UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA
VALPARAÍSO - CHILE

“UNA PROPUESTA DE SOLUCIÓN PARA LA ASIGNACIÓN DE HORARIOS DE TUTORES EN CIAC”

ANGHELO NICOLÁS CARVAJAL CARVAJAL

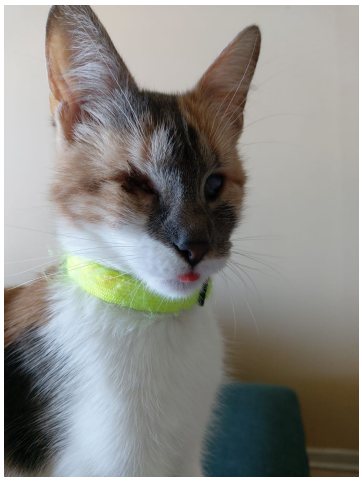
MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN INFORMÁTICA

Profesor Guía: Nicolás Rojas
Profesor Correferente: Andrea Vásquez

Octubre - 2024

DEDICATORIA

Deseo dedicarle este trabajo de memoria a mi gata Toph, la cual ha estado a mi lado diariamente entregándome el apoyo y motivación necesaria para seguir adelante.



AGRADECIMIENTOS

Quiero agradecer a mi familia por todo el apoyo que me ha entregado durante este trayecto universitario. Quiero agradecer también a mi profesor guía Nicolás Rojas y mi profesora correferente Andrea Vásquez por toda la paciencia que me han tenido en este largo y extenuante desarrollo de trabajo de memoria. También quiero agradecer al coordinador de CIAC Paulino Gallardo por su buena disposición y colaboración para la realización del presente trabajo de memoria. Finalmente quiero agradecer a mi gata Toph por entregarme la motivación necesaria a diario para poder realizar el presente trabajo.

RESUMEN

Resumen— El presente trabajo tiene como objetivo desarrollar un algoritmo que optimice la asignación de horarios para el Centro Integrado de Aprendizaje en Ciencias Básicas (CIAC) de la Universidad Técnica Federico Santa María (UTFSM). CIAC es una unidad de apoyo complementario para los estudiantes durante sus primeros años. Este algoritmo busca reducir el tiempo invertido en la planificación de horarios de los tutores y profesores colaboradores que actualmente se realiza de forma manual.

La metodología se basa en técnicas de programación lineal entera, las cuales han demostrado ser eficientes en problemas de asignación de recursos en otros contextos educativos. Esta memoria presenta un modelo matemático que permita optimizar la asignación de horarios del CIAC, considerando las restricciones y la variabilidad propias de su contexto universitario.

Utilizando el *solver* IBM ILOG CPLEX y datos de reales de CIAC como valores de entrada se valida la propuesta presentada. Los resultados muestran que la técnica propuesta permite generar múltiples asignaciones de horarios factibles en un corto periodo.

Palabras Clave— UCTTP; Modelo de programación lineal; Asignación de horarios

ABSTRACT

***Abstract*—** The objective of this work is to develop an algorithm that optimizes the time scheduling for the Centro Integrado de Aprendizaje en Ciencias Básicas (CIAC) at the Universidad Técnica Federico Santa María (UTFSM). CIAC is a complementary support unit for students during their first years. This algorithm aims to reduce the time spent on scheduling for tutors and collaborating professors, which is currently done manually.

The methodology is based on integer linear programming techniques, which have proven to be efficient in resource allocation problems in other educational contexts. This thesis presents a mathematical model that optimizes CIAC's scheduling, considering the constraints and variability inherent to the university context.

Using the IBM ILOG CPLEX solver and real CIAC data as input the proposed solution is validated. The results show that the proposed technique can generate multiple feasible schedule assignments in a short period of time.

***Keywords*—** UCTTP; Integer programming; Timetabling

GLOSARIO

CIAC: Centro Integrado de Aprendizaje en Ciencias básicas.
CSP: *Constraint Satisfaction Problem*.
MATFIS: Matemáticas y Físicas.
NP: *Nondeterministic Polynomial time*.
PACE: Programa de Acceso a la Educación Superior.
UCTTP: *University Course Timetabling Problem*.
UTFSM: Universidad Técnica Federico Santa María.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

En las instituciones educativas, la planificación eficiente de los recursos humanos y académicos es un factor crítico para asegurar el éxito de los procesos de enseñanza y aprendizaje. A medida que las universidades han evolucionado, la correcta asignación horaria de las actividades universitarias se ha vuelto más compleja, especialmente cuando se trata de asignar horarios para actividades académicas que no están ligadas a las asignaturas obligatorias de la universidad, debido a que estas actividades tienen menor libertad para poder realizar dichas asignaciones ya que las personas involucradas suelen tener sus calendarios ocupados por las actividades obligatorias de la universidad u otras actividades no académicas, los cuales pueden variar con forme va avanzando el semestre. En este contexto, el Centro Integrado de Aprendizaje en Ciencias Básicas (CIAC) de la Universidad Técnica Federico Santa María (UTFSM) se ha destacado como una unidad clave en el apoyo académico de los estudiantes de primer año, proporcionando una variedad de servicios que van desde tutorías en áreas fundamentales, como Matemáticas, Física y Química, hasta talleres psicoeducativos.

El CIAC tiene como misión fortalecer las competencias y capacidades que poseen los estudiantes de primer año de la UTFSM, dándoles apoyo integral para que tengan una inserción exitosa, lo cual realiza mediante un abanico de actividades académicas y psicoeducativas. Sin embargo, la organización de la disponibilidad tanto de sus tutores como profesores colaboradores, como la asignación de espacios físicos y virtuales para estas actividades, plantea un problema complejo de optimización. Los desafíos incluyen la necesidad de ajustar los horarios a la disponibilidad de tutores y profesores, sin sobrepasar las limitaciones impuestas por la normativa de la universidad en cuanto a horas de trabajo. Además, desde la pandemia de COVID-19, las actividades del CIAC han adoptado una modalidad híbrida, lo que ha añadido un nivel adicional de complejidad a la planificación logística.

Semestre a semestre uno de los problemas al que se enfrenta CIAC es la asignación eficiente de los horarios para los tutores, quienes, además de desempeñar su rol académico, son estudiantes con obligaciones académicas que limitan su disponibilidad horaria. Esto crea una situación de alta variabilidad y dinamismo en la que los horarios deben ser modificados semana a semana durante los inicios de cada semestre, dependiendo de las nuevas obligaciones que adquieren tanto los tutores como los profesores. En consecuencia, la asignación manual de horarios resulta ineficiente y consume muchas horas de trabajo administrativo.

La importancia de abordar este problema radica en la necesidad de optimizar los recursos académicos, reducir la carga administrativa y mejorar la calidad del apoyo brindado a los estudiantes. En este sentido, esta memoria propone la implementación de un modelo algorítmico que automatice el proceso de asignación de horarios en el CIAC, permitiendo generar soluciones factibles y adaptables a las condiciones cambiantes del semestre académico. La automatización del proceso no solo permitirá reducir el tiempo invertido en estas tareas, sino que también buscará una distribución equitativa de los turnos de trabajo.

La presente memoria tiene como objetivo diseñar e implementar un algoritmo que permita optimizar la asignación de horarios en el CIAC, considerando las restricciones y la variabilidad propias del contexto universitario. Este modelo se basará en técnicas de programación lineal entera, los cuales han sido ampliamente utilizados para resolver problemas complejos de asignación de recursos en diferentes áreas. La propuesta será validada utilizando datos reales del CIAC, con el fin de evaluar su efectividad en términos de eficiencia y reducción de tiempo en la planificación.

La estructura de este trabajo es la siguiente: El capítulo 1 consiste en la definición del problema, el cual explicará en mayor profundidad el contexto del CIAC y el problema que este trabajo de memoria busca solucionar. El capítulo 2 presenta el marco conceptual, el cual realizará un repaso a conceptos previos requeridos para la propuesta de solución del problema y una revisión a la literatura existente. El capítulo 3 presenta la solución propuesta por este trabajo de memoria. El capítulo 4 busca validar la solución propuesta, utilizando datos históricos y actuales de CIAC, corroborando que las soluciones generadas sean factibles y que esta sea adecuada para el CIAC. Finalmente el capítulo 5 presenta las conclusiones de este proyecto.

CAPÍTULO 1
DEFINICIÓN DEL PROBLEMA

Dentro de la Universidad Técnica Federico Santa María (UTFSM) existe una unidad de estudio complementario que apoya a sus estudiantes durante sus primeros años universitarios, llamado Centro Integrado de Aprendizaje en Ciencias básicas (CIAC). El CIAC es un edificio dentro de la UTFSM donde los estudiantes pueden usarlo simplemente como salas de estudio y además pueden utilizar los servicios adicionales entregados por CIAC para complementar sus estudios.

El CIAC se encarga de proveer un amplio abanico de opciones, tanto académicas como de acompañamiento psico-educativo, a los estudiantes de forma complementaria y opcional a las asignaturas obligatorias impartidas por la universidad. Algunas de estas opciones pueden variar por semestre, pero normalmente los apoyos proveídos por el CIAC son:

- Proveer espacios de estudio adicionales a los proveídos por la universidad dentro del edificio del CIAC para que sean usados como salas de estudio.
- Proveer de '**bloque de consultoría**'. Este servicio consiste en disponer de 'tutores académicos' en el edificio del CIAC. Estos tutores académicos están disponibles para resolver consultas de los estudiantes que asisten al CIAC a estudiar. Usualmente hay un tutor de Matemáticas y Físicas (MATFIS) de primer año (un tutor para ambas asignaturas), un tutor de Programación y un tutor de Química, los cuales suelen ir rotando por cada bloque académico.

  Ciertos días específicos (los cuales pueden variar por semestre) también se disponen de tutores de otras asignaturas, como 'Programación y tratamiento de datos para la gestión', 'Física 120', 'Matemática 3', entre otros.
- Horarios de '**mentorías**', en los cuales profesores de distintas asignaturas se encuentran disponibles en horarios específicos durante la semana, para que los estudiantes puedan consultar dudas de sus respectivas asignaturas. A diferencia de los bloques de consultoría realizados por tutores académicos, estos bloques tienen una disponibilidad mucho más acotada.
- Entregar '**apoyos focalizados**' a estudiantes de distintos programas de los cuales es parte la UTFSM, como el Programa de Acceso a la Educación Superior del gobierno (Programa PACE), el programa propedéutico de la universidad, el programa de inclusión de la universidad, entre otros.
- '**Apoyos intensivos**', los cuales son apoyos similares a las ayudantías dictadas por la universidad. Estos son dictados por un profesor o un tutor académico. Consisten en que le entregan una guía al estudiante, la cual está especialmente confeccionada para

los contenidos que están viendo esa semana en clases de cátedra. Estos apoyos tienen un número variable de bloques semanales, los cuales una vez están fijados no se suelen mover de hora. Usualmente se realizan entre 1 y 2 intensivos de cada asignatura por semana. Desde la pandemia del COVID-19, los apoyos intensivos cambiaron de modalidad presencial a una modalidad completamente *online*.

A diferencia de los bloques de consultoría, en donde el tutor encargado debe estar preparado para cualquier consulta relacionada a su asignatura, un apoyo intensivo está enfocado a una asignatura en particular, generando mayor variedad de apoyos intensivos aunque sean de asignaturas similares. Por ejemplo, es normal que se realice un apoyo intensivo para la asignatura **MAT060** y otro apoyo intensivo para **MATE10**, a pesar de que ambas se consideren la asignatura introductoria de Matemática de sus respectivas carreras.

Debido a la modalidad *online* de dichos apoyos, estos suelen requerir de al menos dos personas del CIAC que estén conectados a una reunión *online* para que este apoyo pueda ser realizado:

  - Por una parte se encuentra presente una persona 'académica' de dicha asignatura (ya sea al menos un profesor colaborador o al menos un tutor académico), la cual se encarga de dictar el apoyo intensivo y resolver dudas que puedan tener los estudiantes.
  - Por otra lado también se encuentra una persona 'administrativa', que se encarga de labores técnicas de la transmisión, como gestionar que se realice la correcta grabación del apoyo intensivo, hacer un seguimiento de la asistencia de los estudiantes y poder resolver problemas técnicos que puedan ocurrir.

- CIAC también cuenta con apoyo psicológico independiente del entregado por la universidad, el cual está encargado de hacer acompañamientos psico-educativos a los estudiantes de los programas de la universidad en los que apoya CIAC. Esto es gestionado a través de los apoyos personalizados, las tutorías integrales y los talleres psico-educativos.

- '**Apoyos personalizados**', en los cuales se asigna un grupo de estudiantes a un tutor académico para apoyar en los estudios de ese asignatura en particular, confeccionando guías, haciendo clases y manteniendo contacto con dicho grupo, entregando un servicio más personal y focalizado a las necesidades de dicho grupo.

