

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA
VALPARAÍSO - CHILE



**“AMBIENTE GRAFICO PARA MALLAS CURRICULARES
BALANCEADAS”**

JAVIER IGNACIO OLIVARES FIDALGO

**MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN INFORMÁTICA**

Profesor Guía: Carlos Castro
Profesor Correferente: Hubert Hoffmann

Noviembre - 2023

DEDICATORIA

Quiero dedicar esta memoria para mis amigos/as, y familia, muchas gracias por estar ahí.
Y en especial a mi bisabuela, que en paz descanse.

AGRADECIMIENTOS

Quisiera agradecer a todas las personas que estuvieron ahí en estos años, a mi familia por todo el apoyo que me han brindado estos años, a mi padre que siempre está ahí, dispuesto a ayudar con lo que se necesita, mi madre que siempre se preocupó por mí, recibíendome con cariño cada vez que llegaba de visita a la casa. Mi bisabuela, la Yayita quien no alcanzó a verme llegar hasta aquí, pero estuvo desde el inicio apoyándome y cuidando desde que comenzó este viaje.

Quiero agradecer a los amigos que hice durante el camino, los compañeros con los que empecé la carrera y a los que hice durante el trayecto, al grupo de “Simios”, como íbamos al nido del M a estudiar y pasar el tiempo entre clases y ayudándonos constantemente en las materias que cursábamos juntos. Mis amigos de Santiago, que se juntaron para conmemorar mis estudios fuera de la región. Un agradecimiento muy especial para los “Vladis”, quienes siempre están ahí, para las buenas y las malas, con los que siempre estamos en *Discord* para cualquier cosa, ya sea jugar o pasar el rato conversando. Al Seba, mas que mi amigo eres mi hermano, y mas que mi hermano eres mi *bro*. Al Tomi, quien siempre me recibía y organizábamos algo que hacer. Y gracias a la Zele, gracias por todo tu cariño y apoyo que me has dado estos últimos años. Te estoy muy agradecido.

Y un agradecimiento a los chicos de Yoy, con ellos, di mis primeros pasos en el mundo laboral y, durante el último año, hemos enfrentado juntos múltiples desafíos. Su guía y enseñanzas han sido fundamentales en mi crecimiento, moldeando la persona profesional que soy y que aún sigue aprendiendo.

Y muchas gracias a mi profesor Guía Carlos Castro, gracias por el apoyo, los documentos y herramientas otorgadas y la constante comunicación para poder llevar a cabo este documento.

Este viaje empezó el 2013 y fue un gran cambio, partiendo porque significó ir a estudiar fuera de mi lugar habitual lejos de mis seres queridos, y que los primeros años fueron duros, reprobando y fallando fueron golpes fuertes, pero sin el apoyo de todos ustedes, no lo hubiese logrado. Se me queda grabado la frase de mi madre, “es una carrera, no importa llegar primero, sino terminarla. Ve tranquilo a tu ritmo” ...y pues aquí estamos.

RESUMEN

Resumen—Es habitual para la mayoría de los estudiantes de cualquier carrera no seguir la malla correspondiente, reprobando ramos, dejar otros para el final, o simplemente querer tomar tantos como pueda un semestre y dejar el siguiente con menos se vuelve una práctica habitual, esto puede conllevar a un retraso en el transcurso de la carrera. Es por esto por lo que se plantea implementar un ambiente gráfico con el fin de tener una mejor visualización del avance de la malla curricular del estudiante con el fin de que estos estén conscientes de su progreso, que carga académica esté balanceada y así evitar que se demore en su formación profesional.

Palabras Clave—Ambiente gráfico, carga académica balanceada, interfaz de usuario.

ABSTRACT

Abstract—*It is common for most students of any career not to follow the corresponding curriculum, fail classes, leave others for the end, or simply want to take as many as possible one semester and leave the next one with less becomes a common practice, this can lead to a delay during the career. Therefore, it is proposed to implement a graphic environment to have a better visualization of the progress of the student's curriculum so that they are aware of their progress, that the academic load is balanced and thus avoid delay in professional training.*

Keywords— *Graphic environment, balanced academic curriculum, user interface.*

GLOSARIO

Aquí se deben colocar las siglas mencionadas en el trabajo y su explicación, por orden alfabético. Por ejemplo:

BACP: Balanced Academic Curriculum Problem

CSPLib: Constraint Solver Problem Library

TSP: Travelling Salesman Problem

UTFSM: Universidad Técnica Federico Santa María.

INDICE DE CONTENIDOS

<u>RESUMEN</u>	4
<u>ABSTRACT</u>	4
<u>GLOSARIO</u>	5
<u>INDICE DE FIGURAS</u>	9
<u>INDICE DE TABLAS</u>	10
<u>INTRODUCCIÓN</u>	11
<u>CAPÍTULO 1: DEFINICIÓN DEL PROBLEMA</u>	12
<u>1.1 Contexto</u>	12
<u>1.2 Situación actual</u>	13
<u>1.2.1 Caso de ejemplo</u>	14
<u>1.2.2 Otros estudios</u>	15
<u>1.3 Objetivos</u>	16
<u>1.3.1 Objetivo general</u>	16
<u>1.3.2 Objetivos específicos</u>	16
<u>CAPÍTULO 2: MARCO CONCEPTUAL</u>	18
<u>2.1 BACP en profundidad</u>	18
<u>2.2 Formulando BACP</u>	18
<u>2.3 Representación grafica de un BACP</u>	19
<u>2.3.1 Construcción de grafo</u>	20
<u>2.3.2 Malla curricular como grafo</u>	20
<u>CAPÍTULO 3: PROPUESTA DE SOLUCION</u>	23
<u>3.1 Enfoque de propuesta</u>	23

