencia	m_estudiantes	m_previas
Publicado	3.23	2.40	827.12	4.45
Generado	3.46	2.91	328.00	4.97

Tabla 8.4: Comparación de métricas entre el calendario generado y el calendario oficial publicado para 2s1p.

2s2p	m_curricula	m_coincidencia	m_estudiantes	m_previas
Publicado	3.64	3.42	475.73	4.69
Generado	3.96	3.32	200.76	4.58

Tabla 8.5: Comparación de métricas entre el calendario generado y el calendario oficial publicado para 2s2p.

En el caso de 2s1p (Tabla 8.4), el calendario generado presenta una mejora muy importante en la métrica de estudiantes afectados, que se reduce en más de un 60 %. Además, se observan leves mejoras en m_curricula y m_coincidencia, lo cual refuerza la calidad de la solución obtenida por el modelo para este caso particular. Para el caso de m_previas se observa una leve desmejora.

Finalmente, para 2s2p (Tabla 8.5), también se evidencia una mejora clara en la cantidad de estudiantes afectados, así como un incremento en m_curricula. El resto de las métricas se mantienen en niveles similares o levemente mejores que los del calendario oficial, consolidando el buen desempeño del modelo en escenarios diversos.

8.4.1. Incorporación de mejoras a partir de observaciones cualitativas

A partir de la ejecución obtenida para el caso 1s1p, se comparte el calendario generado con Ema, la cual realiza una evaluación cualitativa de los resultados. La evaluación general fue positiva, destacando la buena distribución de las UC que se encuentran en las currículas sugeridas de las diferentes carreras, así como la ausencia de conflictos evidentes entre cursos que suelen tomarse conjuntamente por un alto número de estudiantes.

Sin embargo, también se señalan varios aspectos que podrían representar oportunidades de mejora. En primer lugar, se identifican posibles inconsistencias entre los datos de inscripción utilizados y la capacidad real de los turnos, lo

que en algunos casos podría implicar una sobreasignación de estudiantes respecto al espacio disponible. Asimismo, se menciona la conveniencia de evitar que cursos pertenecientes al mismo instituto se ubiquen en el mismo turno, debido a limitaciones logísticas vinculadas a la disponibilidad de supervisores. Otro aspecto importante que se señala es la distribución temporal de las evaluaciones de las UC consideradas como más exigentes o más avanzadas, las cuales, en la planificación habitual, se intentan ubicar en la segunda mitad del calendario.

También se destacan particularidades curriculares de ciertos cursos que, si bien tienen un lugar sugerido en un semestre determinado, en la práctica pueden cursarse en múltiples momentos a lo largo de la carrera. Esto puede generar patrones de inscripción más dispersos, que es conveniente tener en cuenta a la hora de construir restricciones de coincidencia o superposición. Finalmente, se observa una cierta concentración de cursos en determinados turnos, lo cual no sólo puede generar problemas de capacidad en caso de imprevistos (como paros o falta de disponibilidad de salones), sino que también va en contra de una utilización más balanceada del calendario.

Todas estas observaciones pueden ser incorporadas al modelo a través de ajustes en los datos de entrada, agregando conjuntos de UC a diferentes conjuntos que agregan limitaciones de superposición. Con el objetivo de evaluar el impacto de estas modificaciones, se realiza una nueva ejecución para el caso 1s1p, incorporando:

- Reducción del factor de capacidad, para tener un mejor margen en la capacidad total utilizada. En las ejecuciones previas se utiliza un valor de 1, por lo que el modelo siempre tuvo disponible la capacidad total de los salones. Esta reducción debería mejorar la distribución, y lograr que no queden turnos muy llenos y otros muy vacíos.
- Agregar datos para evitar la coincidencia de UC del mismo instituto en el mismo turno.
- Agregar datos para forzar la ubicación de UC consideradas más exigentes hacia la segunda mitad del período.

En la ejecución inicial, se establece un tiempo máximo de tres horas de ejecución. Durante la misma, se observa que el solver logra encontrar soluciones factibles y mejorarlas constantemente, mejorando el valor de la función objetivo con cada nueva solución factible encontrada. Sin embargo, al cabo de una hora, se estanca, no logrando mejorar el valor de la función objetivo durante aproximadamente 30 minutos. Ante esta situación, se decide interrumpir la ejecución antes de alcanzar el límite establecido, y se reinicia la misma utilizando la mejor solución encontrada hasta ese momento como punto de partida. Se establece nuevamente un tiempo de ejecución límite de tres horas. En esta segunda

ls1p	m.curricula	m.coincidencia	m.estudiantes	m.previas
Publicado	3.82	3.59	560.00	5.19
Generado Original	3.85	4.38	686.23	5.45
Generado Corregido	4.62	4.35	517.73	4.78

Tabla 8.6: Comparación de métricas entre el calendario generado, el calendario oficial publicado y el calendario generado con correcciones para ls1p.

instancia, se constata que el solver logra mejorar la solución en los primeros minutos, lo que indica que el reinicio permite superar el estancamiento observado previamente.

Esta estrategia se justifica en que, tras un tiempo de ejecución extenso, el solver puede concentrar sus esfuerzos en zonas del espacio de búsqueda que, si bien son prometedoras desde el punto de vista de la cota dual, no conducen necesariamente a soluciones primales de mejor calidad. Al reiniciar la búsqueda con una solución inicial válida, se reconstruye el árbol de decisión, se aplican estrategias alternativas de ramificación y heurísticas, y se exploran regiones del espacio que previamente podrían haber sido descartadas o pasadas por alto. Esto permite, en muchos casos, superar el estancamiento observado y obtener mejoras adicionales en la solución. La efectividad de este enfoque ha sido abordada en estudios sobre la reoptimización de modelos MIP. En particular, [Patel \(2024\)](#) señala que la reutilización de soluciones factibles obtenidas previamente puede, por sí sola, contribuir a mejorar el desempeño del solucionador, incluso sin modificaciones en la configuración del modelo o en los parámetros del proceso de resolución.

Con esta metodología, se obtiene una solución que se observa en el Anexo [B.5](#). La comparación de los valores de las métricas con el calendario oficial y con el calendario previamente generado puede verse en la Tabla [8.6](#).

Analizando los resultados, se observa una mejora en todas las métricas respecto al calendario oficial publicado, y una disminución muy leve en el valor de m.coincidencia respecto al calendario generado originalmente. En particular, se destaca un aumento importante en la métrica asociada a la currícula sugerida, lo que indica que el calendario generado se adapta mejor a los estudiantes que siguen el trayecto recomendado en cada carrera. A su vez, se logra conservar un valor elevado en la métrica de coincidencia, reflejando una buena separación entre UC que suelen ser cursadas de manera conjunta.