- '**Tutorías integrales**', es un apoyo psico-educativo en donde a un tutor académico o un tutor administrativo se le asigna un grupo de estudiantes de primer año. A diferencia de los apoyos personalizados, el tutor se encarga de estar atento del bienestar de su grupo de estudiantes, realizar seguimiento del progreso académico de estos, poder derivar a apoyos personalizados en caso de que alguno de los estudiantes requiera apoyo en algún asignatura particular o derivar al psicólogo (a) en caso de requerir apoyos no educativos.

El equipo de trabajo del CIAC se compone principalmente de los siguientes *stake-holders*:

- **Profesores colaboradores**; los cuales también son profesores de la universidad y tienen clases que impartir de forma independientes del CIAC. Estos pueden cumplir los siguientes roles dentro de CIAC:
  - Estar disponibles dentro del edificio CIAC para responder consultas.
  - Dictar apoyos intensivos.
- **Tutores**; los cuales son estudiantes de la universidad que trabajan con rol de tutor administrativo o académico para el CIAC, pero al ser estudiantes deben cumplir con asistir a sus horarios de cátedras obligatorios y otras obligaciones universitarias. Estos pueden cumplir los siguientes roles dentro de CIAC:
  - Tutores administrativos, los cuales pueden estar a cargo de:
    - Administración del edificio, estando a cargo de la entrada y salida de estudiantes al edificio CIAC, además de abrir y cerrar el edificio CIAC día a día.
    - Entregar apoyo como anfitriones y moderadores en los intensivos online.
    - Trabajar como tutor integral.
  - Tutores académicos, los cuales pueden estar a cargo de:
    - Trabajar en los bloques de consultorías.
    - Hacer apoyos intensivos.
    - Hacer apoyos personalizados.
    - Trabajar como tutor integral.
  - Además, estas personas pueden cumplir algunos roles particulares que pueden ir surgiendo con el tiempo, como son el rol del administrador general, el de soporte técnico online, entre otros.
- **Psicólogo(a)**, encargado de la organización de actividades psico-educativas y apoyo sicológico a estudiantes que lo requieran.

Dado todo este amplio conjunto de apoyos entregado por el gran equipo de trabajo de CIAC, en donde cada uno puede cumplir uno o más roles, se presenta un importante problema semestre a semestre, el cual es la asignación de horarios para cada uno de los colaboradores CIAC.

Una restricción importante a considerar al momento de asignar actividades a los tutores es que la universidad impone un límite a la cantidad de horas que estos pueden trabajar en distintas ayudantías y laboratorios al mes. Un estudiante de la UTFSM puede trabajar 15 horas al mes en actividades remuneradas de la universidad, con la opción de poder solicitar una extensión horaria de hasta 30 horas al mes (la cual puede o no ser aprobada). Un estudiante

es libre de realizar más de una actividad remunerada a la vez dentro de la universidad, de modo que el CIAC debe saber cuantas horas requieren las otras posibles ayudantías que el estudiante también realice, para así saber de cuantas horas dispone al momento de asignarle alguna actividad y no sobrepasar el límite de horas.

De las actividades anteriormente descritas, solo 2 de ellas tienen bloques semanales fijos asignados por la administración de CIAC, los 'bloque de consultoría' y los 'apoyos intensivos'. Además, es necesario asignar bloques a los tutores administrativos que se encargan de la administración presencial del edificio. Antes de poder decidir que personas se les asignarán que bloques de trabajo primero se debe decidir que bloques de trabajo existirá para cada tipo de trabajo.

- **Bloques de tutores administrativos**. Decidir los bloques horarios que se realizarán para tareas administrativas es sencillo. Siempre es necesario que haya 1 tutor administrativo en el edificio CIAC mientras este esté abierto, por lo que los bloques correspondientes a esta actividad serían igual al rango horario en que el CIAC se encuentre abierto.

  En la figura 1 se muestra un ejemplo de este tipo de asignación, siendo estos los nombres en negro. Los nombres en rojo que se muestran en esta figura corresponden a tutores administrativos que fueron asignados a colaborar en apoyos intensivos.

Horario Administradores 1S-2023 (Planificación SEMESTRAL).-					ONLINE
BLOQUES/HRS	LUNES	MARTES	MIÉRCOLES	JUEVES	VIERNES
(3-4) 9:30-10:45	JAVIERA R.	LEONARDO G.	VALENTINA F.	FERNANDO V.	MAXIMILIANO O.
(5-6) 10:50-12:05	JAVIERA R.	LEONARDO G.	ANAIS F.	DANIELA C.	JAVIERA M.
(7-8) 12:10-13:25	MARIA JESUS	FELIPE B.	VALENTINA F.	MARIA JESUS	ANAIS F.
HORA ALMUERZO	MAYRA Z.	ALLAN V.	MAYRA Z.	ALLAN V.	MAYRA Z.
(9-10) 14:30-15:45	MARIA JESUS	BLAS S.	JAVIERA C.	BLAS S.	MAXIMILIANO O.
(11-12) 15:45-17:00	JAVIERA C.	FELIPE B.	MARIA JOSE A.	JAVIERA M.	ALLAN V.
(13-14) 17:05-18:20	MAXIMILIANO O MARTINA L.(Mat021) ARANTZA T.(Mate10)	LEONARDO G. VANESSA R.(Mat023)	DANIELA C. CATA O.(Mat60-70)	MARTINA L. CATA O.(Mat60-70)	DANIELA C. MARTINA L.(Mat021) MARIA JOSE A.(Mat024)
(15-16) 18:25-19:40	CINTHYA A. CASSIDY A.(Qui010) ARANTZA T.(Mate10) VANESSA R.(Mat023)	HECTOR M. CASSIDY A.(Qui010) SCARLETTE B.(Fis120)	JAVIERA R. JAVIERA C.(Mat022)	ARANTZA T. ANAIS S.(Fis100)	NATALIA E. SCARLETTE B.(Fis110)
(17-18) 19:45-21:00	CINTHYA A.	MARIA JOSE A. CAMILA C.(Iwi131)	FERNANDO V. CAMILA C.(Iwi131)	CAMILA C. ANAIS S.(Fis100) NATALIA E.(Inf130)	NATALIA E.

Figura 1: Ejemplo de asignación horaria para tutores administrativos.
Fuente: Horarios CIAC 2023, primer semestre.

- **Bloques de consultoría**. La creación de bloques de este tipo de actividad varía según el asignatura en cuestión, debido a que cada asignatura tiene una demanda distinta por los estudiantes que asisten al CIAC. Por ejemplo, hay mucha mayor demanda de tutores de MATFIS comparado con la demanda existente por tutores de Química o Programación (por esto mismo también se suele contratar una menor cantidad de tutores de estos últimas asignaturas).

  Debido a la alta demanda de tutores de MATFIS, se suele asignar bloques de consultoría de MATFIS para todos los días, desde el bloque anterior al bloque de almuerzo hasta el horario de cierre del edificio CIAC. La figura 2 muestra un ejemplo de asignación de tutores de MATFIS (letras en negro), tutores de la asignatura MAT023 (letras en verde) y tutores de la asignatura FIS120 (letras en celeste).

Horario CONSULTORÍAS MAT-FÍS / 1S-2023 (Planificación SEMESTRAL).-					
BLOQUES/HRS	LUNES	MARTES	MIÉRCOLES	JUEVES	VIERNES
(7-8) 12:15-13:25	NICOLAS J.	ISAIAS O.	DIEGO U.	GONZALO R.	LUCAS C.
(9-10) 14:30-15:40	NICOLAS J.	ISAIAS O.	JORGE N.	GONZALO R.	LUCAS C.
			JOAQUIN V.(Fis120)		
(11-12) 15:50-17:00	DIEGO U.	CLEMENTE M.	VINCENZO R.	LUCIANO D.	MATIAS F.
			JOAQUIN V.(Fis120)		
(13-14) 17:10-18:20	DIEGO U.	CLEMENTE M.	VINCENZO R.	LUCIANO D.	MATIAS F.
		NICOLAS B.(Mat023)	JOAQUIN V.(Fis120)		
(15-16) 18:30-19:40	MANUEL T.	FELIPE A.	RODRIGO B.	BENJAMIN G.	RODRIGO B.
		NICOLAS B.(Mat023)			
(17-18) 19:50-21:00	MANUEL T.	FELIPE A.	RODRIGO B.	BENJAMIN G.	RODRIGO B.
		NICOLAS B.(Mat023)			

Figura 2: Ejemplo de asignación horaria para bloques de consultoría de MATFIS.
Fuente: Horarios CIAC 2023, primer semestre.

En el caso de las consultorías que tienen menor demanda, como las asignaturas de Química y de Programación, se suele crear bloques al día correspondiente al que se suele realizar el certamen del asignatura en cuestión y al día anterior a ese. En la UTFSM las asignaturas suelen realizar 3 grandes evaluaciones escritas, llamadas ‘certamen’. El día que corresponde a los certámenes de ese asignatura se suelen asignar bloques de consultoría solo medio día, mientras que se asigna la mayor parte de los bloques del día para el día anterior al certamen. Por ejemplo, si Química realiza sus certámenes los días miércoles, entonces se crearán bloques de consultoría para la mayor parte del día Martes y la mitad del día Miércoles. La figura 3 muestra un ejemplo de asignación para tutores de QUI010. La “X” en grande de la figura especifica que para este semestre se decidió que no se realizarán consultorías académicas de QUI010 en los días Jueves ni Viernes.

Horario CONSULTORÍAS QUI010 / 1S-2023 (Planificación SEMESTRAL).-					
BLOQUES/HRS	LUNES	MARTES	MIÉRCOLES	JUEVES	VIERNES
(9-10)	JOSEFA I.	SEBASTIAN J.			
14:30-15:40					
(11-12)	JOSEFA I.	FRANCISCO M.			
15:50-17:10					
(13-14)	AKSHAY M.	FRANCISCO M.			
17:10-18:20					
(15-16)	CARLOS B.	CARLOS B.			
18:30-19:40					

Figura 3: Ejemplo de asignación horaria para bloques de consultoría de QUI010.
Fuente: Horarios CIAC 2023, primer semestre.

- **Apoyos intensivos**. De forma similar que la creación de bloques de consultoría de baja demanda, se suele dar prioridad a que los apoyos intensivos se asignen los días anteriores a la realización del certamen, sin incluir el día del certamen mismo. Debido a que estos apoyos se realizan en menor cantidad semanalmente, hay una menor cantidad de personas académicas que se les confíe en dictar estos apoyos y que estos apoyos requieren también un tutor administrativo a cargo. Estos apoyos se suelen crear en base a la disponibilidad de los actores involucrados en vez de crear los bloques horarios deseados y luego intentar calzar a los tutores en dichos bloques horarios. La figura 4 muestra un ejemplo de este tipo de asignación horaria.

CIAC CENTRO INTEGRADO DE APRENDIZAJE EN CIENCIAS BÁSICAS

PLANIFICACIÓN APOYOS INTENSIVOS 1er SEMESTRE 2023.-(versión 2.0)					
FECHA	LUNES	MARTES	MIERCOLES	JUEVES	VIERNES
BLOQUES	TITULAR	TITULAR	TITULAR	TITULAR	TITULAR
7-8					
12:15 a 13:25					
	MAT 021	MAT 023	MAT 060-070	MAT 060-070	MAT 021
	Prof.Antonio Morales	Prof.Antonio Morales	Prof.Alejandro Vidal	Prof.Alejandro Vidal	Prof.Antonio Morales
13-14					
17:10 a 18:20	MATE 10				MAT024
	Prof.Simón Masnu				Prof.Simón Masnu
	QUI 010	QUI 010	MAT 022	FIS 100	FIS110
	Francisco M./	Francisco M./	Prof.Alejandro Vidal	Prof.Daniel Erraz	Prof.Daniel Erraz
15-16	MAT 023	FIS 120			
18:30 a 19:40	Prof.Antonio Morales	Prof.Daniel Erraz			
	MATE 10				
	Prof.Simón Masnu				
		IWI 131	IWI 131	FIS 100	
		Anghelo C./	Anghelo C./	Prof.Daniel Erraz	
17-18				INF-130	
19:50 a 21:00				Anette R./	

Figura 4: Ejemplo de asignación horaria para bloques de apoyos intensivos.
Fuente: Horarios CIAC 2023, primer semestre.

Debido a que el CIAC se encarga de proveer apoyos complementarios a las asignaturas obligatorias dictadas por la universidad, este debe adaptarse a los horarios de los estudiantes a

los que les ofrece sus servicios como a los horarios de los tutores (que a su vez también son estudiantes de la universidad) y los profesores colaboradores.

Idealmente el CIAC asignaría horarios fijos a sus colaboradores antes de que empiece el semestre. Pero esto no es posible debido a que las actividades y horarios de los colaboradores están sujetos a cambios durante el inicio del semestre, como por ejemplo los tutores pueden cambiar sus asignaturas durante las primeras semanas del semestre, que se le asignen nuevas ayudantías, laboratorios, talleres, actividades extra-académicas, entre otros. Lo mismo ocurre con los profesores colaboradores, los cuales también trabajan haciendo clases de cátedra en la universidad y no tienen control sobre los horarios que le asignan los departamentos en los que trabajan.