<u>3.1.1 Representación Gráfica con Graphviz</u>	23
<u>3.1.2 Optimización con Programación Lineal</u>	23
<u>3.2 Lectura de archivo</u>	25
<u>3.3 Modelo formulado de Programación Lineal para BACP</u>	27
<u>3.4 Restricciones de Modelo</u>	29
<u>3.4.1 Restricción por carga académica</u>	29
<u>3.4.2 Restricción por carga semestral máxima</u>	29
<u>3.4.3 Restricción cursar en algún semestre</u>	30
<u>3.4.4 Restricción carga semestral máxima</u>	30
<u>3.4.5 Restricción carga semestral mínima</u>	31
<u>3.4.6 Restricción cantidad máxima de asignaturas por semestre</u>	31
<u>3.4.7 Restricción asignaturas con prerrequisitos</u>	32
<u>3.5 Implementación gráfica con Graphviz</u>	33
<u>3.5.1 Creación de la instancia del grafo</u>	33
<u>3.5.2 Creación de Nodos y Arcos</u>	34
<u>3.5.3 Función para generar Malla Curricular</u>	35
<u>3.5.4 Testing con lista de prueba</u>	37
<u>CAPÍTULO 4: VALIDACIÓN DE LA SOLUCIÓN</u>	38
<u>4.1 Archivo utilizado</u>	38
<u>4.2 Generación de archivo de texto</u>	38
<u>4.3 Generación de modelo y uso de solver</u>	39
<u>4.4 Generación grafica de la malla curricular</u>	40
<u>4.5 Forzar restricciones fuertes</u>	46
<u>4.6 Resultados y análisis</u>	49

<u>CAPÍTULO 5: CONCLUSIONES</u>	50
<u>5.1 Conclusiones generales</u>	50
<u>5.1.1 Metodología y Herramientas de Trabajo</u>	50
<u>5.1.2 Desarrollo</u>	51
<u>5.1.3 Visualización y Representación Gráfica</u>	51
<u>5.2 Logros alcanzados</u>	52
<u>5.2.1 Objetivos específicos</u>	52
<u>5.2.2 Objetivo General</u>	52
<u>5.3 Trabajo futuro</u>	53
<u>REFERENCIAS BIBLIOGRÁFICAS</u>	55
<u>ANEXOS</u>	57

INDICE DE FIGURAS

Figura 1: Malla Curricular Ingeniería civil Informática. Fuente: Departamento de Informática.	13
Figura 2: Duración real. Fuente: Sitio web UTFSM	14
Figura 3: Duración real. Fuente: Sitio web UTFSM	15
Figura 4: Árbol del problema. Fuente: Elaboración propia.	16
Figura 5: Construcción de grafo. Fuente: Solving the Balanced Academic Curriculum Problem Using the ACO Metaheuristic.	20
Figura 6: Malla curricular como grafo horizontal. Fuente: Sitio web profesores electrónica UTFSM	21
Figura 7: Malla curricular como grafo vertical. Fuente: Sitio web Sociedad Peruana de la Computación	22
Figura 8: Flujo de propuesta de solución. Fuente: Elaboración propia.	24
Figura 9: Estructura Malla curricular en documento de texto. Fuente: Elaboración Propia	26
Figura 10: función Leer archivo y agregar restricción. Fuente: Elaboración Propia	27
Figura 11: Restricción carga académica. Fuente: Elaboración Propia	29
Figura 12: Restricción carga semestral máxima. Fuente: Elaboración Propia	30
Figura 13: Restricción cursar semestre. Fuente: Elaboración Propia	30
Figura 14: Restricción carga máxima. Fuente: Elaboración Propia	31
Figura 15: Restricción carga mínima. Fuente: Elaboración Propia	31
Figura 16: Restricción máximo de asignaturas. Fuente: Elaboración Propia	32
Figura 18 y 19: Restricción prerrequisitos y añadir a lista restricciones. Fuente: Elaboración Propia	32-33
Figura 20: Ejemplo de creación de un grafo con Graphviz. Fuente: Elaboración Propia	34
Figura 21: Ejemplo de uso función node y edge. Fuente: Elaboración Propia	35

Figura 22,23 y 24: Función Ordenar ramos, Generar Malla y Asignar Arcos.	
Fuente: Elaboración Propia	36
Figura 25: Lista de prueba. Fuente: Elaboración Propia	37
Figura 26: Malla generada a partir de lista de prueba. Fuente: Elaboración Propia	37
Figura 27: Malla académica de prueba. Fuente: Elaboración Propia	38
Figura 28 y 29: Restricciones malla de <i>Testing</i> . Fuente: Elaboración Propia	41
Figura 30 y 31: Restricciones malla de <i>Testing</i> y definición de variables.	
Fuente: Elaboración Propia	42
Figura 32: Resultado del solver. Fuente: Elaboración Propia	43
Figura 33: Modificación al coloreo en función Generar Malla.	
Fuente: Elaboración Propia	44
Figura 34: Lista basada en resultado del solver. Fuente: Elaboración Propia	45
Figura 35: Malla Curricular generada resultante. Fuente: Elaboración Propia	46
Figura 36: Reposicionamiento restricciones en solver.	
Fuente: Elaboración Propia	47
Figura 37: Lista de asignaturas de restricciones forzadas	48
Figura 38: Malla generada restricciones forzadas. Fuente: Elaboración Propia	48
Figura 39: Ejemplo código de generación de grafo. Fuente: Elaboración Propia	57
Figura 40 Resultado de generación de grafo Ejemplo. Fuente: Elaboración Propia	58
Figura 41: Código fuente en lenguaje DOT. Fuente: Elaboración Propia	58

INDICE DE TABLAS

Tabla 1: Documento de texto de malla de ejemplo Fuente: Elaboración Propia	39
--	----

INTRODUCCION

Las mallas curriculares son un instrumento esencial en la formación académica, delineando la progresión educativa a seguir para alcanzar un grado o título. No obstante, es común que muchos estudiantes se enfrenten a retrasos en su formación debido a diversos factores, algunas veces, se pierde la claridad en las relaciones y prerrequisitos entre asignaturas. Una malla curricular bien balanceada no solo garantiza una educación integral y coherente, sino que también facilita la planificación académica, permitiendo a los estudiantes visualizar su camino educativo de manera clara. Sin embargo, la comprensión de esta malla se torna compleja al considerar las múltiples relaciones y prerrequisitos entre asignaturas. Por ello, es importante contar con una herramienta gráfica que represente estas relaciones de manera estructurada y fácil de interpretar.