8.4.2. Análisis cualitativo del calendario generado

Para realizar un análisis cualitativo del último calendario obtenido, incluyendo las correcciones sugeridas por Ema, se estudian dos casos de potenciales estudiantes que deben rendir evaluaciones durante el período.

Se comienza con un caso básico: un estudiante cursando el primer semestre de la carrera de Ingeniería en Computación, que debe rendir las UC Matemática Discreta 1, Cálculo Diferencial e Integral en una Variable y Programación 1. Según el calendario, el estudiante debería rendir, en orden, Programación 1 el lunes 29, Cálculo Diferencial e Integral en una Variable el sábado 4 y, finalmente, Matemática Discreta 1 el lunes 6. Esto implica que, una vez iniciado el período de evaluaciones, cuenta con tres días hasta su primera evaluación, luego cinco días hasta la segunda, y dos días más hasta la última.

Para tomar el caso de un estudiante más avanzado, se considera un estudiante que está cursando el séptimo semestre de la carrera Ingeniería en Sistemas de Comunicación, que debe rendir las UC Redes de Datos 2, Sistemas Operativos y Antenas y Propagación. En este caso, el calendario indica que debería rendir, en orden, Antenas y Propagación el sábado 27, Sistemas Operativos el lunes 6 y Redes de Datos 2 el jueves 9. Se observa que, luego de su primera evaluación, el estudiante cuenta con más de una semana antes de la segunda, y tres días más hasta la última, permitiéndole distribuir su preparación de forma razonable a lo largo del período.

En ambos casos, se aprecia que las evaluaciones se encuentran razonablemente espaciadas, evitando concentraciones excesivas en días consecutivos y otorgando márgenes adecuados entre exámenes, lo cual es uno de los objetivos buscados en la construcción del calendario.

Capítulo 9

Conclusiones y trabajo futuro

El presente trabajo aborda el desafío de generar calendarios de evaluaciones para la FIng mediante técnicas de programación lineal entera mixta. A lo largo del desarrollo, se logra construir un modelo matemático completo que captura los aspectos esenciales del problema, considerando tanto las restricciones institucionales como los objetivos de optimización relacionados con el bienestar estudiantil. Los resultados obtenidos prueban ser positivos, demostrando el potencial de la optimización matemática para mejorar procesos que tradicionalmente se realizan de forma manual. Este proyecto también evidencia que, si bien un sistema automatizado difícilmente podrá capturar todas las sutilezas que un experto humano incorpora en el proceso manual, la automatización puede constituir un apoyo sustancial que complemente el criterio y la experiencia humana en contextos complejos.

9.1. Principales logros y contribuciones

Los resultados experimentales muestran que, en la mayoría de los casos, los calendarios generados automáticamente presentan ventajas significativas respecto a los calendarios elaborados manualmente.

En dos de los cuatro casos analizados (2s1p y 2s2p), se logra una disminución sustancial (superior al 60 %) en la cantidad de estudiantes que deben enfrentar evaluaciones el mismo día o en días consecutivos, lo que representa una mejora en la experiencia académica de los estudiantes. Adicionalmente, los calendarios generados presentan, en general, una mejor separación entre evaluaciones de

unidades curriculares con alta coincidencia de estudiantes, como lo evidencia la mejora en la métrica `m_coincidencia`, lo que permite a los estudiantes distribuir mejor su tiempo de estudio.

Asimismo, el sistema demostró flexibilidad para incorporar rápidamente nuevas restricciones y consideraciones cualitativas, como se evidenció en la iteración realizada para el caso `1s1p` tras la retroalimentación recibida, lo que confirma su aplicabilidad en contextos reales y cambiantes.

La implementación computacional, basada en Python y utilizando la librería PuLP, proporciona una solución modular y extensible que puede ser fácilmente adaptada a diferentes períodos académicos y conjuntos de unidades curriculares. El uso de Gurobi como solver mostró un rendimiento superior en comparación con otras alternativas, logrando encontrar soluciones factibles dentro de tiempos razonables, incluso para instancias complejas del problema.

Uno de los aportes más valiosos de este trabajo ha sido el desarrollo de métricas de calidad (`m_curricula`, `m_coincidencia`, `m_estudiantes` y `m_previas`) que constituyen herramientas objetivas para evaluar calendarios (presentadas con mayor detalle en el Capítulo 7). Estas métricas no solo permiten comparar las soluciones generadas automáticamente con los calendarios históricos, sino que además representan un instrumento que Ema puede utilizar en su proceso regular de confección de calendarios, incluso cuando estos se realicen manualmente. La capacidad de cuantificar aspectos como la cantidad de estudiantes afectados por superposiciones cercanas o la distribución de evaluaciones según la currícula sugerida proporciona una base analítica que enriquece el proceso de toma de decisiones, aportando objetividad y transparencia a un proceso que tradicionalmente depende en gran medida de la intuición y la experiencia.

9.2. Limitaciones identificadas

A pesar de los resultados positivos, el trabajo también permitió identificar ciertas limitaciones que deben ser consideradas en futuras iteraciones.

Una primera limitación relevante es el tiempo de cómputo necesario para obtener soluciones de alta calidad. La complejidad inherente al problema hace que los tiempos de ejecución sean considerables, del orden de horas, lo que puede restringir la posibilidad de realizar múltiples iteraciones o ajustes en plazos acotados. Esta característica impone un desafío particular en contextos donde los calendarios deben confeccionarse rápidamente en respuesta a cambios de último momento.

Por otra parte, la calidad de los datos de entrada constituye un factor crítico para el desempeño del sistema. La efectividad del modelo depende en gran me-

dida de la precisión de las estimaciones de inscripción y de la información sobre coincidencia de estudiantes entre unidades curriculares. Errores o inexactitudes en estos datos de entrada pueden impactar negativamente en la calidad de los calendarios generados.

Otro aspecto a considerar es la existencia de múltiples objetivos dentro del problema. Como se evidenció en la calibración de los parámetros α y β en la Sección 8.3, mejorar ciertos aspectos del calendario puede deteriorar otros, obligando a realizar compromisos que deben ser cuidadosamente balanceados. La optimización multiobjetivo introduce así una dimensión adicional de complejidad en la definición de lo que constituye una solución de calidad.