Por esto, el CIAC debe poder adaptarse y generar nuevas asignaciones de horarios a sus colaboradores durante el semestre académico, teniendo muchas veces que generar horarios completamente nuevos. Si el CIAC no fuera lo suficientemente flexible para poder adaptarse a estos cambios sobre la marcha entonces no sería capaz de poder ofrecer sus servicios.

El CIAC le permite a sus colaboradores actualizar su disponibilidad horaria y la administración del CIAC se encarga semana a semana de corroborar que los bloques asignados a estos colaboradores se adapten correctamente a su disponibilidad horaria. En caso de que un colaborador ya no tenga disponibilidad en un bloque que tenía previamente asignado entonces se procede a reasignar los horarios de los colaboradores. Esta comprobación y posterior asignación de horarios se realiza de manera semanal y de forma completamente manual por la administración del CIAC.

Esto produce que se gasten muchas horas persona en asignaciones de horarios que satisfagan las restricciones y agenda de todos los involucrados. Considerando información más concreta, la persona encargada de realizar y actualizar los horarios de CIAC demora 2 días laborales (Jueves y Viernes) en generar una nueva asignación factible para los tutores y profesores colaboradores, lo cual se suele repetir durante aproximadamente las primeras 6 semanas del semestre. Esta memoria busca automatizar el proceso de asignación horaria y aprovechar mejor el tiempo de los encargados en realizarla.

Debido a este problema, este trabajo de memoria tiene como **objetivo general** diseñar e implementar un algoritmo que busque minimizar el tiempo utilizado en la asignación de horarios semestral de los colaboradores del CIAC, lo cual buscará realizar a partir de los siguientes pasos

- Analizar modelos matemáticos existentes que abarquen problemas similares al problema de asignación de horarios de CIAC.
- Presentar un modelo matemático que pueda representar el problema de asignación de horarios de CIAC.
- Evaluar la técnica generando calendarizaciones y horarios usando datos reales de CIAC.

CAPÍTULO 2
MARCO CONCEPTUAL

Este capítulo explora los conceptos claves que sustentan el algoritmo que se propondrá en el siguiente capítulo para realizar asignaciones horarias para el CIAC. Veremos conceptos previos y métodos que son utilizados para solucionar problemas matemáticos de optimización. Además se presenta una revisión de la literatura existente que aborda problemas de asignación de horarios universitarios, debido a su similitud con el problema de este trabajo de memoria.

El problema de asignación de horarios es un problema combinatorio muy complejo, el cual se encuentra en la mayoría de instituciones educativas. El problema base suele consistir en asignar un número de profesores a un número de cursos de modo que un profesor no puede tener más de un curso al mismo tiempo.

Los problemas de asignación de horarios en instituciones educativas han sido clasificado en 3 grandes categorías: *school timetabling*, *course timetabling* y *exam timetabling* [Stichting *et al.*, 1996].

Los problemas de asignación de horarios en general tienen muchas variantes como el *Educational timetabling*, *Nurse Rostering*, *Sports timetabling*, *Employee timetabling* y el *Transport timetabling* [Song *et al.*, 2018], todos estos siendo parte de la familia de problemas NP-completo [Bakir y Aksop, 2008].

El concepto de los problemas NP-completos fue introducido por primera vez por Richard Karp en “Reducibility Among Combinatorial Problems” [Karp, 1972]. Un problema NP-completo consiste en un problema donde la factibilidad de una solución puede ser verificada en tiempo polinomial (de forma relativamente rápida), pero que intentar generar una solución factible y que optimice la función objetivo, toma un tiempo exponencial.

En este caso este trabajo se enfocará en el problema basándonos en el *University Course Timetabling Problem* (UCTTP), el cual ha sido altamente estudiado a lo largo de los años debido a su alto rango de aplicaciones [Mirhassani, 2006]. Este es una de las aplicaciones más cercanas al problema de asignación de horarios para CIAC, pero que al mismo tiempo no contempla todas las variaciones que requiere CIAC.

2.1. Conceptos previos

En esta sección revisaremos los conceptos que se utilizan dentro del UCTTP y las técnicas usualmente utilizadas para atacar este problema. Papers como el de [Feizi Derakhshi *et al.*, 2012] trabajan en mucha más profundidad el como estas técnicas han

sido aplicadas para resolver variantes del UCTTP.

2.1.1. *Constraint satisfaction problem*

Los problemas de satisfacción de restricciones (CSP por su sigla en Inglés) son problemas matemáticos que modelan problemas del mundo real. Estos consisten en 3 partes, un conjunto de la variables del problema, un conjunto de dominios no vacíos para estas variables, y un conjunto de restricciones. Para resolver un CSP se debe encontrar un conjunto de valores para cada una de las variables del sistema y poder satisfacer las restricciones predefinidas. Este método suele ser modelado a través de modelos de programación lineal entera, las cuales aplican matemática discreta. Casos como el de [Eledum, 2017], el cual ya se ha discutido en este escrito, usa una variante de este método para poder resolver este problema.

2.1.2. Coloreo de grafos

Este problema fue resuelto por primera vez en la literatura en 1967 usando el problema de coloreo de grafos [Welsh y Powell, 1967]. Sin embargo el método presentado no podía resolver instancias de este problema que tuviera sesiones pre-asignadas. En 1985, De Werra describió el UCTTP como un modelo de coloreo de grafos no direccionales [de Werra, 1985], en donde se apunta a colorear el grafo usando un número dado de colores donde ningún nodo adyacente conectado con una restricción puede tener el mismo color y se debe colorear con la menor cantidad de colores posible. Este método considera a los eventos (profesores, estudiantes y salas de clases) como nodos, restricciones como arcos del grafo y bloques de clases como colores, esto se puede apreciar en la figura 5. Nodos coloreados son equivalentes al número de bloques, de modo que habrá un conflicto si dos bloques adyacentes tienen el mismo color.

Eventualmente otros autores han iterado en este método para solucionar el UCTTP, pero todas estas soluciones tienen el mismo problema que tiene el problema de coloreo de grafos base, el cual es que encontrar el número requerido de colores para poder colorear el grafo tiene tiempo NP duro, de modo que encontrar soluciones para este problema puede tomar demasiado tiempo.

2.1.3. Métodos meta-heurísticos

Este tipo de métodos buscan solucionar problemas usando heurísticas generales en vez de técnicas completas, debido a que estas últimas tienen tiempos de resolución muy alto y este tipo de métodos pueden entregar una solución mucho más rápido. Hay principalmente dos enfoques meta-heurísticos en la literatura que abordan este problema, el enfoque basado

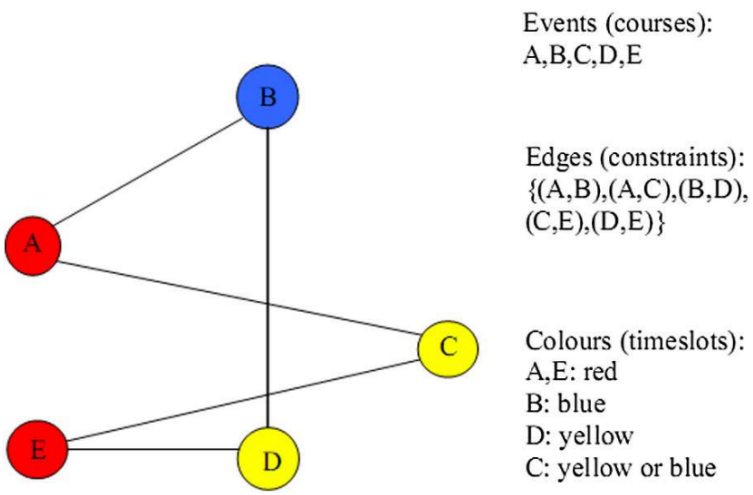


Figura 5: El problema de asignación de horarios representado en el método de coloreo de grafos.

Fuente: [de Werra, 1985]

en población y el enfoque de solución única o trayectorial.

Los enfoques basados en población empiezan con un conjunto inicial de soluciones, llamada la población inicial, para luego mejorar estas soluciones de forma iterativa. Usualmente, cada nuevo conjunto de soluciones reemplaza a la población actual, repitiendo el proceso hasta que se alcance una solución deseada usando una función de verificación o se utilicen los recursos definidos para resolver el problema. Algunos ejemplos que se han utilizado para resolver el UCTTP usando este enfoque incluyen:

- Algoritmos evolutivos y genéticos. Estos simulan la evolución natural, incluyendo selección, cruce y mutaciones para mejorar la población. Khonggamnerd y Innet [Khonggamnerd e Innet, 2009] usaron un algoritmo genético para generar la asignación de horario universitaria, resultando en una mayor utilización de las salas de clases y sus respectivas capacidades.

- Optimización de colonia de hormigas. Inspirado por el comportamiento de las hormigas, donde estas siempre están intentando buscar comida y dejan un camino de feromonas que pueden usar otras hormigas, el cual eventualmente se evapora de modo que dicho camino se vuelve menos atractivo para las siguientes hormigas. El objetivo de este enfoque es que el movimiento de las hormigas artificiales encuentren la ruta más corta. Este tipo de sistema puede ser utilizado para resolver el UCTTP al buscar construir un camino óptimo de modo que cada ruta genere un grafo que permita asignar los cursos a los bloques de horarios afectados por la cantidad de feromonas en un rango [Socha *et al.*, 2002]. Otro método de resolver este problema es en el cual cada hormiga asigna los eventos a las salas y los bloques de horarios usando dos tipos de

feromonas [Nothegger *et al.*, 2012]. Este último algoritmo tiene buen rendimiento y puede entregar buenas soluciones en un plazo de ejecución acotado.

- Algoritmo memético. Este tipo de algoritmo se basa en el algoritmo genético y *hill climbing*. Un *meme* es considerado como una unidad de información generada por las personas. El *meme* se diferencia de un gen en que el meme se transmite entre las personas y cada una de estas puede adaptarlo para mejorarlo según su criterio, mientras que el gen se transmite sin cambios. Jat y Shengxiang usaron un algoritmo memético para resolver el UCTTP usando una combinación de búsqueda local y algoritmos genéticos [Jat, 2008], una de las búsquedas locales se realizaba para el conjunto de profesores, estudiantes y salas de clases, mientras que la otra se realizaban para los bloques de horarios.

Por otro lado, los enfoques trayectoriales son métodos que usan una única solución y buscan mejorarla iterativamente, enfocándose en optimizar algún criterio dado. Algunas ejemplos que se han utilizado para resolver el UCTTP usando este enfoque incluyen:

- *Hill climbing*. Busca una solución apuntando a optimizar un criterio, moviéndose en un espacio de soluciones el cual es generado al modificar la solución de la iteración actual. La solución elegida en cada iteración siempre es la de mejor calidad en la vecindad actual, de modo que este enfoque entrega óptimos locales. Este algoritmo termina al encontrar un óptimo local, alcanzar una cantidad de iteraciones entregada como parámetro o una vez que pase cierta cantidad de tiempo dado.

- Búsqueda tabú. Este método consiste en tomar una solución inicial y crear una vecindad de posibles soluciones generadas en base a pequeños cambios de la solución actual. Este algoritmo cambia a la mejor solución de la vecindad actual, siempre y cuando esta solución no se encontrara en la lista tabú. Al realizar este movimiento, se actualiza la lista tabú de modo que se agrega la solución anterior para evitar volver a ella en el futuro y entrar a un ciclo. Este método fue aplicado por primera vez para solucionar el problema de asignación de horarios en 2002, entregando buenos resultados [Alvarez-Valdes *et al.*, 2002].

- *Simulated Annealing*. Este método es una búsqueda local inspirada en el calentamiento de sólidos en la física. Este enfoque evita quedarse atrapado en soluciones óptimas locales. Las soluciones obtenidas por la búsqueda local reemplazan la solución actual, iterando el proceso hasta que hasta que se cumpla el criterio para dejar de iterar. Este método se diferencia de una búsqueda local ordinaria en que se tiene en consideración una “temperatura inicial”, la cual decrece en cada iteración en base a parámetros iniciales. A temperaturas mayores es más probable que el sistema salte a soluciones de más baja calidad que podrían no ser la óptima en cada iteración. Este proceso permite que el sistema se estanque en óptimos locales, mejorando la exploración de soluciones. Una de las formas de resolver el UCTTP usando este método es el descrito en

[Tuga *et al.*, 2007], donde proponen un híbrido de este método con el *Kempe Chain Neighborhood*. Ellos proponen reformular las restricciones duras como restricciones blandas relajadas, para luego utilizar el método *Simulated Annealing* para minimizar las violaciones de restricciones blandas.