Ante esta necesidad, el presente documento propone una herramienta capaz de leer la malla curricular de un alumno, buscar un equilibrio y, posteriormente, generar un grafo que refleje de forma coherente una malla curricular tradicional. Esta herramienta no solo facilita la visualización de la malla, sino que también permite una comprensión más profunda de las conexiones y dependencias entre las asignaturas, esencial para una planificación académica efectiva.

Este documento introduce una herramienta capaz de interpretar la malla curricular de un estudiante, buscar un balance y proyectarla en un grafo representativo de una malla tradicional. La estructura de esta memoria se despliega en cinco capítulos esenciales: en el primero, se define la problemática de las mallas curriculares y su impacto en la progresión estudiantil; el segundo ofrece un marco conceptual que profundiza en los principios subyacentes de las mallas y su visualización; el tercero expone la herramienta propuesta, su diseño y la metodología aplicada en conjunto de funcionalidades que tienen la intención de abarcar esta problemática; el cuarto detalla la validación de esta solución, mostrando la fase de pruebas y los resultados alcanzados; y finalmente, el quinto capítulo concluye con un resumen de los resultados y se habla de áreas de mejora identificadas a lo largo del proyecto.

CAPÍTULO 1: DEFINICIÓN DEL PROBLEMA

1.1. CONTEXTO

En toda institución educacional las cuales tienen carreras universitarias, las cuales están compuestas por una malla curricular la cual sirve de referencia para tanto para docentes como para estudiantes para seguir un curso de asignaturas para la formación profesional del estudiante, existiendo similitudes de una misma carrera en distintas instituciones. Pero algo que suele ocurrir es que no siempre un estudiante sigue la malla tal como está planificada, ya sea por reprobación o querer terminar antes hay un reajuste por parte del estudiante el cual puede traer efectos adversos como sobrecarga académica, estrés universitario y/o también, atrasarse en los estudios.

Las carreras universitarias se caracterizan por repartir los cursos (generalmente en semestres) de manera en que en un semestre se impartan materias de distintas áreas, ya sean conocimientos matemáticos, el área científica, introducción a la programación y manejo de datos, e incluso ramos deportivos y otros que van enfocados en el área de humanidades, al menos en el caso de la carrera Ingeniería Civil Informática de la UTFSM. De todas maneras, que un estudiante sea capaz de terminar sus estudios en el tiempo estimado suele no ser el caso de la mayoría de éstos. Ya sea por problemas personales o falta de estudio, reprobación de asignaturas es un evento que muchos estudiantes pasan todos los semestres. También ocurre el caso de que omiten toparse con ciertos ramos un semestre debido a la dificultad que implica tenerlos el mismo tiempo. Este es el momento en el que la malla curricular de cada estudiante empieza a flaquear, y por ende corre el peligro de retrasarse, sobre todo si sus siguientes asignaturas no se imparten en el semestre actual.

A continuación, se puede observar la malla curricular de Ingeniería Civil Informática de la UTFSM, donde claramente se ve como los cursos se reparten en los semestres que corresponderían.

Malla Curricular Ingeniería Civil Informática

Plan 73 13

AÑO 1				AÑO 2				AÑO 3				AÑO 4				AÑO 5		AÑO 5 1/2	
SEMESTRE I	SEMESTRE II	SEMESTRE III	SEMESTRE IV	SEMESTRE V	SEMESTRE VI	SEMESTRE VII	SEMESTRE VIII	SEMESTRE IX	SEMESTRE X	SEMESTRE XI									
IWI-131 Programación 3 5	QUI-010 Química y Sociedad 3 5	INF-134 Estructuras de Datos 1 2 3 5	INF-253 Lenguajes de Programación 1 2 3 5	INF-239 Bases de Datos 1 2 3 5	INF-236 Análisis y Diseño de Software 1 2 3 5	INF-225 Ingeniería de Software 1 2 3 5	INF-322 Diseño de Interfaces Usuaras 1 2 3 5	INF-302 Electivo Informática II 3 5											
MAT-021 Matemáticas I 5 8	MAT-022 Matemáticas II 2 3 5 7	MAT-023 Matemáticas III 7 9 4 7	MAT-024 Matemáticas IV 13 4 6	INF-245 Arquitectura y Organización de Computadores 1 2 3 5	INF-246 Sistemas Operativos 1 2 3 5	INF-256 Redes de Computadores 1 2 3 5	INF-343 Sistemas Distribuidos 1 2 3 5	INF-303 Electivo Informática III 3 5	INF-304 Electivo Informática IV 3 5										
FIS-100 Introducción a la Física 3 6	FIS-110 Física General I 2 3 5 8	FIS-130 Física General III 7 9 4 8	FIS-120 Física General II 7 9 4 8	FIS-140 Física General IV 14 20 4 8	INF-276 Ingeniería, Informática y Sociedad 1 2 3 5	ICN-270 Información y Matemáticas Financieras 1 2 3 5	INF-301 Electivo Informática I 3 5	INF-311 Electivo I 3 5	INF-313 Electivo III 3 5										
	IWI-101 Introducción a la Ingeniería 2 3	INF-152 Estructuras Discretas 1 2 3 5	INF-155 Informática Teórica 1 2 3 5	INF-280 Estadística Computacional 1 13 3 5	INF-221 Algoritmos y Complejidad 1 2 3 5	INF-285 Computación Científica 1 2 3 5	INF-295 Inteligencia Artificial 1 2 3 5	INF-312 Electivo II 3 5	INF-314 Electivo IV 3 5										
HRW-132 Humanística I 2 3	HRW-133 Humanística II 2 3	INF-260 Teoría de Sistemas 3 5	INF-170 Economía IA 1 2 3 5	INF-270 Organizaciones y Sistemas de Información 1 2 3 5	INF-292 Optimización 1 2 3 5	INF-293 Investigación de Operaciones 1 2 3 6	INF-266 Sistemas de Gestión 1 2 3 5	INF-360 Gestión de Proyectos de Informática 1 2 3 5	INF-228 Taller Desarrollo de Proyecto de Informática 1 2 3 5										
DEW-100 Educación Física I 1 2	DEW-101 Educación Física II 1 2	INF-1 Libre1/ Actividad co-curricular 1 2	INF-2 Libre2/ Actividad co-curricular 1 2	INF-3 Libre3/ Actividad co-curricular 1 2	INF-4 Libre4/ Actividad co-curricular 1 2	INF-5 Libre5/ Actividad co-curricular 1 2	INF-6 Libre6/ Actividad co-curricular 1 2	INF-7 Libre7/ Actividad co-curricular 1 2	INF-309 Trabajo de Título 1 1 2	INF-310 Trabajo de Título 2 1 2									
14 24	18 28	18 32	18 31	17 30	16 27	16 28	16 27	16 27	16 27	12 20									
BACHILLER EN CIENCIAS DE LA INGENIERÍA				LICENCIADO EN CIENCIAS DE LA INGENIERÍA															