Finalmente, algunas consideraciones cualitativas que Ema incorpora de manera intuitiva en el proceso manual de confección de calendarios son difíciles de formalizar en el modelo matemático. Preferencias específicas de ciertos departamentos, particularidades de determinadas unidades curriculares o necesidades coyunturales son ejemplos de aspectos que requieren flexibilidad y sensibilidad humanas, y que aún representan un desafío para su modelado sistemático.

9.3. Trabajo futuro

A partir de la experiencia adquirida y las lecciones aprendidas durante el desarrollo, se proponen varias líneas de trabajo futuro que podrían enriquecer y ampliar los resultados obtenidos.

Uno de los hallazgos más valiosos en las etapas finales del proyecto fue la efectividad de utilizar soluciones intermedias como punto de partida para nuevas ejecuciones del solver. Esta técnica, implementada con éxito en el caso 1s1p revisado, permitió superar estancamientos en la búsqueda y obtener mejoras significativas en la calidad de las soluciones. Sin embargo, para ese caso particular, se realizó de forma totalmente manual, tomando la mejor solución obtenida en la primera ejecución y utilizándola como punto de partida de una nueva ejecución. Se propone sistematizar e incorporar este procedimiento como parte estándar del proceso de generación de calendarios, aplicándolo de manera consistente a todos los períodos. De esta forma, se podría, por ejemplo, detener automáticamente la ejecución cuando el gap de optimalidad, o el valor de la función objetivo, no mejoran por un tiempo predeterminado, y comenzar automáticamente una nueva ejecución partiendo de la mejor solución encontrada hasta el momento como punto de partida.

Una limitación importante del sistema actual es la ausencia de una interfaz gráfica que permita a usuarios no técnicos como Ema interactuar directamente con la herramienta. El desarrollo de una interfaz intuitiva que facilite la visualización de calendarios, la modificación de parámetros, la edición manual de

asignaciones y la comparación de alternativas constituye una prioridad para la adopción efectiva de la solución en el entorno institucional. Actualmente, la generación de un nuevo calendario requiere conocimientos técnicos para modificar los archivos de entrada y ejecutar el código, lo cual representa una barrera significativa para la adopción del sistema por parte de usuarios como Ema. Una interfaz gráfica no solo facilitaría el uso de la herramienta, sino que también permitiría una integración más fluida del sistema en los procesos institucionales existentes. Además, se podría desarrollar una funcionalidad adicional que permita a los usuarios visualizar en tiempo real el impacto de mover evaluaciones dentro del calendario, utilizando un sistema interactivo de tipo drag and drop. Por ejemplo, Ema podría arrastrar una evaluación de una fecha a otra dentro del calendario y, de manera inmediata, observar cómo este cambio afecta las métricas relevantes. Esta funcionalidad facilitaría la toma de decisiones informadas y permitiría ajustar el calendario de manera dinámica, adaptándose mejor a las necesidades institucionales y de los estudiantes.

El modelo matemático también podría beneficiarse de mejoras adicionales, incorporando consideraciones como la distribución balanceada de evaluaciones a lo largo de todo el período, evitando concentraciones excesivas en ciertos días o turnos. Esto, si bien ya es posible controlar parcialmente mediante el parámetro *fac_cp*, puede resultar interesante incorporar restricciones explícitas que lo controlen.

Como se menciona anteriormente, este es un problema que se presenta en diversas instituciones, no únicamente en la FIng, por lo que otra línea de trabajo interesante sería la utilización y evaluación de esta metodología en otros contextos académicos. Esto permitiría validar la versatilidad del modelo y enriquecerlo con las particularidades de diferentes entornos educativos. Finalmente, se considera interesante realizar pruebas extensivas con estudiantes y docentes para evaluar la percepción de calidad de los calendarios generados y recopilar retroalimentación para futuras mejoras. Esta validación con usuarios finales permitiría afinar el modelo y asegurar que las soluciones generadas satisfacen adecuadamente las necesidades y expectativas de la comunidad académica.

De este modo, la experiencia acumulada a lo largo del proyecto pone de manifiesto que la integración equilibrada entre el conocimiento experto y las capacidades computacionales representa la vía más prometedora para abordar este tipo de problemas. El objetivo final no es reemplazar el criterio humano, sino complementarlo, permitiendo que expertos como Ema puedan dedicar más tiempo a consideraciones cualitativas y ajustes finos, mientras las restricciones básicas y objetivos de optimización son manejados de manera eficiente por el sistema automatizado. El desarrollo de una interfaz gráfica representaría un paso crucial para lograr esta integración efectiva.

En última instancia, el éxito de este tipo de herramientas radica en su capacidad para integrarse en los procesos institucionales existentes, aportando efi-

ciencia sin sacrificar la calidad y adaptabilidad que caracterizan la planificación académica en la FIng. Los avances logrados a través de este proyecto sientan las bases para una gestión académica más eficiente, equitativa y centrada en el bienestar de los estudiantes.

Referencias

- Akkan, C., Külünk, M. E., y Koças, C. (2016). Finding robust timetables for project presentations of student teams. *European Journal of Operational Research*, 249(2), 560–576.
- Al-Yakoob, S. M., Sherali, H. D., y Al-Jazzaf, M. (2010). A mixed-integer mathematical modeling approach to exam timetabling. *Computational Management Science*, 7(1), 19–46. doi: 10.1007/s10287-007-0066-8
- Andrade, P. R. L., Steiner, M. T. A., y Góes, A. R. T. (2019). Optimization in timetabling in schools using a mathematical model, local search and iterated local search procedures. *Gestão & Produção*, 26(4), e3421. doi: 10.1590/0104-530X3241-19
- Buscador timbó foco. (2009). Descargado 2024-06-06, de <https://timbo.org.uy/home>
- Canseco-González, A., Sánchez-Partida, D., Zuñiga-Alcaraz, C., y Olivares-Benitez, E. (2016). Aplicación de programación lineal para la asignación de horarios en una institución educativa mexicana. *Revista Ingeniería Industrial*, 15(2), 135–146.
- Carle, F., de Sierra, L., y Spira, F. (2025). *Propuesta automática de calendario de evaluaciones en facultad de ingeniería*. Descargado de <https://github.com/FacundoSpira/proyecto-grado>
- Chvátal, V. (1999). *Linear programming*. W.H. Freeman and Company.
- Coin-or: Computational infrastructure for operations research. (2016). Descargado 2025-01-11, de <https://www.coin-or.org/>
- Colibri: Facultad de ingeniería. (2014). Descargado 2024-06-06, de <https://www.colibri.udelar.edu.uy/jspui/handle/20.500.12008/22>
- da Fonseca, G. H. G., Santos, H. G., Toffolo, T. A. M., Brito, S. S., y Souza, M. J. F. (2016). Goal solver: a hybrid local search based solver for high school timetabling. *Annals of Operations Research*, 239(1), 77–97.