2.2. Revisión literatura

El presente trabajo de memoria se enfoca principalmente en la asignación de horarios de distintas actividades universitarias, el cual debe apegarse a las necesidades de CIAC. En la literatura, lo más cercano a dicho problema es el *University Course Timetabling Problem* (UCTTP). El UCTTP es un problema que se encuentra presente en la mayoría de instituciones académicas, pero la primera vez que este problema ha sido planteado en la literatura fue en el año 1963 [GOTLIEB, 1963]. La versión más básica de este problema trata acerca de la asignación de un número dado de profesores a una cantidad de cursos de modo que un profesor no puede tener dos cursos al mismo tiempo. Sin embargo, existen variantes como la asignación de horarios de colegios, asignación de horarios para evaluaciones, asignación de horarios universitarios para múltiples periodos académicos, problemas que tienen (o no tienen) en consideración la disponibilidad de los estudiantes o sus posibles topes horarios, entre otros [Eledum, 2017].

El problema de la asignación de horarios es un problema muy difícil, perteneciendo a la familia de problemas NP-completo [Bakir y Aksop, 2008], siendo este un problema que no puede ser resuelto en tiempo polinomial.

Se han utilizado muchas técnicas para resolver variantes del UCTTP, usando tanto técnicas completas como incompletas. A continuación se explican algunos casos del mundo real y como fueron resueltos.

2.3. *The University Timetabling Problem: Modeling and Solution Using Binary Integer Programming with Penalty Functions* [Eledum, 2017]

Este estudio se enfoca en el problema de asignación de horarios para cursos del Departamento de Matemáticas en la Universidad de Taibah, en Saudi Arabia. Este departamento ofrece un programa de bachillerato que contiene cursos obligatorios y electivos, incluyendo actividades prácticas. Cada estudiante normalmente inscribe 18 horas de clases y actividades a la semana, pero tienen la posibilidad de tomar hasta un máximo de 21 horas. Se espera que los estudiantes se gradúen en 8 semestres. Los cursos obligatorios pueden ser tomados una vez que su prerrequisito sea cumplido. Los cursos electivos se pueden tomar desde el quinto semestre.

El concepto básico de este estudio es el de construir una malla académica, pero este va un

paso más allá, debido a que a diferencia de las mallas académicas de la UTFSM donde la asignación de bloques y salas se modifica cada semestre, este busca asignar una sala a cada curso y sus bloques semanales al momento de la creación de la malla. Para construir una asignación de horarios adecuada para este departamento de la Universidad de Taibah se deben cumplir varios criterios:

- Evitar que un estudiante, profesor o sala de clases se le asigne más de una actividad en un momento dado.
- Cada curso tiene 2 clases semanales, en las cuales siempre debe haber un día de diferencia entre ambas clases.
- Cada día consisten en 8 bloques fijos, donde cada uno dura 1.5 horas.
- Todas las clases de un curso dado deben ser dictadas en la misma sala de clases.
- Si un curso consiste de clases de teoría y de práctica, entonces la clase práctica debe ser realizada después de la clase teórica, de modo que ambas deben ser realizadas en la misma semana.
- La carga académica de un estudiante regular no puede sobrepasar las 18 horas semanales.
- Se debe respetar los prerrequisitos de los cursos.

Los autores proponen un modelo de programación entera binaría linear para poder solucionar este problema, la cual tiene como función objetivo la insatisfacción de las personas involucradas. Sin embargo, debido a la alta complejidad del modelo (el cual una instancia debía trabajar con varios millones de variables) decidieron reducirlo y descomponerlo en 2 sub-modelos, un modelo reducido específicamente para los estudiantes y otro para los profesores.

Además, los autores aplicaron una función de penalización exterior para transformar el problema discreto original en un problema continuo sin restricciones. Para poder obtener la mejor solución posible, los autores propusieron 2 funciones de penalización: Penalización de varianza, la cual busca relacionar las restricciones y el promedio de cada variable al aplicarle un peso a cada una; y penalización pseudo-convexa, la cual relaciona la función objetivo al cuadrado y las restricciones del problema.

Debido a que los autores transformaron su modelo a uno continuo los autores también diseñaron un algoritmo para poder producir una aproximación entera de la solución continua entregada por su modelo modificado. Al analizar los resultados numéricos, los autores remarcan que todas las funciones de penalización entregaban buenos resultados, pero que la función de penalización de varianza era levemente mejor.

2.4. *A memetic algorithm for University Course Timetabling* [Jat, 2008]

Este estudio busca solucionar el UCTTP usando una meta-heurística, en particular usa un Algoritmo Memético para poder resolverlo.

En este estudio buscan resolver un UCTTP clásico, donde tienen que asignar tanto clases y personas a una cantidad fija de bloques de clases, teniendo en consideración la asignación de salas de clases, y un conjunto de características que cada sala de clases posee y que son requeridas por cada clase. En este estudio en particular hay una pequeña cantidad de restricciones duras y blandas. El autor lista las siguientes restricciones duras:

- Ningún estudiante puede asistir a más de una clase al mismo tiempo.
- La sala es lo suficientemente grande para que quepan todos los estudiantes que asistan a dicha cátedra.
- La sala debe satisfacer los requerimientos pedidos por la clase.
- Solo una clase puede ser dictada en una sala al mismo tiempo.

El autor también lista las siguientes restricciones blandas, las cuales penalizan a la solución entregada cada vez que una de estas es violada.

- Un estudiante no debería tener una clase en el último bloque del día.
- Un estudiante no debería tener más de dos clases una tras otra.
- Un estudiante tiene una única clase en un día.

El algoritmo que proponen toma una instancia del problema, generando un conjunto de soluciones inicial, en donde a cada una le aplica dos tipos de búsquedas locales (LS1). Posteriormente, en cada iteración se selecciona una solución del conjunto y se le aplican posibles cruzamientos, posibles mutaciones y los dos algoritmos de búsqueda local antes mencionados, esta solución resultante reemplaza a la peor solución del conjunto de soluciones actual.

El algoritmo de búsqueda local LS1 se encarga de verificar y arreglar violaciones a restricciones duras, ignorando las restricciones blandas. LS1 aplica movimientos a la solución como mover una clase asignada de un bloque a otro libre, intercambiar los bloques asignados de de dos clases e intercambiar el bloque asignado de tres clases. Estos movimientos los realiza de forma iterativa hasta que se logre una solución mejor que la original o que se haya alcanzado una cantidad dada de iteraciones. Si al acabar ya no quedan violaciones a las restricciones duras entonces LS1 repite el proceso con las restricciones blandas, en caso contrario este algoritmo termina.

El algoritmo de búsqueda local toma el bloque horario que tenga un mayor número de violaciones de restricciones, luego toma una de las clases asignadas a este bloque y lo cambia a un bloque libre. De esta forma, LS2 suele lograr mejorar la cantidad de restricciones violadas.

Debido a la mezcla del algoritmo memético y las búsquedas locales que el autor propone, los resultados experimentales produjeron soluciones de mejor calidad, siendo capaces de encontrar soluciones muy cercanas al óptimo para los problemas de prueba.

2.5. *A Utilization-based Genetic Algorithm for Solving the University Timetabling Problem (UGA)* [Abdelhalim y El Khayat, 2016]

Este *paper* explica como los autores resuelve el problema de asignación de horarios ajustado específicamente para las necesidades de la Facultad de Comercio de la Universidad Alexandria en Egipto, usando un algoritmo genético.

Para esta versión del UCTTP se tienen en consideración las siguientes restricciones duras y blandas:

Restricciones duras:

- A los profesores se les asignan bloques horarios de su preferencia de acuerdo a cierta medida de prioridad.
- La cantidad de estudiantes es compatible con la capacidad de las salas de clase.
- Los profesores no son asignados a diferentes salas de clase en el mismo bloque horario.
- Grupos de estudiantes no son asignados a salas de clases diferentes en el mismo bloque horario.
- Los Viernes y los bloques horarios del final del día no están permitidos (La semana empieza el Sábado y termina el Miércoles).
- Una tabla de horarios debe incluir todas las clases que se dictan.

Restricciones blandas:

- Un profesor no debe dictar más de dos cátedras de un mismo asignatura al día.
- Un grupo de estudiantes no se le debe asignar más dos cátedras de un mismo asignatura al día.

- Un grupo de estudiantes no debe tener un espacio libre de cátedras de un mismo asignatura si estas son el mismo día.
- Un profesor no puede tener espacios libres entre cátedras de un mismo asignatura el mismo día.
- El porcentaje de utilización de una no debe ser menor al 75 % o debe ser igual al 0 %.

Los autores de este *paper* presentan un algoritmo genético para resolver este problema.

Chromosome 1

Room \ Slot	11	12	13	14
201	281 (83%)	3 (85%)	**4** (33%)	
302	22 (75%)	5 (85%)		
403	**300** (55%)		50 (100%)	52 (79%)
503		284 (77%)	10 (87%)	
507	282 (91%)		**40** (60%)	45 (80%)

Chromosome 2

Room \ Slot	11	12	13	14
201	**40** (36%)	281 (83%)	50 (100%)	282 (80%)
302	22 (75%)			**4** (53%)
403				**300** (55%)
503	52 (97%)	284 (77%)		10 (87%)
507		5 (93%)	45 (80%)	3 (93%)

Offspring Created

Room \ Slot	11	12	13	14
201		281 (83%)	50 (100%)	282 (80%)
302	22 (75%)		**40** (85%)	
403				
503	52 (97%)	284 (77%)	**4** (66%)	10 (87%)
507	**300** (77%)	5 (93%)	45 (80%)	3 (93%)

Figura 6: Ejemplos de cromosomas para poder ilustrar la lógica del cruzamiento
Fuente: [Abdelhalim y El Khayat, 2016]

Este trabajo utiliza una representación de un cromosoma bidimensional. La primera columna indica cada sala de clases mientras que la primera fila indica el identificador de cada bloque horario. Los identificadores de bloques horarios están compuestos por 2 dígitos, el primero indica el día de la semana (por ejemplo el día sábado teniendo el valor “1”) mientras que el segundo dígito indica el bloque de horario dentro de ese día. Cada casilla de esta tabla indica posibles combinaciones de sala de clase y bloque horario posible para una cátedra. Cada una de estas casillas puede tener cero o más identificadores de eventos (entiéndase evento como una actividad a realizar) el cual empaqueta tanto al identificador del profesor,

el identificador del grupo de alumnos y el identificador de la asignatura. Este modelo permite que hayan múltiples combinaciones de eventos en la misma combinación de bloque y sala debido a que estos pueden representar al mismo profesor y asignatura pero distinto grupo de alumnos.

Los algoritmos genéticos requieren una población inicial, es decir un conjunto soluciones iniciales (sean factibles o infactibles), para poder mejorarla de forma iterativa. Una forma común para la generación de la población inicial es generarla de forma completamente aleatoria, lo cual suele generar soluciones infactibles como punto de inicio. Los autores decidieron evitar eso y generar soluciones factibles que no violen ninguna restricción dura desde un inicio usando un conjunto de heurísticas, debido a que esto disminuye el tiempo de computación requerido para convertir soluciones infactibles en soluciones factibles y dirigir la búsqueda de soluciones hacía una mejor convergencia para mejores soluciones. La primera heurística, llamada *Largest degree first* (LD), consiste en ordenar la disponibilidad horaria de los profesores, empezando con aquellos que tengan menor disponibilidad y teniendo en consideración los eventos ordenados en forma decreciente (según el número de conflictos que tienen con los demás eventos). Esta heurística toma en consideración pesos para cada tipo de profesor, donde a mayor peso mayor prioridad tendrá para que este sea asignado. La prioridad de estos, listada de mayor prioridad a menor prioridad, es A) los profesores *part-time*, B) según su rango académico, C) la cantidad de grupos a la que dicho profesor tiene que enseñar y D) si son adultos mayores. La segunda heurística *Largest enrollment first* (LE) tiene en consideración la cantidad de grupos que un profesor tiene asignado, dándole prioridad a los profesores con un mayor número de grupos a su cargo.

Una función *fitness* es aquella que se encarga de medir que tan buena es una solución en cada iteración. Estas varían por cada algoritmo de modo que se ajuste mejor a las necesidades del problema. Los autores presentan una función *fitness* que consiste en la suma de las restricciones blandas violadas, donde cada una posee un peso distinto.

Los autores proponen un operador de cruzamiento de genes que llaman “cruzamiento por utilización”, el cual se enfoca en las tasas de utilización de las salas de clases, con lo cual se busca disminuir la cantidad de salas sobre utilizadas y sub utilizadas. Se toma de forma aleatoria los eventos sub y sobre utilizados de un cromosoma, y estos se utilizan para reemplazar las salas de otro cromosoma (el cual debe estar en el mismo intervalo de tiempo).

Por otro lado los autores también proponen un operador de mutación el cual llaman “mutación dirigida”, el cual se enfoca en mutar solo algunas partes del cromosoma a diferencia de mutaciones más estándar que buscan permutar el cromosoma entero. Se utiliza una búsqueda descendente simple como búsqueda local para mejorar el cromosoma. Se obtiene una lista de eventos subutilizados y se eligen aleatoriamente eventos de la lista para examinar sus vecindarios y mejorar la utilización. Esto implica mover eventos a salas diferentes dentro del mismo intervalo de tiempo para respetar las preferencias de los profesores y mejorar la eficiencia. La mutación exitosa ocurre cuando no se violan restricciones y la nueva solución es mejor en términos de aptitud.