DIAGNOSTICO BACHILLER

DIAGNOSTICO LICENCIATURA

Código asignatura

FIS-110

0

Número asignatura

Nombre asignatura

Física General I

Pre Requisito

(2)(3)(5)(8)

Créditos USM SCT

Matemáticas, Física y Química

Transversal y de Integración

Humanistas, Educación Física y Libres

Industrial y Comercial

Fundamentos de Informática

Sistemas de Información y de Decisión

Ingeniería de Software y Datos

Infraestructura TIC

Computación Aplicada en Ciencia e Ingeniería

Electivos Informática y Electivos

i

Al reverso perfil de egreso, inglés, prácticas, titulación, otros

Departamento de Informática

Figura 1: Malla Curricular Ingeniería Civil Informática.

Fuente: Departamento de Informática.

1.2. Situación actual

La desigualdad en las mallas curriculares puede causar problemas significativos para los estudiantes universitarios. Si las mallas curriculares no están equilibradas, es decir, si no se distribuyen adecuadamente los créditos y la carga académica a lo largo del tiempo, los estudiantes pueden enfrentar dificultades para completar su programa de estudios en el tiempo previsto, lo que puede afectar negativamente su desempeño académico y su bienestar emocional.

Por ejemplo, como sea el caso, se ha vuelto un problema común el que los estudiantes estén tomando ramos que chocan con otros y se alejan de la malla curricular definida, sumando o posponiendo otras asignaturas. Muchas veces, los estudiantes en sus últimos años toman ramos que dejaron pendientes para al final estar al día, pero la manera en que varios estudiantes se organizan respecto a su malla curricular tiende a ser iterativamente antes de la "toma de ramos".

Además, la desigualdad en las mallas curriculares también puede tener un impacto negativo en el bienestar emocional de los estudiantes. Los estudiantes pueden sentirse estresados,

ansiosos o abrumados por la carga académica excesiva, lo que puede afectar su salud mental y física. Esto puede conducir a la disminución de la motivación, la falta de interés por los estudios y la baja autoestima.

En este contexto, es crucial abordar la importancia de mantener mallas curriculares equilibradas. La falta de equilibrio puede generar tensiones académicas y emocionales, comprometiendo la calidad del aprendizaje. Además, las mallas desbalanceadas pueden conducir a trayectorias académicas no óptimas, afectando la eficiencia en la culminación de los estudios y, en última instancia, impactando las experiencias educativas.

1.2.1. Caso de ejemplo

Si tomamos por ejemplo la malla curricular del siguiente ejemplo, donde un estudiante que acercándose a su quinto semestre, que reprobó MAT-023 dos veces y no puede tomar MAT-024 ya se ve atrasado 1 año en el área matemática, bloqueándole también el paso al ramo de economía o IWN-170, esto debido a que ambas asignaturas tienen como prerequisite MAT-023.

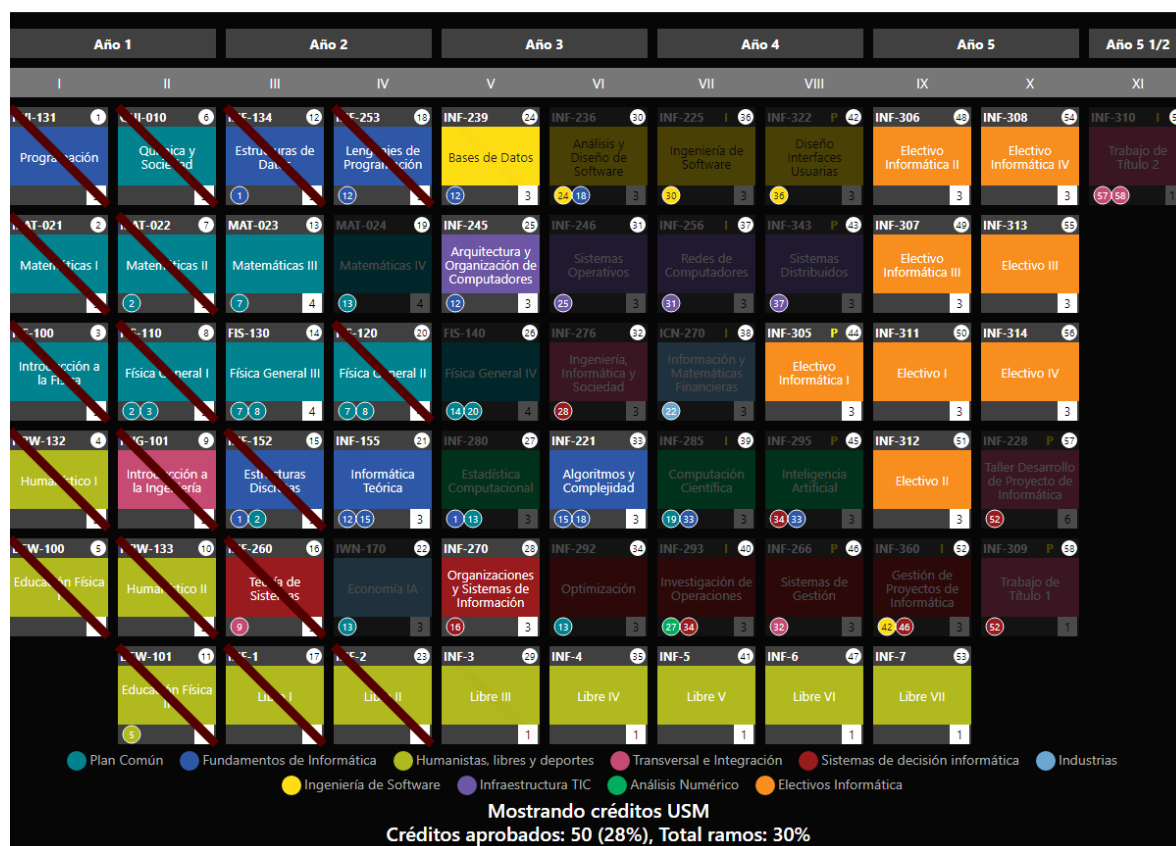


Figura 2: Avance de estudiante civil informática tercer año.