- De Werra, D. (1985). An introduction to timetabling. *European Journal of Operational Research*, 19(2), 151–162. doi: 10.1016/0377-2217(85)90167-5
- Facultad de Ingeniería, U. d. l. R. (2025). *Cursos*. Descargado de <https://www.fing.edu.uy/es/bedelia/cursos-de-grado-posgrado-y-actualizaci%C3%B3n-profesional> (Accedido el 10 de mayo de 2025)
- Feng, X., Wang, X., y Zhang, H. (2014). Passenger transfer efficiency optimization modelling research with simulations. *International Journal of Simulation Modelling*, 13(2), 210–218. doi: 10.2507/IJSIMM13(2)CO7
- Feng, Z., Cao, C., Liu, Y., y Zhou, Y. (2018). A multiobjective optimization for train routing at the high-speed railway station based on tabu search algorithm. *Mathematical Problems in Engineering*, 2018, 1–22. doi: 10.1155/2018/8394397
- Forrest, J., Vigerske, S., Ralphs, T., Forrest, J., Hafer, L., jpfasano, ... to st (2024, agosto). *coin-or/clp: Release releases/1.17.10*. Zenodo. Descargado de <https://doi.org/10.5281/zenodo.13347196> doi: 10.5281/zenodo.13347196
- Fourer, R., Gay, D. M., y Kernighan, B. W. (1990). A modeling language for mathematical programming. *Management Science*, 36(5), 519–554. Descargado de https://ampl.com/_archive/first-website/REFS/amlopt.pdf doi: 10.1287/mnsc.36.5.519
- Free Software Foundation. (2025). *Gnu linear programming kit*. <https://www.gnu.org/software/glpk/>. (Accedido: 10 de mayo de 2025)
- Gurobi. (2024, 08). *Mip solvers - gurobi optimization*. Descargado 2025-04-06, de <https://www.gurobi.com/faqs/mip-solvers/>
- Gurobi. (2025, 03). *Mixed-integer programming (mip) – a primer on the basics*. Descargado 2025-04-06, de <https://www.gurobi.com/resources/mixed-integer-programming-mip-a-primer-on-the-basics/>
- Gurobi Optimization, LLC. (2024). *Gurobi Optimizer Reference Manual*. Descargado de <https://www.gurobi.com>
- Ibm ilog cplex optimization studio. (2022, 12). Descargado 2025-01-12, de <https://www.ibm.com/docs/en/icos/22.1.1?topic=optimizers-users-manual-cplex>
- itertools — functions creating iterators for efficient looping. (2025). Descargado 2025-01-12, de <https://docs.python.org/3/library/itertools.html>
- Kitchenham, B. (2004). *Procedures for performing systematic reviews* (Inf. Téc.). Software Engineering Group, Department of Computer Science, Keele University: Keele University and National ICT Australia Ltd. (Technical Report TR/SE-0401)

- Lee, J. (2024). *A first course in linear optimization* (4.0.8 ed.). Re-ex Press. Descargado 2025-04-06, de https://github.com/jon77lee/JLee_LinearOptimizationBook
- Lee, M. K. (2024, 12). *Integer programming: The cutting plane algorithm (gomory's cutting plane) — medium*. Descargado de https://medium.com/@minkyunglee_5476/integer-programming-the-cutting-plane-algorithm-26bbabf04815
- Machado, E., y Baráibar, I. (2021). *Confección del fixture para competencias artísticas: una aplicación al carnaval uruguayo* (Proyecto de Grado). Facultad de Ingeniería, Universidad de la República, Montevideo, Uruguay.
- Micarrera. (2019). Descargado 2025-02-10, de https://github.com/cedarcode/mi_carrera
- Misra, R., y Mckean, M. (2000, 01). College students'academic stress and its relation to their anxiety, time management, and leisure satisfaction. *American journal of health studies*, 16, 41-51.
- Mitchell, S., O'Sullivan, M., y Dunning, I. (2011, September). Pulp: A linear programming toolkit for python. *Stuart Mitchell Consulting and Department of Engineering Science, The University of Auckland, Auckland, New Zealand*. Descargado de <http://www.coin-or.org/PuLP/>
- Neri, P., y Peirano, F. (2024). *Herramienta de calendarización de parciales fmg* (Proyecto de Grado). Facultad de Ingeniería, Universidad de la República, Montevideo, Uruguay.
- pandas development team, T. (2024, septiembre). *pandas-dev/pandas: Pandas*. Zenodo. Descargado de <https://doi.org/10.5281/zenodo.13819579> doi: 10.5281/zenodo.13819579
- Patel, K. K. (2024). *Progressively strengthening and tuning mip solvers for reoptimization*. Descargado de <https://arxiv.org/abs/2308.08986>
- Post, G., Ahmadi, S., y Geertsema, F. (2012). Cyclic transfers in school timetabling. *OR Spectrum*, 34(1), 133–154. doi: 10.1007/s00291-010-0227-y
- Shanno, D. F., y Weil, R. L. (1971). 'linear' programming with absolute-value functionals. *Operations Research*, 19(1), 120–124. Descargado 2025-01-05, de <http://www.jstor.org/stable/168871>
- Vasily. (2019, 03). *Introduction to mixed integer programming*. Efidgy. Descargado 2025-04-06, de <https://medium.com/efidgy/introduction-to-mixed-integer-programming-3560a704f91e>
- Vermuyten, H., Lemmens, S., Marques, I., y Beliën, J. (2016). Developing compact course timetables with optimized student flows. *European Journal of Operational Research*, 251(3), 651–661.