Al probar el algoritmo propuesto por los autores, este tenía un tiempo de ejecución rápido para la instancia de mayor tamaño, mientras que este tomaba un tiempo un poco mayor al ejecutarse en la instancia de tamaño medio. El rendimiento general de este algoritmo es prueba de ser una herramienta efectiva para generar asignaciones de horarios para grandes universidades.

2.6. Resumen

La revisión de la literatura demuestra que el problema de asignación de horarios, en sus diferentes variantes, ha sido abordado con una variedad de técnicas. Desde enfoques basados en programación entera hasta métodos metaheurísticos como algoritmos genéticos y meméticos han sido utilizados para esta tarea.

Los estudios revisados presentan soluciones aplicables a diferentes contextos académicos, teniendo estos en común con CIAC la necesidad de asignar recursos limitados como profesores, tutores, salas de clases y comparten la necesidad de encontrar una forma eficiente de distribuir estos recursos sin solapamientos.

Al mismo tiempo, las necesidades de CIAC se diferencian en varios puntos a los casos típicos de asignación horaria universitaria. Mientras que los estudios revisados se centran principalmente en la asignación de horarios para clases obligatorias y electivas en un contexto académico tradicional, el caso de CIAC es más específico y está relacionado con la asignación de actividades complementarias (consultorías, tutorías, apoyos intensivos) que no son parte del plan formal de estudios, por lo cual CIAC no realiza asignaciones horarias para los estudiantes individuales a los que están dirigidas sus actividades. En los estudios revisados, los horarios tienden a ser más rígidos, con bloques fijos de clases que deben asignarse de manera constante a lo largo del semestre. En contraste, CIAC enfrenta un entorno más dinámico, donde los horarios de los tutores y profesores pueden cambiar a lo largo del semestre, y el sistema debe ser capaz de entregar nuevas asignaciones horarias rápidamente.

CAPÍTULO 3
PROPUESTA DE SOLUCIÓN

En el presente capítulo se expondrá y explicará el modelo matemático y algoritmo propuesto para solucionar el problema de asignación de horarios de tutores en CIAC. Primero se recontextualizará el problema al replantearlo como un conjunto de restricciones que se deben cumplir para una correcta asignación horaria, el cual se discute en el capítulo 3.1, para luego pasar a explicar el algoritmo propuesto.

Debido a la complejidad de la asignación horaria de las actividades del CIAC, se ha optado por un diseño de algoritmo de que permite "dividir y conquistar", dividiendo el problema en dos subproblemas y resolviendo ambos subproblemas como modelos de programación lineal entera independientes. Dividir el problema en dos resulta natural, debido a que CIAC genera sus distintos tipos de bloque horario de dos formas distintas. El primer subproblema, que se discutirá en el capítulo 3.2, se encarga de generar bloques horarios para los apoyos intensivos impartidos por CIAC en base de la disponibilidad de los tutores y profesores colaboradores correspondientes, para luego asignar los correspondientes tutores y profesores colaboradores a dichos bloques creados. Por otro lado, el segundo subproblema, que se discutirá en el capítulo 3.3, se encarga únicamente de realizar la asignación tutores para los bloques de consultoría y bloques de administración, donde estos bloques horarios ya han sido creados con anticipación.

Debido al posible solapamiento que se podría generar en la asignación horaria de los tutores y profesores colaboradores al resolver cada modelo de forma independiente, no es posible simplemente resolver ambos modelos de forma paralela. Además, aunque esto fuese posible, es necesario poder unir ambas soluciones en una única solución total y verificar que esta nueva solución sea una solución factible, es decir, no viole ninguna de las restricciones propuestas. Este algoritmo de integración se explicará en el capítulo 3.4, el cual estará acompañado de pseudo-código para expresar la lógica de este algoritmo.

3.1. Contexto

Para poder satisfacer la asignación de horarios del CIAC, un modelo matemático debe ser capaz de tener en consideración las siguientes restricciones:

3.1.1. Restricciones duras

1. Cada tipo de tutor o profesor colaborador se le deben asignar actividades correspondientes a su labor dentro del CIAC. Por ejemplo, se debe evitar que tutores adminis-

trativos se le asignen bloques de consultoría académica.

2. Los turnos de trabajo de los tutores deben ajustarse a la disponibilidad que ellos mismos entregaron. Cabe destacar que esta disponibilidad puede llegar a ser muy volátil para algunos de los estudiantes, produciendo que horarios generados para una semana puedan quedar inválidos para la siguiente.

3. Cada estudiante tiene un límite de horas que puede trabajar como ayudante dentro de la universidad. Este límite es compartido con otras ayudantías que el estudiante podría estar haciendo dentro de la universidad.

4. Los apoyos intensivos nunca se deben realizar el mismo día que el día que corresponde al certamen del asignatura respectivo.

5. Los apoyos intensivos que están a cargo de profesores deben ajustarse a la disponibilidad de los profesores.

6. Otras actividades que no tienen un bloque fijo semanal se deben contabilizar para el límite de horas trabajadas mensualmente, pero no para la disponibilidad horaria del estudiante. Algunas de estas actividades son, confección de guía y pauta, tutorías integrales, soporte técnico, administración general.

7. Los apoyos intensivos *online* se les debe asignar un *link* único para la realización de éste durante el semestre. Dicho *link* está ligado a una cuenta digital en particular. Esto es similar a como funciona la asignación de salas a las cátedras de cada asignatura.

8. Existe una cantidad limitada de cuentas digitales para la realización de apoyos intensivos *online*. Actualmente CIAC dispone de 6 cuentas digitales, lo cual implica que no se pueden realizar más de 6 actividades *online* en paralelo.

9. Se debe asegurar de que cada apoyo intensivo *online* no haga tope de cuenta digital con otro intensivo *online* en el mismo bloque. Esto es similar a evitar que se le asigne la misma sala a dos o más asignaturas al mismo tiempo para clases tradicionales.

10. La asignación de *links* debe considerar que el apoyo intensivo de un asignatura puede dictarse más de una vez a la semana.

11. La utilización de cuentas digitales para la realización de actividades debe ser lo más pareja posible entre ellas, evitando que se sobre utilice una cuenta en particular siendo que hay otras que se estén usando en menor medida.

3.1.2. Restricciones blandas

1. *Idealmente* se busca que todos los tutores tengan una cantidad equilibrada de turnos a trabajar.

2. *Idealmente* se busca que los apoyos intensivos sean asignados en días anteriores al día que corresponde el certamen del asignatura respectivo. Es decir, si un asignatura tiene certamen los días Jueves, idealmente los apoyos intensivos se asignarían para los días Martes y Miércoles.

3. A algunos tutores se les intenta dar preferencia para la realización de algunos roles.

Tal como se explicó en capítulos anteriores, CIAC genera sus distintos tipos bloques horarios de dos formas distintas; la primera consiste en crear los bloques horarios para la actividad y luego calzar a los tutores disponibles en estos bloques, usado para los bloques de consultorías y para los bloques de tutores administrativos; y la segunda consiste en crear bloques horarios para la actividad principalmente basándose en la disponibilidad de los tutores y profesores colaboradores que están asignados a dicha actividad, usado para la creación y asignación de apoyos intensivos.

Teniendo esto en consideración, este trabajo de memoria presenta una solución que se divide en dos algoritmos de programación lineal entera, lo cual permite simplificar cada modelo al no tener que considerar las peculiaridades del modelo contrario. El primer algoritmo se encarga de generar los bloques de apoyos intensivos para todos los asignaturas del semestre actual y además se encarga de asignar los respectivos tutores y profesores colaboradores a estos bloques de apoyos intensivos recién creados. El segundo algoritmo consiste únicamente en asignar tutores a actividades en bloques horarios preestablecidos, sin tener que también decidir en que horario se realizará cada actividad. Este último algoritmo sería utilizado para la asignación de horarios para los bloques de consultoría y para los bloques de tutores administrativos.

Al ser dos algoritmos distintos existiría la posibilidad de que al ejecutar ambos estos asignen a la misma persona en el mismo bloque a dos actividades distintas. Para evitar esta situación, estos 2 algoritmos no se utilizarán en paralelo, si no que se ejecutará primero el algoritmo de generación de intensivos y las asignaciones generadas por dicho algoritmo se descontarán de la disponibilidad horaria de los tutores, de modo que al ejecutar el segundo algoritmo este se le hará imposible asignar actividades a tutores en horarios que ya tengan asignados para la realización de intensivos.

A continuación se presentarán los dos algoritmos, ambos diseñados como modelos de programación de lineal entera.

3.2. Modelo de generación de apoyos intensivos

3.2.1. Dominios

Se define la siguiente nomenclatura para poder referirse a los distintos tipos de conjuntos. Estos conjuntos se utilizan como subíndices para las variables de decisión y para las constantes. Toman valores desde 0 hasta ($largo\_del\_conjunto - 1$).

- $i \in [0, I-1]$: Identificador único de cada persona. Usado para los tutores y profesores colaboradores. Por ejemplo, si consideramos un conjunto de 4 personas (Felipe, Valentina, Ignacio y Javiera) se les asignarían los índices 0, 1, 2 y 3 respectivamente.
- $t \in [0, C-1]$: Tarea a realizar. Sea por ejemplo una tarea de bloque de tutor administrador, tarea de apoyo intensivo de la asignatura de MAT021, tarea de consultoría de programación, etc. Cada apoyo intensivo está dividido como dos tareas distintas, uno para el apoyo intensivo en si mismo y un duplicado para la labor que debe desempeñar un tutor administrativo.
- $b \in [0, H-1]$: Bloque horario. Se define de forma consecutiva para todos los bloques de la semana, es decir si el bloque 9 es el último bloque del día lunes entonces el bloque 10 es el primer bloque del día martes. Debido a que hay 10 bloques al día y 5 bloques a la semana se definen 50 bloques en total. La figura 1 muestra un ejemplo de asignación de índices para cada bloque de la semana.

	Bloque UTFSM	Lunes	Martes	Miércoles	Jueves	Viernes
08:15 - 09:25	1-2	0	10	20	30	40
09:40 - 10:50	3-4	1	11	21	31	41
11:05 - 12:15	5-6	2	12	22	32	42
12:30 - 13:40	7-8	3	13	23	33	43
13:40 - 14:40	Almuerzo	4	14	24	34	44
14:40 - 15:50	9-10	5	15	25	35	45
16:05 - 17:15	11-12	6	16	26	36	46
17:30 - 18:40	13-14	7	17	27	37	47
18:55 - 20:05	15-16	8	18	28	38	48
20:30 - 21:40	17-18	9	19	29	39	49

Tabla 1: Ejemplo de asignación de índices para linealizar los bloques de la semana. Fuente: Elaboración propia.

- $s \in [0, U-1]$: Sala a utilizar para la realización de tareas. Esta puede referirse a una sala presencial (lugar físico dentro de la universidad) o una sala virtual (una cuenta en particular de las cuentas Zoom disponibles).

3.2.2. Constantes

Para poder construir este modelo matemático necesitamos definir conjuntos de valores constantes en el modelo. Estos valores son utilizados como valores de entrada y afectarán la generación de bloques de apoyos intensivos. Los valores específicos de cada conjunto dependen de cada instancia particular del problema.

1. A_{it} $\begin{cases} 1 & \text{Si el tutor } i \text{ puede realizar la tarea } t \text{ (por si es tutor de la asignatura } t) \\ 0 & \text{en caso contrario} \end{cases}$

2. D_{ib} $\begin{cases} 1 & \text{Si el tutor } i \text{ tiene disponibilidad en el bloque } b \\ 0 & \text{en caso contrario} \end{cases}$

3. M_i: Cantidad máxima de bloques que el tutor i puede trabajar a la semana.

   Esta constante nos permite controlar cuantos bloques a la semana podemos usar para cada tutor y profesor colaborador. Se ve afectada por factores externos como que el tutor realice ayudantías fuera del CIAC, por lo que tendría menos horas disponibles para ser utilizadas por CIAC; o incluso por factores externos como CIAC utilizando asignando actividades que no estén ligadas a bloques horarios al tutor en cuestión.

4. S_t: Cantidad exacta de bloques requeridos por semana para la tarea t.

5. F_{tb} $\begin{cases} 1 & \text{Si el bloque } b \text{ es factible para la tarea } t \\ 0 & \text{en caso contrario} \end{cases}$

   Permite controlar si una tarea en particular no se debería realizar en ciertos días o bloques específicos. La factibilidad de un bloque se determina a priori en base a motivos externos, como evitar topar con los bloques en que se suelen realizar las evaluaciones de la asignatura respectiva o buscar horarios en que los estudiantes suelan no tener clases de cátedra.