Fuente: Elaboración propia.

Una alternativa es que el alumno para el quinto semestre es, aparte de retomar Matemáticas III, tomar INF-155 e INF-221, este último siendo un ramo que debería tomarse el semestre siguiente, pero de esa forma el estudiante de ejemplo podría recuperar créditos que perdería al no poder cursar FIS-140 o MAT-023 o incluso INF-280. Pero el escenario mas esperado es que el estudiante a este punto se encuentre estresado al tener que retomar el mismo curso y quiere evitar dos ramos de la misma área y querer enfocarse en el reprobado, no tomando en cuenta así su carga académica en el quinto semestre.

1.2.2. Otros estudios

Con respecto a la situación de tiempo en la que un estudiante excede los semestres previstos, [Valenzuela20] en un estudio se observa que en la USM los estudiantes se demoran en promedio 17,54 semestres en obtener su título universitario. Es decir, su estadía en la universidad supera los 8 años.

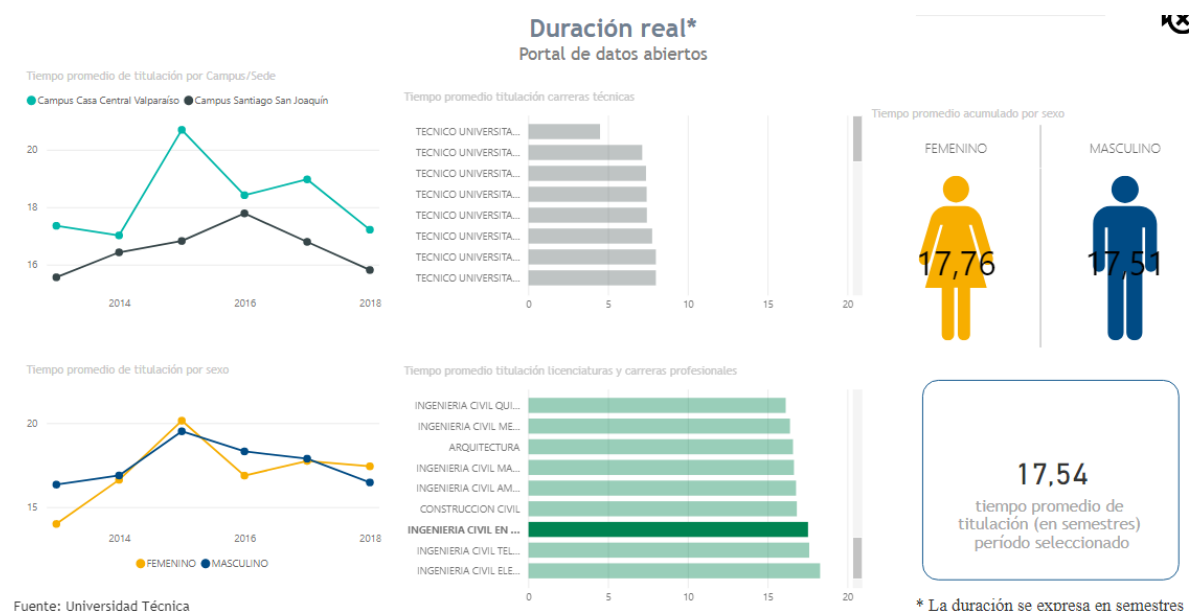


Figura 3: Duración real.

Fuente: Sitio web UTFSM.

Si bien existen varios factores que provocan la estadía dentro de la universidad por ejemplo dificultad de un ramo o falta de cupos, poder organizar los semestres a por venir teniendo en cuenta el avance actual del estudiante utilizando una herramienta puede ser bastante beneficioso.

Se puede resumir las aristas del problema con el siguiente árbol del problema para poder tener una mejor perspectiva:

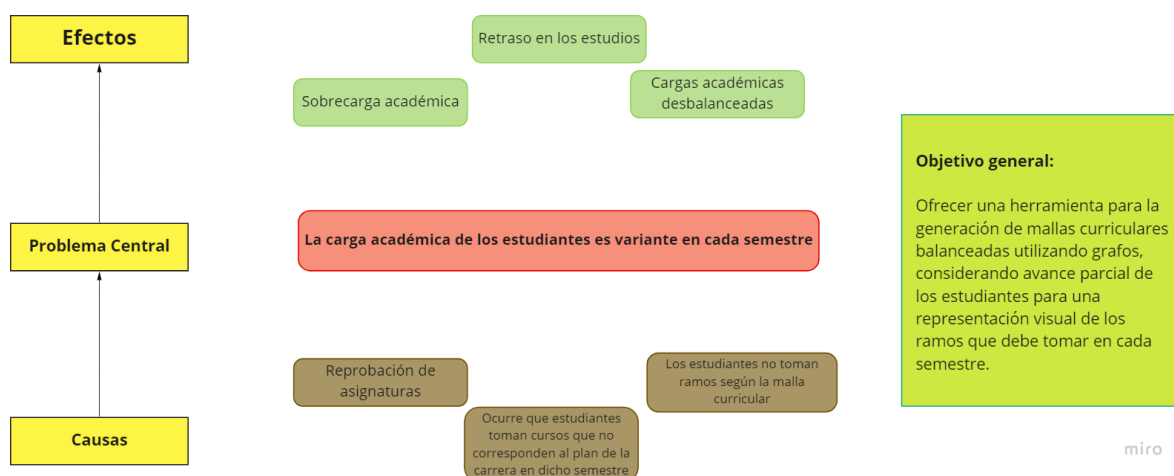


Figura 4: Árbol del problema.