Wikipedia contributors. (2025). *Domain-specific language*. https://en.wikipedia.org/wiki/Domain-specific_language. (Accedido: 10 de mayo de 2025)

Anexo A

Categorización de la literatura revisada

Texto	Problema abordado		Metodología utilizada				
	OH	OF	PM	PE	H	EH	MO
A mixed-integer mathematical modeling approach to exam timetabling Al-Yakoob y cols. (2010)	X		X	X		X	
A Multiobjective Optimization for Train Routing at the High-Speed Railway Station Based on Tabu Search Algorithm Z. Feng y cols. (2018)		X	X				X
An introduction to timetabling De Werra (1985)	X		X		X		
Aplicación de Programación Lineal para la Asignación de Horarios en una Institución Educativa Mexicana Canseco-González y cols. (2016)	X		X	X			
<i>Continúa en la siguiente página</i>							

Texto	Problema abordado		Metodología utilizada				
	OH	OF	PM	PE	H	EH	MO
Confección del fixture para competencias artísticas: una aplicación al Carnaval uruguayo Machado y Baráibar (2021)	X		X	X			
Cyclic Transfers in School Timetabling Post y cols. (2012)	X				X		
Developing compact course timetables with optimized student flows Vermuyten y cols. (2016)		X	X	X			
Finding robust timetables for project presentations of student teams Akkan y cols. (2016)	X		X	X		X	X
GOAL solver: a hybrid local search based solver for high school timetabling da Fonseca y cols. (2016)	X				X	X	
Herramienta de calendarización de parciales FING Neri y Peirano (2024)	X					X	
Optimization in timetabling in schools using a mathematical model, local search and Iterated Local Search procedures An-drade y cols. (2019)	X		X				
Passenger transfer efficiency optimization modelling re-search with simulations X. Feng y cols. (2014)		X	X		X		X

OH: optimización de horarios

OF: optimización de flujos

PM: programación matemática

H: uso de heurísticas y/o metaheurísticas

PE: programación entera mixta

EH: enfoque híbrido

MO: múltiples objetivos

Anexo B

Calendarios generados

A continuación se presentan los calendarios generados para los diferentes casos de prueba:

- Primer semestre 2024, primeros parciales (**1s1p**).
- Primer semestre 2024, segundos parciales (**1s2p**).
- Segundo semestre 2024, primeros parciales (**2s1p**).
- Segundo semestre 2024, segundos parciales (**2s2p**).
- Primer semestre 2024, primeros parciales, luego de la incorporación de nuevos datos a los conjuntos tras la revisión de Ema (**1s1p revisado**).

B.1. Calendario 1s1p

	Turno 1	Turno 2	Turno 3
Sábado 27/4	MATEMATICA DISCRETA 2 (1026) MECANICA NEWTONIANA (1122) SEÑALES Y SISTEMAS (1457) INT. A LA INVESTIGACION DE OPERACIONES (1610) TECNOLOGIA DEL HORMIGON (2371) INGENIERIA DE LAS REACCIONES QUIMICAS 1 (Q69)	GEOMETRIA Y ALGEBRA LINEAL 1 (1030) ADMINISTRACION DE OPERACIONES (1915) IMAGENES MEDICAS: ADQUISICION, INSTRUMENTACION Y GESTION (5707) TECNOLOGIAS DE REDES Y SERVICIOS DE TELECOMUNICACIONES (5866) TRATAMIENTO BIOLOGICO DE EFLUENTES Y RESID.SOLIDOS (Q56) INGENIERIA BIOQUIMICA (Q81)	
Domingo 28/4			
Lunes 29/4	PROGRAMACION 2 (1321) SISTEMAS OPERATIVOS (1537) TRANSFERENCIA DE CALOR 2 (1824) ESTRUCTURAS DE ACERO (2393) ELECTROTECNICA (5850) TERMODINAMICA APL. A LA ING.DE PROCESOS (Q21)	COMPORTAMIENTO MECANICO DE MATERIALES 1 (1764)	ELASTICIDAD (2366)
Martes 30/4	FISICA 1 (1151) PROGRAMACION 1 (1322) ENERGIA 1- COMBUSTION (1850)	MECANICA ESTRUCTURAL (2381)	CALCULO VECTORIAL (1063) ENERGIA 2:GEN ENERG PLANTAS VAPOR Y GAS (2394)
Miercoles 1/5			
Jueves 2/5	GEOMETRIA Y ALGEBRA LINEAL 2 (1031) ELECTROTECNICA I (2108)	DISEÑO DE PROCESOS QUIMICOS (Q196)	ADMINISTRACION GENERAL PARA INGENIEROS (1944)
Viernes 3/5	FISICA 3 (1153)		ELEMENTOS DE MECANICA DE LOS FLUIDOS (1805) REDES DE DATOS 2 (5855)
Sábado 4/5	FISICA 2 (1152) ELECTRONICA AVANZADA 2 (5717)	CALCULO DIF. E INTEGRAL EN UNA VARIABLE (1061) SUBESTACIONES EN MEDIA TENSION (5514)	
Domingo 5/5			
Lunes 6/5	PROBABILIDAD Y ESTADISTICA (1025) ELEMENTOS DE MAQUINAS (1720) CALIDAD DE AGUAS (1854)	CALCULO DIF. E INTEGRAL EN VARIAS VARIABLES (1062) DINAMICA Y CONTROL DE PROCESOS (Q94)	
Martes 7/5	ELECTRONICA DE POTENCIA (5609)		FUNCIONES DE VARIABLE COMPLEJA (1066) HORMIGON ESTRUCTURAL 1 (2382) INT. AL CONTROL INDUSTRIAL (2707)
Miercoles 8/5		PATOLOGIA DE LAS ESTRUCTURAS (2389)	INSTRUMENTACION INDUSTRIAL (2706) TECNOLOGIA Y SERV. INDUSTRIALES 1 (Q71)
Jueves 9/5	TEORIA DE LENGUAJES (1325) INT. A LA INGENIERIA DE SOFTWARE (1716) GESTION DE CALIDAD (1954) REDES ELECTRICAS (5508)	LOGICA (1027) MEDIDAS ELECTRICAS (1461) PRINCIPIOS DE QUIMICA GENERAL (1620) INT. A LA MECANICA DE SUELOS (2207) ANTENAS Y PROPAGACION (5817) TRANSFERENCIA DE CALOR Y MASA 1 (Q74)	MATEMATICA DISCRETA 1 (1023) MATERIALES Y ENSAYOS (1508) INT. A LA CIENCIA DE MATERIALES (1723) METALURGIA DE TRANSFORMACION (1724) MAQUINAS PARA FLUIDOS I (1856) FENOMENOS DE TRANSPORTE EN ING. DE PROCESOS (Q22)

Figura B.1: Calendario generado para los primeros parciales del primer semestre de 2024