6. T_{ts} $\begin{cases} 1 & \text{Si la tarea } t \text{ se puede realizar en la sala } s \\ 0 & \text{en caso contrario} \end{cases}$

   Permite controlar que tareas se pueden realizar en cuales salas. Se utiliza principalmente para forzar que los apoyos intensivos se realicen en salas *online*.

7. L_t: Cantidad mínima de tutores que requiere una tarea t

   Ciertos intensivos requieren más de un único tutor o profesor colaborador que los dicten.

8. K_t: Tarea que está enlazada a la tarea t.

Esto es requerido debido a que nuestro modelo duplica todos los apoyos intensivos en dos tareas distintas (uno representando quienes dictan el apoyo y el otro representando quienes realizan labores administrativas). Esto nos permite saber cuales tareas deben estar enlazadas entre sí con una relación uno-es-a-uno.

9. $Q_s \begin{cases} 1 & \text{Si la sala } s \text{ debe ser considerada en la optimización de salas} \\ 0 & \text{en caso contrario} \end{cases}$

   La optimización de salas se refiere a repartir de forma equitativa la asignación de tareas entre las salas disponibles. Esta constante nos permite controlar cuales salas serán consideradas dentro de este proceso de optimización.

3.2.3. Variables de decisión

Definimos 5 conjuntos de variables que consideraremos la salida de este modelo. La primera variable X_{itb} es la variable principal de este modelo, es la cual nos permite realizar la asignación de un tutor o profesor colaborador a un bloque horario y que tarea esta persona deberá realizar. Las demás variables de decisión permiten generar la asignación de tarea a una sala y a un bloque de horario.

1. $X_{itb} \begin{cases} 1 & \text{Si al tutor } i \text{ se le asigna la tarea } t \text{ en el bloque } b \\ 0 & \text{en caso contrario} \end{cases}$

2. $P_{tb} \begin{cases} 1 & \text{Si a la tarea } t \text{ se asigna para ser realizada en el bloque } b \\ 0 & \text{en caso contrario} \end{cases}$

3. $Z_{ts} \begin{cases} 1 & \text{Si a la tarea } t \text{ se le asigna la sala } s \\ 0 & \text{en caso contrario} \end{cases}$

4. $O_{tbs} \begin{cases} 1 & \text{Si la tarea } t, \text{ bloque } b \text{ se le asigna la sala } s \\ 0 & \text{en caso contrario} \end{cases}$

5. W: Asignación de salas más alta entre todas las tareas.

   Esto es utilizado más adelante en la función objetivo para poder minimizar la cantidad de tareas asignadas cada sala y que así se puedan asignar de forma más equitativa.

3.2.4. Restricciones del modelo

Aquí construimos el conjunto de restricciones requerido para una correcta generación y asignación de bloques de apoyos intensivos.

- Solo se le puede asignar una tarea a un tutor si es que dicho tutor está capacitado para realizar esa tarea.

$$X_{itb} \leq A_{it}, \forall i, b, t \tag{1}$$

- Los bloques asignados a un tutor no pueden superar el máximo de bloques que este tiene disponible semanalmente.

$$\sum_{b} \sum_{t} X_{itb} \leq M_i, \forall i \tag{2}$$

- Se debe respetar la disponibilidad horaria de cada tutor.

$$\sum_{t} X_{itb} \leq D_{ib}, \forall i, b \tag{3}$$

- Un tutor puede tener un máximo de una tarea por bloque.

$$\sum_{t} X_{itb} \leq 1, \forall b, i \tag{4}$$

- Se debe asignar la cantidad exacta de bloques requerida por cada actividad.

$$\sum_{b} P_{tb} = S_t, \forall t \tag{5}$$

- Cada tarea de cada bloque debe ajustarse a su factibilidad correspondiente.

$$P_{tb} \leq F_{tb}, \forall t, b \tag{6}$$

- Solo asignar una tarea a una sala si solo si la tarea es posible realizarla en dicha sala.

$$Z_{ts} \leq T_{ts}, \forall t, s \tag{7}$$

- Todas las tareas deben tener una única sala asignada.

$$\sum_{s} Z_{ts} = 1, \forall t \tag{8}$$

- Si una tarea se realiza, esta debe tener la cantidad mínima de tutores requerida.

$$\sum_{i} X_{itb} \geq L_t * P_{tb}, \forall t, b \tag{9}$$

- Si una tarea no se realiza, no se deben asignar tutores a esta.

$$X_{itb} \leq P_{tb}, \forall i, t, b \tag{10}$$

- Asignar el mismo bloque a las tareas que requieran estar enlazadas entre si.

$$P_{tb} = P_{K_t b}, \forall t, b \tag{11}$$

Se requiere esta restricción debido a que algunas tareas como los intensivos deben tener su sala enlazada a tareas administrativas.

- Asignación de tareas por sala a ser optimizado.

$$\sum_t Z_{ts} * Q_s \leq W, \forall s \tag{12}$$

- Prohibir dos tareas en un mismo bloque en una misma sala.

$$\sum_t P_{tb} * Z_{to} \leq 1, \forall b, o \tag{13}$$

Idealmente se tendría una restricción como la anterior, pero tanto P_{tb} como Z_{to} son variables de decisión, por lo que no es posible multiplicarlas en un modelo de programación lineal.

Como estas 2 variables de decisión son *booleanas*, es posible reescribir esta restricción sin multiplicar variables de decisión al introducir una variable auxiliar y un subconjunto de restricciones:

- Solo es posible realizar una única tarea en una sala al mismo tiempo.

$$\sum_t O_{tbs} \leq 1, \forall b, s \tag{14}$$

- Con las siguientes 2 restricciones podemos simular la restricción 13. Visualmente, estas restricciones representan la tabla de verdad 2.

P_{tb}	Z_{ts}	O_{tbs}
1	1	1
1	0	0
0	1	0
0	0	0

Tabla 2: Visualización de valores para modelar multiplicación de variables.
Fuente: Elaboración propia.

$$P_{tb} + Z_{ts} \geq 2 \cdot O_{tbs}, \forall t, b, s \tag{15}$$

$$P_{tb} + Z_{ts} - 1 \leq 2 \cdot O_{tbs}, \forall t, b, s \tag{16}$$

3.2.5. Función objetivo

Un aspecto importante a tener en consideración es que, debido a la naturaleza *online* de muchas de las actividades que realiza CIAC, las salas en que se dictan estas actividades realmente corresponde a una cuenta de Zoom. Debido a esto se busca que la asignación de salas se reparta de forma equitativa. El objetivo es tener mayor holgura en caso de posibles imprevistos durante el semestre, como requerir asignar actividades adicionales de una única ocasión correspondientes a una de las tareas. Teniendo esta holgura es posible generar bloques adicionales de la tarea en cuestión sin tener que cambiar la actividad de sala *online*, lo cual se traduce a que se puede reutilizar el mismo enlace de la reunión Zoom preasignado.

Lo anterior es modelado minimizando la cantidad más alta de tareas asignadas a una sala.

$$\text{mín}\{W\} \tag{17}$$

Donde la definición de W está "empujada" por la restricción de la ecuación 12.

3.3. Modelo de asignación de consultorías

Este modelo es muy similar al anterior, pero es más sencillo en muchos aspectos debido a que no tiene que crear los bloques horarios requeridos, estos son ingresados al modelo como datos de entrada.

3.3.1. Dominios

La nomenclatura utilizada es la misma que en el modelo anterior, con la diferencia de que ya no se utiliza el subíndice s debido a que no es necesario realizar asignación de salas por la naturaleza misma de estas actividades.

- $i \in [0, I-1]$: Identificador único de cada persona. Usado para los tutores y profesores colaboradores. Por ejemplo, si consideramos un conjunto de 4 personas (Felipe, Valentina, Ignacio y Javiera) se les asignarían los índices 0, 1, 2 y 3 respectivamente.
- $t \in [0, C-1]$: Tarea a realizar. Sea por ejemplo una tarea de bloque de tutor administrador, tarea de apoyo intensivo de la asignatura de MAT021, tarea de consultoría de programación, etc.
- $b \in [0, H-1]$: Bloque horario. Se define de forma lineal para todos los bloques de la semana, es decir si el bloque 9 es el último bloque del día lunes entonces el bloque 10

es el primer bloque del día martes. Debido a que hay 10 bloques al día y 5 bloques a la semana se definen 50 bloques en total. La figura 1 muestra un ejemplo de asignación de índices para cada bloque de la semana.

3.3.2. Constantes

Al igual que en el modelo anterior, se define un conjunto de valores constantes en el modelo, los cuales se utilizarán como valores de entrada para la generación de los resultados.

1. A_{it} $\begin{cases} 1 & \text{Si el tutor } i \text{ puede realizar la tarea } t \\ 0 & \text{en caso contrario} \end{cases}$
2. D_{ib} $\begin{cases} 1 & \text{Si el tutor } i \text{ está disponible en el bloque } b \\ 0 & \text{en caso contrario} \end{cases}$
3. M_i: Cantidad máxima de bloques que un tutor i puede trabajar a la semana.
4. E_{tb}: Cantidad exacta de tutores requeridos para la tarea t en el bloque b.

3.3.3. Variables de decisión

Definimos 2 conjuntos de variables que consideraremos la salida de este modelo.

Al igual que en el modelo anterior, la variable X_{itb} es la variable principal del modelo, tomando el mismo rol de decidir la asignación de un tutor o profesor colaborador a un bloque horario y a una tarea correspondiente.

La segunda variable (Y_t) la utilizaremos para poder minimizar la carga asignada a los tutores, de modo que la asignación de tareas se produzca de forma equitativa y evitemos que a un único tutor se le asignen muchos más bloques que a los demás incluso si esta persona tiene la disponibilidad horaria y disponibilidad de cantidad de bloques que puede llegar a trabajar.

1. X_{itb} $\begin{cases} 1 & \text{Si al tutor } i \text{ se le asigna la tarea } t \text{ en el bloque } b \\ 0 & \text{en caso contrario} \end{cases}$
2. Y_t $\Big\{$Asignación de bloques más alta entre todos los tutores, según tipo de tarea

3.3.4. Restricciones del modelo

Aquí se presenta como se modelaron las restricciones para la asignación de bloques de consultoría y bloques de tutor administrador.

- Los bloques asignados a un tutor no pueden superar el máximo de bloques que este tiene disponible semanalmente.

$$\sum_{b}\sum_{t} X_{itb} * A_{it} \leq M_i, \forall i \tag{18}$$

- Cada tarea por cada bloque debe tener la cantidad exacta de tutores requerida.

$$\sum_{i} X_{itb} * A_{it} = E_{tb}, \forall b, t \tag{19}$$

- Se debe respetar la disponibilidad horaria de cada tutor.

$$\sum_{t} X_{itb} \leq D_{ib}, \forall i, b \tag{20}$$

- Un tutor puede tener un máximo de una tarea por bloque.

$$\sum_{t} X_{itb} \leq 1, \forall b, i \tag{21}$$

- Asignación de carga semanal por tutor para ser optimizado.

$$\sum_{b}\sum_{t} X_{itb} \leq Y_t, \forall i \tag{22}$$

3.3.5. Función objetivo

Se busca minimizar la carga (cantidad de bloques asignados) más alta de entre todos los tutores, según el tipo de tarea, de modo que esta carga se reparta de forma más equitativa entre todos.

$$\text{mín}\{Y_t\}, \forall t \tag{23}$$

Donde la definición de Y es “empujada” por la ecuación 22,

3.4. Integración de ambos modelos

Tal como se explicó anteriormente, ejecutar ambos modelos de forma paralela podría generar soluciones infactibles debido a que cada modelo no está teniendo en consideración las asignaciones que está realizando el otro modelo.

Para evitar este problema, ejecutamos los modelos en serie. Por ende, primero ejecutamos el primer modelo, usamos los resultados de este primer modelo para actualizar los datos de la instancia del segundo modelo (como la disponibilidad de los tutores y la cantidad máxima de bloques que cada tutor puede trabajar a la semana), y luego ejecutamos el segundo modelo. De este modo, el segundo modelo tendrá en consideración las asignaciones realizadas por el primer modelo.

Esta forma de abordar el problema no es del todo ideal, debido a que puede llevar a que el segundo modelo, en algunas ocasiones, genere soluciones infactibles. Una forma sencilla de enfrentarse a esta situación consiste en que si el segundo modelo genera una solución infactible entonces volvemos a empezar y generamos una nueva primera solución utilizando el primer modelo, la cual volvemos a utilizar como entrada para el segundo modelo. El algoritmo 1 bosqueja esta forma de resolver el problema, encerrando la generación de ambas soluciones dentro de un ciclo infinito, el cual solo se detendrá una vez que se encuentren soluciones factibles para ambos modelos. Al mismo tiempo, la forma en que está diseñado este algoritmo permite que lo invoquemos reiteradas veces para poder obtener distintos pares de soluciones, para así poder ofrecer variadas propuestas de asignaciones horarias y que el CIAC pueda seleccionar la que mejor prefiera.