Fuente: Elaboración propia.

Es por esto por lo que uno de los objetivos a los que se apunta es ayudar a los estudiantes a que tengan un mejor conocimiento y recomendación de que asignaturas deberían tomar en los semestres a por venir tomando en cuenta varios factores como lo son la carga académica y el avance de la carrera de cada usuario.

1.3. OBJETIVOS

1.3.1. OBJETIVO GENERAL

- Ofrecer una herramienta para la generación de mallas curriculares balanceadas utilizando grafos, considerando avance parcial de los estudiantes para una representación visual de los ramos que debe tomar en cada semestre.

1.3.2. OBJETIVOS ESPECIFICOS

- Poder tomar el output de un solver que resuelva el problema del balanceo de la carga académica y representar su resultado
- Representar su avance visualmente utilizando grafos conectando los ramos que son prerrequisitos de otros.

- Ordenar los grafos de tal forma que luzca como una malla curricular, donde el eje x represente los semestres y los ramos se posicionen en el eje y
- Validar el mismo procedimiento con distintas cargas y avances académicos.

CAPÍTULO 2: MARCO CONCEPTUAL

El problema de balanceo de carga académica o mejor conocido como BACP, es un problema que se ha planteado desde hace más de 40 años, consiste en asignar cursos a periodos de docencia que satisfagan los requisitos previos y equilibren la carga de los estudiantes en términos de créditos y número de cursos [Chiarandini12]. Otros autores [Rubio13] añade que el horizonte de planificación del BACP se divide en años académicos, y cada año académico se divide en periodos. En Chile es normal que un semestre de un año es un período lectivo en el que pueden tener lugar los cursos. El problema consiste en encontrar una asignación de cursos a periodos que satisfaga ciertos límites de carga y requisitos previos.

2.1 BACP en profundidad

Varios autores han propuesto modelamientos matemáticos para resolver este problema, La formulación original de BACP se propuso en [Castro01] y se incluyó como punto de referencia, llamado prob030, en CSPLib. El BACP se ha abordado utilizando programación con restricciones (PR) [Monette07], programación lineal entera (PLE) [Hnich02, Castro07], técnicas híbridas basadas en algoritmos genéticos y propagación de restricciones [Lambert05, Lambert06], programación entera (IP) y métodos híbridos de búsqueda local [Chiarandini12]

Una solución, es una asignación de cursos a periodos que pueda satisfacer las restricciones de requisitos previos (cursos que son prerrequisitos) mientras se equilibra la carga de trabajo de los periodos. La carga de trabajo de un período es la suma de los créditos de los cursos impartidos durante este período. Restricciones adicionales que limitan la número mínimo y máximo de cursos y créditos para cada período se introdujeron originalmente en [Castro01]. Sin embargo, estas restricciones adicionales no se consideran aquí ya que las soluciones equilibradas de las instancias en CSPLIB siempre las respetan. De hecho, en los límites en el número de créditos por período son naturalmente respetado por soluciones equilibradas, a menos que el problema sea inviable. Los límites en el número de asignaturas por periodo se respetan porque el número de créditos de cada curso no varía mucho y estos límites son bastante grande en las instancias de CSPLib [Monnette07]

2.2 Formulando BACP

En esta sección se presentará un estándar en cuanto a una instanciación de un BACP, [Chiarandini12] en su texto, presentó una formulación y se puede caracterizar por:

- Cursos, los cuales son el conjunto de asignaturas que se imparten en la malla curricular

- Periodos, que sería el horizonte de planeación, dividido en años académicos y cada año académico dividido en 2 términos, que serían los semestres, donde en cada semestre están asignados distintos cursos.
- Carga académica, cada curso tiene asignado un valor que representa los créditos o el esfuerzo que requiere para aprobarlo.
- Límite de carga, definidas por un mínimo y un máximo número de cursos que se pueden asignar en un semestre.
- Prerrequisitos, en su contenido, un prerrequisito son un conjunto de ramos que se requieren completar para poder tomar el curso en cuestión.
- Distribución equitativa de la carga, la distribución de créditos entre los periodos académicos debe de ser equilibrada. Idealmente, cada semestre debería tener el mismo número de créditos.

Si bien hay algunas variaciones del problema, como por ejemplo tener presente cursos que se imparten a la misma hora que otros, ocasionando topes de horario, no se considerarán para este caso ya que ese paradigma depende de la organización o casa de estudios y no es el mismo en todos los casos.

2.3 Representación gráfica de un BACP

Para efectos del usuario, poder tener una forma clara de ver su avance académico es fundamental para poder llevar su avance personal, tener conocimiento de que asignaturas cursar no solo es una forma de tener íntegramente el progreso en la universidad, si no que también estar consciente de como a medida que uno avanza va planificando su desarrollo a futuro.

Según Barnard (1921) "Un gráfico vale más que mil palabras". El uso de Organizadores gráficos es una estrategia recomendada para mejorar las habilidades de comprensión lectora que responde a los estudiantes de aula de hoy en día expuestos a imágenes visuales, a través de herramientas tecnológicas [Tayo18].

Los cursos de "Teoría de grafos y sus aplicaciones" forman parte hoy en día de los estudios de matemáticas, de investigación operativas, de matemática discreta, de diversas especialidades de ingeniería (organización de obras, edificación, eléctrica, telecomunicación...) y, por supuesto, están presentes en todos los estudios informáticos. [Alvarez13] Es por esto por lo que se ha propuesto el uso de nodos para representar los ramos de forma que se cumpla el entendimiento del flujo respecto a las materias a cursar.

2.3.1 Construcción de grafo

El gráfico que representa la asignación de los cursos a períodos determinados sigue un comportamiento particular, pudiéndolo comparar con un TSP señalan los autores [Rubio13]. Con el detalle de que los cursos C están alineados a los periodos P , en este ejemplo tendríamos $|C| \times |P|$ nodos.