B.2. Calendario 1s2p

	Turno 1	Turno 2	Turno 3
Jueves 4/7	MATEMATICA DISCRETA 1 (1023) TEORIA DE LENGUAJES (1325) SEÑALES Y SISTEMAS (1457) ELEMENTOS DE GESTION LOGISTICA (2244) INT. A LA PREVENCIÓN DE RIESGOS LABORALES (Q16) METALURGIA DE TRANSFORMACION (TIM54)	MATEMATICA DISCRETA 2 (1026) PROGRAMACION LOGICA (1340) MEDIDAS ELECTRICAS (1461) INT. A LA CIENCIA DE MATERIALES (1723) TRATAMIENTO DE EFLUENTES Y RESIDUOS SOLIDOS (Q104) INGENIERIA DE LAS REACCIONES QUIMICAS 1 (Q69)	COMUNICACIONES DIGITALES (1462) MOTORES DE COMBUSTION INTERNA (1818) INSTRUMENTACION INDUSTRIAL (2706) TERMODINAMICA APL. A LA ING.DE PROCESOS (Q21) MATEMATICA 1 (TIM20) ELEMENTOS DE MAQUINAS (TIM53)
Viernes 5/7	MECANICA NEWTONIANA (1122) PROGRAMACION FUNCIONAL (1354) ELEMENTOS DE MECANICA DE LOS FLUIDOS (1805) MAQUINAS PARA FLUIDOS I (1856) HORMIGON ESTRUCTURAL 1 (2382) TRANSFERENCIA DE CALOR Y MASA 2 (Q68)	MATEMATICA 3 (TIM22)	ANTENAS Y PROPAGACION (5817) CONTROL DE CALIDAD (TIM61)
Sabado 6/7	CALCULO DIF. E INTEGRAL EN UNA VARIABLE (1061) PROGRAMACION PARA INGENIERIA ELECTRICA (1307)	LOGICA (1027) FISICA 2 (1152) SISTEMAS OPERATIVOS (1537) INGENIERIA BIOQUIMICA (Q81) MAQUINAS ELECTRICAS (TIM71)	
Domingo 7/7			
Lunes 8/7	FISICA 1 (1151) INT. A LA MECANICA DE SUELOS (2207) PATOLOGIA DE LAS ESTRUCTURAS (2389) INT. AL CONTROL INDUSTRIAL (2707)	FISICA 3 (1153) INGLES TECNICO (TIM81)	COMPORTAMIENTO MECANICO DE MATERIALES 1 (1764)
Martes 9/7	COMPORTAMIENTO MEC. DE MATERIALES (TIM52)	PROGRAMACION 2 (1321) & ADMINISTRACION DE OPERACIONES (1915)	ELECTRONICA AVANZADA 2 (5717)
Miercoles 10/7	PROGRAMACION 1 (1373) MATERIALES Y ENSAYOS (1508)	CALIDAD DE AGUAS (1854)	PLAN. ESTRAT. Y ESTRATEGIA COMPETITIVA (2391) SUBESTACIONES EN MEDIA TENSION (5514)
Jueves 11/7	PROGRAMACION 4 (1324) INT. A LA INGENIERIA DE SOFTWARE (1716)	CALCULO VECTORIAL (1063)	TRANSFERENCIA DE CALOR (TIM41)
Viernes 12/7	ENERGIA 2: GEN ENERG PLANTAS VAPOR Y GAS (2394)	ELECTROTECNICA I (2108)	GEOMETRIA Y ALGEBRA LINEAL 2 (1031) ADMINISTRACION GENERAL PARA INGENIEROS (1944)
Sabado 13/7	GEOMETRIA Y ALGEBRA LINEAL 1 (1030) MECANICA ESTRUCTURAL (2381) REDES DE DATOS 2 (5855) TECNOLOGIAS DE REDES Y SERVICIOS DE TELECOMUNICACIONES (5866) MAQUINAS PARA FLUIDOS (TIM42) INT. A LA CIENCIA DE LOS MATERIALES (TIM50)	PROBABILIDAD Y ESTADISTICA (1025) INT. A LA INVESTIGACION DE OPERACIONES (1650) TRANSFERENCIA DE CALOR 2 (1824) ELECTRONICA DE POTENCIA (5609) ELECTRONICA FUNDAMENTAL (5715) FISICA 2 (TIM31)	
Domingo 14/7			
Lunes 15/7	CALCULO DIF. E INTEGRAL EN VARIAS VARIABLES (1062) GESTION DE CALIDAD (1954) REDES ELECTRICAS (5508) MATEMATICA INICIAL (MI2) DINAMICA Y CONTROL DE PROCESOS (Q94) INSTRUMENTACION Y CONTROL (TIM72)	TECNOLOGIA DEL HORMIGON (2371) ELECTROTECNICA (5850) TECNOLOGIA Y SERV. INDUSTRIALES 1 (Q71) MATEMATICA 2 (TIM21) FISICA 1 (TIM30) METALURGIA FISICA (TIM51)	INT. A LOS MICROPROCESADORES (1513) PRINCIPIOS DE QUIMICA GENERAL (1620) METALURGIA DE TRANSFORMACION (1724) ENERGIA 1- COMBUSTION (1850) ELASTICIDAD (2366) FENOMENOS DE TRANSPORTE EN ING. DE PROCESOS (Q22)

Figura B.2: Calendario generado para los segundos parciales del primer semestre de 2024