Algoritmo 1 Algoritmo para generar ambas soluciones

```
procedure generar_ambas_soluciones
    while true do
        semilla ← generar_semilla()
        ins_intensivos ← generar_ins_intensivos()
        sol_intensivos ← resolver_modelo(ins_intensivos, semilla)
        if not sol_factible(sol_intensivos) then
            continue
        end if

        semilla ← generar_semilla()
        ins_consultorias ← generar_ins_consultorias()
        ins_consultorias ← actualizar(ins_consultorias, sol_intensivos)
        sol_consultorias ← resolver_modelo(ins_consultorias, semilla)
        if sol_factible(sol_consultorias) then
            return (sol_intensivos, sol_consultorias)
        end if
    end while
end procedure
```

3.5. Conclusiones

Este capítulo abordó la solución propuesta al problema, en el cual se ha recontextualizado el problema de asignación de horarios de CIAC, abordándolo desde un enfoque que considera un conjunto de restricciones esenciales. Estas restricciones, tanto duras como blandas, son fundamentales para garantizar una asignación efectiva y equitativa de los tutores y profesores colaboradores, al mismo tiempo que modelan las realidad y necesidades del CIAC de forma ad-hoc para este trabajo.

La solución propuesta consiste en un algoritmo que se basa en un enfoque de dividir y conquistar, descomponiendo el problema en dos subproblemas independientes: la generación de bloques de apoyos intensivos y la asignación de tutores a bloques predefinidos. Esta división evita que un único modelo tenga que lidiar y abarcar los distintos tipos de asignación horaria que realiza el CIAC. Debido a esta división es necesario integrar las soluciones entregadas por ambos modelos posteriormente.

La posterior integración de soluciones es crucial para evitar conflictos en las asignaciones horarias. Al ejecutar los modelos en serie, se asegura que la disponibilidad de los tutores se actualice según las asignaciones previas, evitando el riesgo de solapamientos.

CAPÍTULO 4
VALIDACIÓN DE LA SOLUCIÓN

Para validar que las soluciones generadas por los modelos propuestos en el capítulo anterior se utilizaron datos reales de disponibilidad horaria de los tutores CIAC del año 2024, segundo semestre. Esto contempla:

- I: Un total de 61 colaboradores: 53 tutores y 8 profesores colaboradores.
- C: 41 tareas: 17 apoyos intensivos *online* + 17 respectivas labores de administración para esos intensivos *online*, 6 bloques de consultoría y la labor de administración presencial.
- H: 50 bloques a la semana: 10 bloques diarios por 5 días.
- U: 7 'salas': 6 cuentas online Zoom y una 'sala' adicional para referirse a las actividades presenciales.

Para la generación de soluciones se utilizó el *solver* **IBM ILOG CPLEX 22.1.1.0**. Este fue utilizado a través de la interfaz del lenguaje de programación Python que este provee, la cual permite mayor flexibilidad para expresar los modelos en comparación a las herramientas nativas que entrega CPLEX. Estas pruebas de ejecución fueron realizadas en un computador personal Intel Core i5-6400 2.7GHz y 12 GiB RAM.

Bajo las condiciones antes nombradas, el algoritmo propuesto es capaz de generar soluciones factibles demorándose un tiempo variable, el cual suele rondar entre 20 segundos y 1 minuto con 30 segundos. Además este algoritmo genera una planilla *Excel*, para ordenar y representar los horarios generados de manera adecuada para fácil lectura. La figuras 7, 9 y 11 muestran un ejemplo de solución por el algoritmo en cuestión, mientras que las figuras 8, 10 y 12 muestran una segunda solución generada con este algoritmo.

	Lu	Ma	Mi	Ju	Vi
1-2	BENJAMIN URRUTIA	CATALINA OLGUIN	DULIA ESCALANTE	CAMILO CAMPOS	YOSELIN PRIETO
3-4	JAVIERA MONTERO	YOSELIN PRIETO	FERNANDO VEGA	CAMILO CAMPOS	NICOLAS BARAHONA
5-6	NICOLAS BARAHONA	JAVIERA MONTERO	JAVIERA ROJAS	JAVIERA ROJAS	JAVIERA DIAZ
7-8	IGNACIO DE LA ROSA	JOSE GUERRERO	DULIA ESCALANTE	JOSE GUERRERO	MARIA JESUS BASAURE
Bloque almuerzo	IGNACIO DE LA ROSA	CATALINA OLGUIN	DULIA ESCALANTE	JOSE GUERRERO	CAMILO CAMPOS
9-10	MAXIMILIANO OLGUIN	CATALINA OLGUIN	ARANTZA TIRAPEGUY	YOSELIN PRIETO	JORGE MAGAÑA
11-12	FERNANDO VEGA	ARANTZA TIRAPEGUY	FRANCISCO LOPEZ	JAVIERA MONTERO	ALAN FARIAS
13-14	FRANCISCO LOPEZ	MAXIMILIANO OLGUIN	FRANCISCO LOPEZ	JAVIERA CORTES	CASSIDY ALLENDES
15-16	JAVIERA DIAZ	ARANTZA TIRAPEGUY	ANAIS SOZA	FELIPE ROMERO	SCARLETTE BAEZA
17-18					

Figura 7: Primer ejemplo de asignación horaria generada para tutores administrativos, únicamente en actividades presenciales.

Fuente: Elaboración propia

	Lu	Ma	Mi	Ju	Vi
1-2	JOSE GUERRERO	CATALINA OLGUIN	DULIA ESCALANTE	NICOLAS BARAHONA	YOSELIN PRIETO
3-4	FRANCISCO LOPEZ	YOSELIN PRIETO	FERNANDO VEGA	JAVIERA MONTERO	NICOLAS BARAHONA
5-6	IGNACIO DE LA ROSA	YOSELIN PRIETO	JAVIERA ROJAS	JAVIERA MONTERO	JAVIERA DIAZ
7-8	IGNACIO DE LA ROSA	FELIPE BAHAMONDEZ	DULIA ESCALANTE	JAVIERA ROJAS	MARIA JESUS BASAURE
Bloque almuerzo	DULIA ESCALANTE	FELIPE BAHAMONDEZ	MAXIMILIANO OLGUIN	BENJAMIN URRUTIA	CAMILO CAMPOS
9-10	MAXIMILIANO OLGUIN	CATALINA OLGUIN	CAMILO CAMPOS	JAVIERA MONTERO	JORGE MAGAÑA
11-12	FRANCISCO LOPEZ	MARIA JESUS BASAURE	CASSIDY ALLENDES	JAVIERA CORTES	ALAN FARIAS
13-14	FRANCISCO LOPEZ	ARANTZA TIRAPEGUY	VALENTINA FLORES	FELIPE ROMERO	SCARLETTE BAEZA
15-16	JOSE GUERRERO	ARANTZA TIRAPEGUY	FELIPE ROMERO	JAVIERA CORTES	SCARLETTE BAEZA
17-18					

Figura 8: Segundo ejemplo de asignación horaria generada para tutores administrativos, únicamente en actividades presenciales.

Fuente: Elaboración propia

	Lu	Ma	Mi	Ju	Vi
1-2					
3-4					
5-6					
7-8	MATIAS FARIAS	ITALO BOLELLI	MATIAS FARIAS	ITALO BOLELLI	ITALO BOLELLI
Bloque almuerzo					
9-10	MATIAS FARIAS	LUCAS CONTRERAS	CLEMENTE MUJICA	AGUSTIN LEIVA	JAVIER MESA
11-12	AGUSTIN LEIVA	JAVIER MESA	JAVIER MESA	MATIAS OLIVARES	SEBASTIAN MEDINA
13-14	AGUSTIN LEIVA	LUCAS CONTRERAS	BENJAMIN GALVEZ	MATIAS OLIVARES	LUCAS CONTRERAS
15-16	BENJAMIN GALVEZ	MATIAS OLIVARES	CLEMENTE MUJICA	CLEMENTE MUJICA	BENJAMIN GALVEZ
17-18					

Figura 9: Primer ejemplo de asignación horaria generada para tutores académicos de MATFIS.

Fuente: Elaboración propia

	Lu	Ma	Mi	Ju	Vi
1-2					
3-4					
5-6					
7-8	ITALO BOLELLI	MATIAS OLIVARES	MATIAS FARIAS	MATIAS OLIVARES	MARTINA MUÑOZ
Bloque almuerzo					
9-10	ITALO BOLELLI	JAVIER MESA	SEBASTIAN MEDINA	AGUSTIN LEIVA	SEBASTIAN MEDINA
11-12	ITALO BOLELLI	JAVIER MESA	MATIAS FARIAS	CLEMENTE MUJICA	SEBASTIAN MEDINA
13-14	AGUSTIN LEIVA	JAVIER MESA	MATIAS FARIAS	CLEMENTE MUJICA	LUCAS CONTRERAS
15-16	BENJAMIN GALVEZ	MATIAS OLIVARES	CLEMENTE MUJICA	BENJAMIN GALVEZ	BENJAMIN GALVEZ
17-18					

Figura 10: Segundo ejemplo de asignación horaria generada para tutores académicos de MATFIS.

Fuente: Elaboración propia

	Lu	Ma	Mi	Ju	Vi
13-14	**inten_mat023** Prof. Antonio Morales JAVIERA DIAZ 019	**inten_fis120** Prof. Diego Teca MARIA JESUS BASAURE 019	**inten_mate20** Prof. Jefferson Prada FELIPE ROMERO 018	**inten_mate11** Prof. Jefferson Prada JORGE MAGAÑA 020	**inten_mat021** Prof. Antonio Morales ALAN FARIAS 015
	inten_mat024 Prof. Pablo Madariaga SCARLETTE BAEZA 018	**inten_iwi131** ANGHELO CARVAJAL CASSIDY ALLENDES 020	**inten_mat061** Prof. Alejandro Vidal JAVIERA ROJAS 015	**inten_mat071** Prof. Alejandro Vidal FELIPE ROMERO 015	**inten_mat024** Prof. Pablo Madariaga SCARLETTE BAEZA 018
	inten_fis100 Prof. Gustavo Pulgar BENJAMIN URRUTIA 017	**inten_qui010** FRANCISCO MELLADO JORGE MAGAÑA 015	**inten_fis110** Prof. Luciano Bravo VALENTINA FLORES 019		
15-16	**inten_mat021** Prof. Antonio Morales BENJAMIN URRUTIA 015	**inten_iwi131** ANGHELO CARVAJAL VALENTINA FLORES 020		**inten_mat060** Prof. Alejandro Vidal VALENTINA FLORES 015	**inten_mat022** Prof. Fabian Ramirez CASSIDY ALLENDES 017
	inten_mat022 Prof. Fabian Ramirez JAVIERA CORTES 017	**inten_qui010** FRANCISCO MELLADO FERNANDO VEGA 015		**inten_fis120** Prof. Diego Teca JAVIERA CORTES 019	**inten_mat023** Prof. Antonio Morales ALAN FARIAS 019
	inten_fis110 Prof. Luciano Bravo ANAIS SOZA 019			**inten_inf130** ANNETTE RAMIREZ ANAIS SOZA 016	
17-18					

Figura 11: Primer ejemplo de asignación horaria generada para apoyos intensivos, incluyendo además el profesor encargado de dictar el apoyo intensivo, el tutor administrador a cargo y la cuenta de Zoom a utilizar.

Fuente: Elaboración propia

	Lu	Ma	Mi	Ju	Vi
13-14	**inten_mat022** Prof. Fabian Ramirez ARANTZA TIRAPEGUY 019	**inten_mate11** Prof. Jefferson Prada MARIA JESUS BASAURE 016	**inten_iwi131** ANGHELO CARVAJAL JAVIERA ROJAS 019	**inten_mate20** Prof. Jefferson Prada JAVIERA CORTES 015	**inten_mat061** Prof. Alejandro Vidal CASSIDY ALLENDES 015
	inten_mat024 Prof. Pablo Madariaga JAVIERA DIAZ 017	**inten_mat023** Prof. Antonio Morales MAXIMILIANO OLGUIN 019		**inten_mat071** Prof. Alejandro Vidal JORGE MAGAÑA 019	**inten_mat021** Prof. Antonio Morales ALAN FARIAS 016
	inten_fis110 Prof. Luciano Bravo BENJAMIN URRUTIA 018	**inten_qui010** FRANCISCO MELLADO JORGE MAGAÑA 017		**inten_mat024** Prof. Pablo Madariaga SCARLETTE BAEZA 017	
15-16	**inten_mat021** Prof. Antonio Morales BENJAMIN URRUTIA 016	**inten_mat060** Prof. Alejandro Vidal FERNANDO VEGA 015	**inten_mat023** Prof. Antonio Morales ANAIS SOZA 019	**inten_fis120** Prof. Diego Teca ANAIS SOZA 017	**inten_mat022** Prof. Fabian Ramirez ALAN FARIAS 019
	inten_fis110 Prof. Luciano Bravo JAVIERA DIAZ 018	**inten_iwi131** ANGHELO CARVAJAL ANAIS SOZA 019		**inten_inf130** ANNETTE RAMIREZ VALENTINA FLORES 019	**inten_fis100** Prof. Gustavo Pulgar CASSIDY ALLENDES 020
	inten_fis120 Prof. Diego Teca FERNANDO VEGA 017	**inten_qui010** FRANCISCO MELLADO VALENTINA FLORES 017			
17-18					

Figura 12: Segundo ejemplo de asignación horaria generada para apoyos intensivos, incluyendo además el profesor encargado de dictar el apoyo intensivo, el tutor administrador a cargo y la cuenta de Zoom a utilizar.