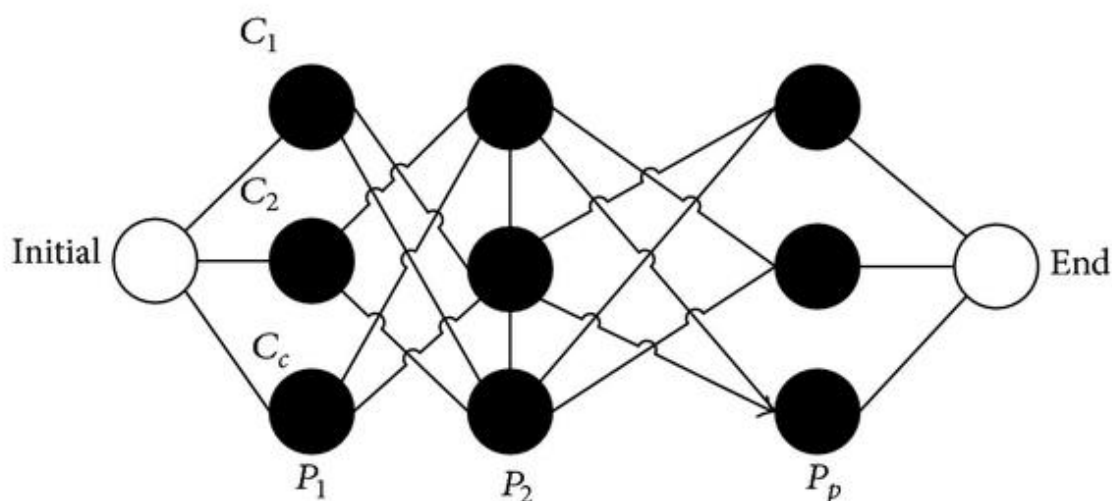


Figura 5: Construcción de grafo.

Fuente: Solving the Balanced Academic Curriculum Problem Using the ACO Metaheuristic. [Rubio13]

Originalmente esta representación es de carácter informativo y da a entender una ruta que debe cumplirse, para este trabajo tomará en gran consideración como está estructurado en la figura 5. Representar la malla curricular en forma de un grafo interconectado. Donde los cursos se reparten en columnas, las cuales serían los periodos o semestres académicos, los cursos que sean prerrequisitos de otras asignaturas estarán conectadas de modo que el usuario o estudiante pueda saber que el orden que requiere seguir para avanzar en la carrera.

2.3.2 Malla curricular como grafo

Con lo mencionado anteriormente, la idea ahora es poder representar el avance de cualquier carrera. Para este ejemplo tomaremos al de un estudiante de primer año, donde

su malla curricular debería ser el mismo que el de todos los demás estudiantes, su malla representada como un grado debería quedar algo como la siguiente imagen:

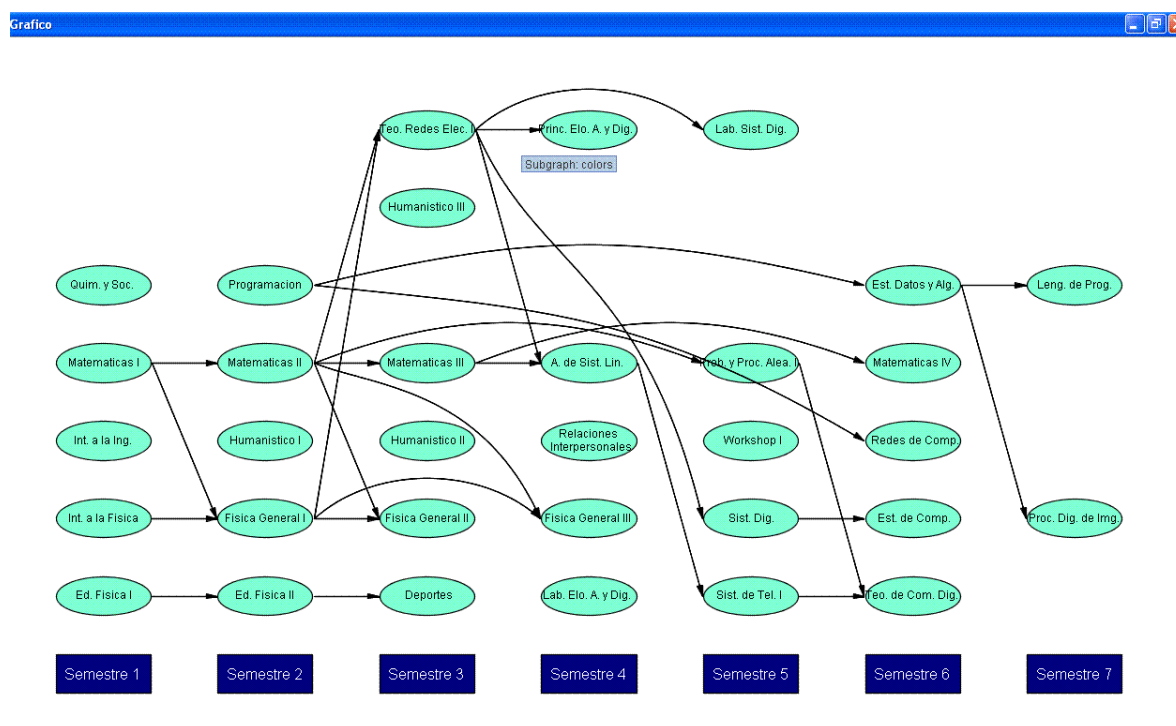


Figura 6: Malla curricular como grafo sentido horizontal

Fuente: Sitio web profesores electrónica UTFSM

En la figura 6 que representa la malla curricular de Ingeniería civil electrónica de la UTFSM, podemos ver una representación de grafos que sigue una idea de malla curricular tradicional, en donde se posicionan los cursos sobre los semestres en que se deberían impartir.

La idea principal es que esta malla curricular vaya cambiando según progrese cada estudiante, lo más común sea que para el segundo o tercer año existan cursos reprobados y por ende los grafos cambiaran de lugar, pero manteniendo la idea de que ciertas

asignaturas tienen que ir antes que otras ya que se debe cumplir los prerrequisitos de los cursos para poder tomarlos.