B.3. Calendario 2s1p

	Turno 1	Turno 2	Turno 3
Sábado 21/9	MATEMATICA DISCRETA 1 (1023) FISICA TERMICA (1123) ELECTROMAGNETISMO (1128) VIBRACIONES Y ONDAS (1144) ELECTROTECNICA II (2109) GEOLOGIA DE INGENIERIA (2206)	MECANICA NEWTONIANA (1122) PROGRAMACION 3 (1323) GENERADORES DE VAPOR (1815) SISTEMAS OLEOHIDRAULICOS Y NEUMATICOS (2200) INSTALACIONES ELECTRICAS (5507) TECNOLOGIA Y SERV. INDUSTRIALES 2 (Q100)	
Domingo 22/9			
Lunes 23/9	PROGRAMACION 1 (1373) SEÑALES ALEATORIAS Y MODULACION (1460) TRANSFERENCIA DE CALOR 1 (1823) INT. AL TRANSPORTE (2510) TECNOLOGIA DE SERVICIOS AUDIOVISUALES (5851) INT. A LA INGENIERIA BIOQUIMICA (Q77C)	DISEÑO DE REDES DE COND.EN ING.SANITARIA (1852) RESISTENCIA DE MATERIALES 1 (2368) INGENIERIA CLINICA (5718) TRANSFERENCIA DE CALOR Y MASA 1 (Q74) MATEMATICA 1 (TIM20) FISICA 1 (TIM30)	REDES DE COMPUTADORAS (1446) GENERADORES DE VAPOR (TIM44) COSTOS PARA INGENIERIA (TIM60) INGLES TECNICO (TIM81)
Martes 24/9	FLUIDODINAMICA (Q66)	MET. COMPUT.APLIC.AL CALCULO ESTRUCTURAL (2380)	CALCULO VECTORIAL (1063) MOTORES DE COMB. INT. Y TURBINAS DE GAS (TIM43)
Miércoles 25/9	CALCULO DIF. E INTEGRAL EN UNA VARIABLE (1061) & MATEMATICA INICIAL (MI2)		MATEMATICA DISCRETA 2 (1026) FUNDAMENTOS DE BASES DE DATOS (1911) FUNDAMENTOS DE APRENDIZAJE AUTOMATICO Y RECONOCIMIENTO DE PATRONES (5852)
Jueves 26/9	METODOS NUMERICOS (1033) CALCULO DIF. E INTEGRAL EN VARIAS VARIABLES (1062)		SUBESTACIONES EN MEDIA TENSION (5514) DISEÑO DE SISTEMAS MEDICOS IMPLANTABLES ACTIVOS (5714)
Viernes 27/9	REFRIGERACION INDUSTRIAL (TIM45)		PROBABILIDAD Y ESTADISTICA (1025) GEOMETRIA Y ALGEBRA LINEAL 1 (1030) CONTROL DE CALIDAD (1510)
Sábado 28/9	GEOMETRIA Y ALGEBRA LINEAL 2 (1031)	METALURGIA FISICA (1719) SISTEMAS Y CONTROL (5905) INGENIERIA DE LAS REACCIONES QUIMICAS 2 (Q70) SEGURIDAD INDUSTRIAL (TIM65)	
Domingo 29/9			
Lunes 30/9	TEORIA DE CIRCUITOS (1456) MATEMATICA 2 (TIM21) GESTION DEL MANTENIMIENTO (TIM62) ESTUDIO DEL TRABAJO (TIM63) INT. A LA ELECTROTECNIA (TIM70)	INT. A LA INGENIERIA DE PROCESOS (Q59B)	FISICA 3 (1153) PROGRAMACION 2 (1321) DINAMICA DE MAQUINAS Y VIBRACION (1718) MAQUINAS ELECTRICAS (5602) INGENIERIA BIOMEDICA (5710) ELEMENTOS DE MAQUINAS (TIM53)
Martes 1/10	REFRIGERACION (1638) HIGIENE Y SERV. DE PLANTAS PROC. DE ALIMENTOS (A65) DINAMICA Y CONTROL DE PROCESOS (Q94) INT. A LA CIENCIA DE LOS MATERIALES (TIM50) METALURGIA FISICA (TIM51) PROYECTO DE INVERSION (TIM64)	MECANICA DE LOS FLUIDOS (1226) RESISTENCIA DE MATERIALES 2 (1312) COMPUTACION 1 (1411) TIEMPOS Y METODOS (2038) REDES DE DATOS 1 (5824) INT. A LA TERMODINAMICA (TIM32)	FISICA 1 (1151) FISICA 2 (1152) COMPORTAMIENTO MECANICO DE MATERIALES 2 (1765) COSTOS PARA INGENIERIA (1922) MECANICA APLICADA (ING.QUÍM) (Q73) INT. A LA MECANICA DE LOS FLUIDOS (TIM40)
Miércoles 2/10	INT. A LAS ECUACIONES DIFERENCIALES (1064) INT. A LA INGENIERIA SANITARIA (2315)	ARQUITECTURA DE COMPUTADORAS (1443)	

Figura B.3: Calendario generado para los primeros parciales del segundo semestre de 2024

B.4. Calendario 2s2p

	Turno 1	Turno 2	Turno 3
Viernes 22/11	COMPUTACION 1 (1411) TRANSFERENCIA DE CALOR 1 (1823) INT. A LA INGENIERIA SANITARIA (2315) DISEÑO DE SISTEMAS MEDICOS IMPLANTABLES ACTIVOS (5714) SISTEMAS Y CONTROL (5905) HIGIENE Y SERV. DE PLANTAS PROC. DE ALIMENTOS (A65)	FISICA 1 (1151) DISEÑO LOGICO (1512) GENERADORES DE VAPOR (1815) INT. AL TRANSPORTE (2510) FISICA 1 (TIM30) INT. A LA MECANICA DE LOS FLUIDOS (TIM40)	RESISTENCIA DE MATERIALES 2 (1312) REFRIGERACION (1638) DISEÑO DE REDES DE COND.EN ING.SANITARIA (1852) COSTOS PARA INGENIERIA (1922) ESTUDIO DEL TRABAJO (TIM63) INGLES TECNICO (TIM81)
Sábado 23/11	PROGRAMACION 2 (1321) TEORIA DE CIRCUITOS (1456) FLUIDODINAMICA (Q66) DINAMICA Y CONTROL DE PROCESOS (Q94) INT. A LA ELECTROTECNIA (TIM70)	CALCULO VECTORIAL (1063) SUBESTACIONES EN MEDIA TENSION (5514) INT. A LA CIENCIA DE LOS MATERIALES (TIM50) ELEMENTOS DE MAQUINAS (TIM53)	
Domingo 24/11			
Lunes 25/11		SEÑALES ALEATORIAS Y MODULACION (1460) ELECTROTECNICA II (2109) MAQUINAS ELECTRICAS (5602) INGENIERIA BIOMEDICA (5710) METALURGIA FISICA (TIM51)	GEOMETRIA Y ALGEBRA LINEAL 2 (1031) TECNOLOGIA Y SERV. INDUSTRIALES 2 (Q100)
Martes 26/11	GEOMETRIA Y ALGEBRA LINEAL 1 (1030) MATEMATICA 2 (TIM21)	TIEMPOS Y METODOS (2038) INT. A LA TERMODINAMICA (TIM32)	MET. COMPUT.APLIC.AL CALCULO ESTRUCTURAL (2380)
Miércoles 27/11	INT. A LAS ECUACIONES DIFERENCIALES (1064)		PROBABILIDAD Y ESTADISTICA (1025) CALCULO DIF. E INTEGRAL EN VARIAS VARIABLES (1062)
Jueves 28/11	CALCULO DIF. E INTEGRAL EN UNA VARIABLE (1061) MATEMATICA INICIAL (MI2)	GENERADORES DE VAPOR (TIM44)	CONTROL DE CALIDAD (1510) MATEMATICA 1 (TIM20)
Viernes 29/11	REDES DE DATOS 1 (5824) INT. A LA INGENIERIA BIOQUIMICA (Q77C)	MOTORES DE COMB. INT. Y TURBINAS DE GAS (TIM43)	
Sábado 30/11	COMPORTAMIENTO MECANICO DE MATERIALES 2 (1765) INT. AL PROCESAMIENTO DE LENG. NATURAL (1828) TRANSFERENCIA DE CALOR Y MASA 1 (Q74)	FISICA 3 (1153) PROGRAMACION 1 (1373) ARQUITECTURA DE COMPUTADORAS (1443) REDES DE COMPUTADORAS (1446)	
Domingo 1/12			
Lunes 2/12	VIBRACIONES Y ONDAS (1144) INSTALACIONES ELECTRICAS (5507) ELECTRONICA AVANZADA 1 (5716)	PROGRAMACION 3 (1323) INT. A LA INGENIERIA DE PROCESOS (Q59B) GESTION DEL MANTENIMIENTO (TIM62)	FISICA TERMICA (1123) REFRIGERACION INDUSTRIAL (TIM45)
Martes 3/12	MATEMATICA DISCRETA 1 (1023) ELECTROMAGNETISMO (1128) PROGRAMACION FUNCIONAL (1354) DINAMICA DE MAQUINAS Y VIBRACION (1718) RESISTENCIA DE MATERIALES 1 (2368)	MAQUINAS PARA FLUIDOS II (1859) TEORIA DE RESTRICCIONES (1927) GEOLOGIA DE INGENIERIA (2206) INGENIERIA CLINICA (5718) INGENIERIA DE LAS REACCIONES QUIMICAS 2 (Q70) SEGURIDAD INDUSTRIAL (TIM65)	METODOS NUMERICOS (1033) MECANICA DE LOS FLUIDOS (1226) METALURGIA FISICA (1719) SISTEMAS OLEOHIDRAULICOS Y NEUMATICOS (2200) COSTOS PARA INGENIERIA (TIM60) PROYECTO DE INVERSION (TIM64)
Miércoles 4/12	MATEMATICA DISCRETA 2 (1026) MECANICA NEWTONIANA (1122) FISICA 2 (1152) FUNDAMENTOS DE BASES DE DATOS (1911) FUNDAMENTOS DE APRENDIZAJE AUTOMATICO Y RECONOCIMIENTO DE PATRONES (5852) MECANICA APLICADA (ING.QUÍM) (Q73)		