Fuente: Elaboración propia

4.1. Análisis de resultados

De ambas soluciones presentadas anteriormente, podemos ver que ambas son soluciones factibles. Por ejemplo, para el semestre actual (año 2024, segundo semestre), CIAC requiere tener exactamente 1 tutor administrador en sus correspondientes bloques de Lunes a Viernes, desde el bloque 1-2 hasta el bloque 15-16, lo cual es posible ver que se cumple en ambas soluciones. Lo mismo ocurre con los bloques de tutoría académica de MATFIS, en donde es requerido un tutor correspondiente de Lunes a Viernes, en los bloques 7-8 hasta el 15-16, sin contar el bloque de almuerzo.

Por otro lado, cada apoyo intensivo tiene sus propias restricciones con respecto a que día puede ser realizado, por ejemplo CIAC requiere que los apoyos intensivos de “Programación” (IWI131) deben ser dictados los días anteriores al día Jueves, los apoyos intensivos de “Química” (QUI010) deben ser dictados los días Lunes o Martes, y los apoyos intensivos de “Programación para la gestión” (INF130) deben ser dictados los días Jueves. Es posible ver que ambas soluciones siguen estas restricciones. CIAC requiere que los apoyos intensivos de las siguientes asignaturas sean dictados dos veces a la semana durante este semestre: MAT021, MAT022, MAT023, MAT024, FIS110, FIS120, IWI131, QUI010, mientras que los siguientes deberían ser dictados únicamente una vez a la semana: MATE11, MATE20, MAT060, MAT061, MAT071, FIS100, INF130; ambas soluciones siguen estas cantidades especificas de veces que debe dictarse cada apoyo apoyo intensivo.

CIAC también requiere que a cada apoyo intensivo se le asigne la misma cuenta Zoom en cada ocasión que este apoyo es realizado. Ambas soluciones mostradas siguen esta restricción. Por ejemplo, en la figura 11 ambos bloques de apoyo intensivo de MAT023 (Lunes 13-14 y Viernes 15-16) tienen asignada la cuenta Zoom con el código 019, mientras que IWI131 (Martes 13-14 y Martes 15-16) utilizan la cuenta 020; en la figura 12 los apoyos intensivos de MAT024 (Lunes 13-14 y Jueves 13-14) usan ambos la cuenta 017, mientras que FIS120 (Lunes 15-16 y Jueves 15-16) usan la cuenta 017. También es posible ver que ninguna cuenta Zoom es utilizada para apoyos intensivos distintos que son dictados en el mismo bloque. Otra restricción que requiere CIAC es que ningún tutor dos o más bloques distintos en el mismo horario; lo cual es posible notar al comprar las asignaciones de los tutores administrativos de las figuras 7 y 8 con las figuras 11 y 12 respectivamente.

Con respecto a la carga de los tutores, siendo la carga la cantidad de bloques semanales asignados a la persona, a ningún tutor se le asignó más de 3 bloques a la semana en ambas soluciones. En el caso de los tutores administradores, la gran mayoría tiene una carga de 3 bloques semanales, donde tan solo 4 personas (en la primera solución) y 7 personas (en la segunda solución) tienen solo 2 bloques de carga semanal, lo cual se traduce en una carga promedio de 2.72 bloques por persona. En el caso de los bloques de consultoría de MATFIS (figuras 9 y 10) la gran mayoría también tiene una carga de 3 bloques semanales, donde tan solo 2 personas (en la primera solución) y 3 personas (en la segunda solución) tienen menos de 3 bloques de carga semanal, dando una carga promedio de 2.5 bloques por persona a la semana. Por lo tanto, se concluye que el balance de carga funciona como es esperado.

4.2. Validación de la solución

Como métrica para medir el desempeño de esta solución propuesta se decidió comparar cuantas soluciones factibles es posible obtener en el mismo tiempo correspondiente a lo que se demora la persona encargada de realizar los horarios de los tutores y profesores colaboradores de CIAC. Para este ejercicio se utilizó tanto datos históricos de disponibilidad horaria de los tutores como disponibilidad horaria correspondiente a este semestre. Tal como se explicó anteriormente en este trabajo de memoria, el encargado demora 2 días en realizar una única asignación horaria, por lo que se decidió medir cuantas soluciones factibles es posible generar en el tiempo equivalente a 2 días laborales, en otras palabras, 18 horas. Por lo cual, se modificó el algoritmo para que pueda ser ejecutado durante este tiempo y que además contabilice la cantidad de soluciones factibles que este genere. En dicho tiempo fue posible generar alrededor de 1800 posibles asignaciones horarias factibles, aproximadamente 100 soluciones factibles por hora de ejecución.

Para poder verificar que las soluciones generadas efectivamente se adecúan a las necesidades y restricciones del CIAC, se realizaron pruebas a este sistema bajo la tutela del encargado de realizar las asignaciones horarias de CIAC. También se usaron tanto datos históricos como datos actuales de disponibilidad horaria. Idealmente las soluciones generadas se hubieran puesto a prueba usándolas como los horarios reales para los tutores y profesores colaboradores de CIAC, pero debido a la temporalidad y agenda del CIAC, esto fue imposible. Por ende, se optó por únicamente verificar de forma manual las soluciones generadas.

Estas pruebas dejaron muy contento y satisfecho al encargado de CIAC, especialmente con la rapidez con la cual se genera una asignación horaria y con la facilidad con la que se pueden generar variadas propuestas de asignación horaria. Al recolectar su *feedback*, el encargado solicitó que las planillas generadas hicieran uso de colores para poder distinguir mejor por tipo de tutor y su tarea. Además solicitó mayor granularidad entre los tipos de tareas, permitiendo diferenciar entre tutores administrativos que puedan realizar solo actividades presenciales, solo actividades virtuales o ambas, lo cual el algoritmo puede soportar actualmente con tan solo ajustes en las constantes de la instancia. También el encargado solicitó que el sistema final de asignación horaria tenga una interfaz usuaria sencilla y “a prueba de todo tipo de usuario”, lo cual escapa al enfoque de este trabajo de memoria pero puede ser abordado como trabajo futuro.

Finalmente, el encargado solicitó de forma adicional nuevas restricciones blandas para la asignación de horarios. La primera consiste en poder especificar que algunos apoyos intensivos idealmente deberían poder ser realizados en dos bloques contiguos del mismo día en vez de simplemente permitir que se realicen en días separados. La segunda consiste en poder especificar que un tutor administrador asignado a un apoyo intensivo *online* idealmente se encargaría de administrar todos los bloques en que se dicte dicho apoyo intensivo, intentando evitar que un mismo apoyo intensivo dictado en dos bloques distintos se le asignen tutores administradores separados. Debido a que estas restricciones no fueron planteadas con la suficiente anterioridad y debido a que son restricciones blandas o no estrictamente

requeridas, estas no se consideraron para este trabajo de memoria, pero también pueden ser abordadas como trabajo futuro.

CAPÍTULO 5
CONCLUSIONES

En este trabajo de memoria se ha presentado una solución basada en un algoritmo de programación lineal entera para abordar el problema de asignación de horarios en el Centro Integrado de Aprendizaje en Ciencias Básicas (CIAC) de la Universidad Técnica Federico Santa María (UTFSM). Debido a la complejidad de la asignación horaria de CIAC se ha optado por dividir el problema en dos submodelos de programación lineal: uno para la generación de bloques de apoyos intensivo y otro para la asignación de consultorías académicas y administrativas, facilitando el planteamiento de la solución final. Esta división del problema resulta natural debido a que representa mejor la forma en que el CIAC agrupa sus distintas actividades.

Los resultados obtenidos validaron que la solución propuesta reduce considerablemente el tiempo necesario para realizar las asignaciones horarias, cumpliendo con las restricciones requeridas por el CIAC. Además se midió que es posible generar miles de asignaciones horarias distintas y factibles en el mismo periodo de tiempo que anteriormente requería varios días de trabajo manual.

Una de las principales ventajas de esta propuesta de solución es la flexibilidad que provee, debido a que es posible generar múltiples propuestas de asignación horaria en cuestión de minutos, permitiendo que el CIAC pueda simplemente decidir cual opción le parece mejor y posteriormente adaptarla de forma manual en caso de ser necesario.

Aunque la solución presentada cumple con los objetivos planteados, existen áreas de mejora y expansión como trabajo futuro. Se sugiere integrar con el sistema actual de encuestas utilizado para recolectar la disponibilidad horaria de los tutores. Esto permitiría automatizar el proceso de actualización de datos, reduciendo aún más la intervención manual.

Otro aspecto que podría optimizarse es la incorporación de una interfaz gráfica de usuario que permita a los encargados de CIAC interactuar con el algoritmo de una forma más intuitiva. Una interfaz gráfica facilitaría su uso por parte de personas con menos conocimientos técnicos, mejorando la experiencia de usuario y ampliando su aplicabilidad.

Finalmente, también queda como trabajo futuro el poder integrar las restricciones que fueron planteadas durante el proceso de validación de la solución, como poder especificar ciertos apoyos intensivos que deben ser dictados en bloques contiguos o que idealmente se evite que un mismo apoyo intensivo que sea dictado en dos bloques distintos se le asignen tutores administradores distintos.

Este trabajo ha sentado las bases para una solución eficiente, adaptable y escalable al problema de asignación de horarios en el CIAC, abriendo nuevas oportunidades para la mejora continua del proceso en el futuro.

REFERENCIAS BIBLIOGRÁFICAS

[Abdelhalim y El Khayat, 2016] Abdelhalim, E. y El Khayat, G. (2016). A utilization-based genetic algorithm for solving the university timetabling problem (uga). *Alexandria Engineering Journal*, 55.

[Alvarez-Valdes *et al.*, 2002] Alvarez-Valdes, R., Crespo, E., y Tamarit, J. M. (2002). Design and implementation of a course scheduling system using tabu search. *European Journal of Operational Research*, 137(3):512–523.

[Bakir y Aksop, 2008] Bakir, M. y Aksop, C. (2008). A 0-1 integer programming approach to a university timetabling problem. *Hacettepe Journal of Mathematics and Statistics*, 37.

[de Werra, 1985] de Werra, D. (1985). An introduction to timetabling. *European Journal of Operational Research*, 19(2):151–162.

[Eledum, 2017] Eledum, H. (2017). The university timetabling problem: Modeling and solution using binary integer programming with penalty functions. *International Journal of Applied Mathematics and Statistics*, 56:164–178.

[Feizi Derakhshi *et al.*, 2012] Feizi Derakhshi, M. R., Babaei, H., y Heidarzadeh, J. (2012). A survey of approaches for university course timetabling problem.

[GOTLIEB, 1963] GOTLIEB (1963). The construction of class - teacher timetables. *IFIP Congress*, 62:73–77.

[Jat, 2008] Jat, S. (2008). A memetic algorithm for the university course timetabling problem. volumen 1, pp. 427 – 433.

[Karp, 1972] Karp, R. (1972). Reducibility among combinatorial problems. volumen 40, pp. 85–103.

[Khonggamnerd e Innet, 2009] Khonggamnerd, P. e Innet, S. (2009). On improvement of effectiveness in automatic university timetabling arrangement with applied genetic algorithm. En *2009 Fourth International Conference on Computer Sciences and Convergence Information Technology*, pp. 1266–1270.

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

[Nothegger *et al.*, 2012] Nothegger, C., Mayer, A., Chwatal, A., y Raidl, G. (2012). Solving the post enrolment course timetabling problem by ant colony optimization. *Annals OR*, 194:325–339.

[Socha *et al.*, 2002] Socha, K., Knowles, J., y Sampels, M. (2002). A max-min ant system for the university course timetabling problem. En Dorigo, M., Di Caro, G., y Sampels, M., editores, *Ant Algorithms*, pp. 1–13, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Song *et al.*, 2018] Song, T., Liu, S., Tang, X., Peng, X., y Chen, M. (2018). An iterated local search algorithm for the university course timetabling problem. *Applied Soft Computing*, 68.

[Stichting *et al.*, 1996] Stichting, C., Centrum, M., y Schaerf, A. (1996). A survey of automated timetabling. *Artificial Intelligence Review*, 13.

[Tuga *et al.*, 2007] Tuga, M., Berretta, R., y Mendes, A. (2007). A hybrid simulated annealing with kempe chain neighborhood for the university timetabling problem. pp. 400–405.

[Welsh y Powell, 1967] Welsh, D. J. A. y Powell, M. B. (1967). An upper bound for the chromatic number of a graph and its application to timetabling problems. *The Computer Journal*, 10(1):85–86.