Figura B.4: Calendario generado para los segundos parciales del segundo semestre de 2024

B.5. Calendario 1s1p revisado

	Turno 1	Turno 2	Turno 3
Sábado 27/4	GEOMETRIA Y ALGEBRA LINEAL 2 (1031) COMPORTAMIENTO MECANICO DE MATERIALES 1 (1764) TRANSFERENCIA DE CALOR 2 (1824) ADMINISTRACION DE OPERACIONES (1915) TECNOLOGIA DEL HORMIGON (2371) DINAMICA Y CONTROL DE PROCESOS (Q94)	GEOMETRIA Y ALGEBRA LINEAL 1 (1030) CALIDAD DE AGUAS (1854) ENERGIA 2: GEN ENERG PLANTAS VAPOR Y GAS (2394) REDES ELECTRICAS (5508) IMAGENES MEDICAS: ADQUISICION, INSTRUMENTACION Y GESTION (5707) ANTENAS Y PROPAGACION (5817)	
Domingo 28/4			
Lunes 29/4	PROGRAMACION 2 (1321) MEDIDAS ELECTRICAS (1461) MATERIALES Y ENSAYOS (1508) ENERGIA 1- COMBUSTION (1850) ESTRUCTURAS DE ACERO (2393) INGENIERIA DE LAS REACCIONES QUIMICAS 1 (Q69)	PROGRAMACION 1 (1373) PRINCIPIOS DE QUIMICA GENERAL (1620) TERMODINAMICA APL. A LA ING. DE PROCESOS (Q21)	
Martes 30/4	INT. A LA INVESTIGACION DE OPERACIONES (1650) MECANICA ESTRUCTURAL (2381)	FISICA 3 (1153)	GESTION DE CALIDAD (1954) DISEÑO DE PROCESOS QUIMICOS (Q196)
Miércoles 1/5			
Jueves 2/5	ADMINISTRACION GENERAL PARA INGENIEROS (1944) ELECTROTECNICA I (2108)	PROBABILIDAD Y ESTADISTICA (1025)	PATOLOGIA DE LAS ESTRUCTURAS (2389)
Viernes 3/5	ELEMENTOS DE MAQUINAS (1720)		CALCULO DIF. E INTEGRAL EN VARIAS VARIABLES (1062) FUNCIONES DE VARIABLE COMPLEJA (1066)
Sábado 4/5	CALCULO DIF. E INTEGRAL EN UNA VARIABLE (1061)	CALCULO VECTORIAL (1063) TEORIA DE LENGUAJES (1325) ELECTRONICA AVANZADA 2 (5717)	
Domingo 5/5			
Lunes 6/5	LOGICA (1027) MAQUINAS PARA FLUIDOS I (1856) ELASTICIDAD (2366)	SISTEMAS OPERATIVOS (1537)	MATEMATICA DISCRETA 1 (1023)
Martes 7/5	HORMIGON ESTRUCTURAL 1 (2382)	ELECTRONICA DE POTENCIA (5609)	INT. A LA CIENCIA DE MATERIALES (1723)
Miércoles 8/5	MATEMATICA DISCRETA 2 (1026)	FENOMENOS DE TRANSPORTE EN ING. DE PROCESOS (Q22)	FISICA 2 (1152) SEÑALES Y SISTEMAS (1457) METALURGIA DE TRANSFORMACION (1724)
Jueves 9/5	FISICA 1 (1151) TECNOLOGIAS DE REDES Y SERVICIOS DE TELECOMUNICACIONES (5866) TRATAMIENTO BIOLOGICO DE EFLUENTES Y RESID. SOLIDOS (Q56)	INT. A LA INGENIERIA DE SOFTWARE (1716) INSTRUMENTACION INDUSTRIAL (2706) SUBESTACIONES EN MEDIA TENSION (5514) REDES DE DATOS 2 (5855) TRANSFERENCIA DE CALOR Y MASA 1 (Q74) INGENIERIA BIOQUIMICA (Q81)	MECANICA NEWTONIANA (1122) ELEMENTOS DE MECANICA DE LOS FLUIDOS (1805) INT. A LA MECANICA DE SUELOS (2207) INT. AL CONTROL INDUSTRIAL (2707) ELECTROTECNICA (5850) TECNOLOGIA Y SERV. INDUSTRIALES 1 (Q71)

Figura B.5: Calendario revisado generado para los primeros parciales del primer semestre de 2024