Existen también otras formas de representar una malla curricular con grafos, si bien la anterior figura nos muestra un avance horizontal existe de la misma manera una vertical como en el siguiente ejemplo:

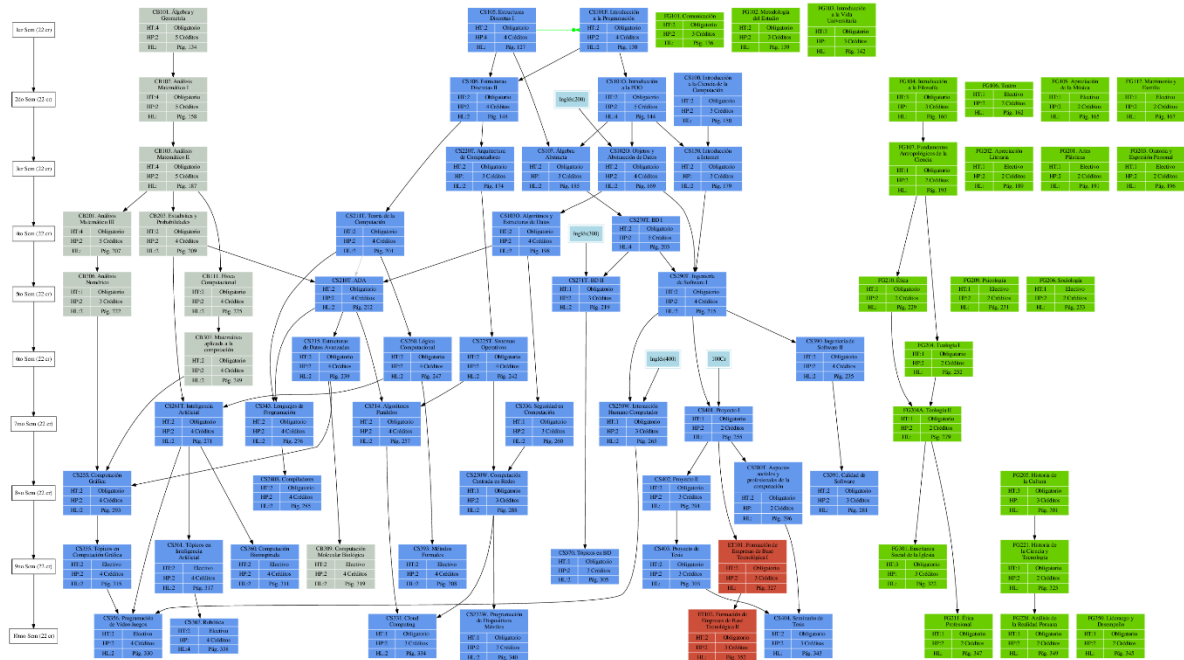


Figura 7: Malla curricular como grafo sentido vertical
Fuente: Sitio web Sociedad Peruana de la Computación

En la figura 7 podemos apreciar la malla curricular representada como grafo de la carrera Ciencias de la Computación de la Universidad Católica de San Pablo, Arequipa Perú, la cual avanza de arriba hacia abajo, en comparación a la figura 6, esta representación se aleja a una malla normal corriente haciendo que se vuelva un poco engorroso para un usuario común. Es por esto por lo que hay que poner esfuerzo en que la visualización sea lo mas cercana a una malla curricular posible y que sea entendible por el usuario.

CAPÍTULO 3: PROPUESTA DE SOLUCION

3.1 Enfoque de propuesta

Este capítulo propone un enfoque integral que utiliza dos herramientas clave para abordar el problema de las mallas curriculares desbalanceadas: la representación gráfica mediante Graphviz y la resolución de un modelo de programación lineal específico.

3.1.1 Representación Gráfica con Graphviz

La representación visual de las mallas curriculares desempeña un papel fundamental en nuestra propuesta. Utilizando las capacidades de la librería Graphviz en Python, se busca proporcionar una representación gráfica clara y efectiva de las asignaturas. Cada nodo en el grafo representa una asignatura, y las aristas dirigidas indican relaciones de prerrequisito entre ellas. Esta visualización le permitirá a estudiantes y planificadores académicos comprender de manera intuitiva la estructura de las mallas curriculares y las conexiones entre asignaturas.

En el ámbito de la visualización de datos y estructuras gráficas, Graphviz se erige como una herramienta poderosa y versátil. Graphviz proporciona un conjunto de herramientas para la creación y representación visual de grafos y redes complejas. Esta librería, implementada en varios lenguajes de programación, permite a los usuarios generar de manera eficiente diagramas y visualizaciones que capturan las relaciones y conexiones inherentes en conjuntos de datos.

Graphviz se destaca por su simplicidad de uso y flexibilidad. A través de la definición de nodos, aristas y propiedades específicas, podemos modelar y representar gráficamente estructuras complejas. La capacidad de ajustar la apariencia visual y la disposición de los elementos del grafo ofrece un control preciso sobre la presentación de la información, haciéndolo particularmente valioso en contextos como la representación de mallas curriculares y relaciones de prerrequisitos.

En este contexto, Graphviz se convierte en una herramienta esencial en nuestra propuesta, permitiéndonos visualizar de manera clara y efectiva las complejidades de las mallas curriculares y destacar las conexiones esenciales entre asignaturas.

3.1.2 Optimización con Programación Lineal

Para garantizar un equilibrio eficiente en las mallas curriculares, proponemos la implementación de un modelo de programación lineal. Este modelo considerará la distribución óptima de asignaturas a lo largo de los semestres, teniendo en cuenta restricciones específicas basadas en la tesis de [Vergara]. Cada asignatura se asignará a un

semestre de manera que se optimice la carga académica, mitigando desequilibrios y maximizando la eficiencia en el progreso académico

Optaremos por una implementación en Python que lea un archivo de texto con información detallada sobre asignaturas, incluyendo sus siglas, créditos y los ramos predecesores, para las asignaturas que tengan. A través de funciones dinámicas, generaremos líneas de restricciones que se ajustarán a las especificidades de cada conjunto de datos. Estas restricciones serán posteriormente utilizadas en la resolución de un modelo de programación lineal con un solver, se utilizará LPSolve IDE esta propuesta. Del conjunto de soluciones obtenidas mediante LPSolve, extraeremos los valores de las variables que definen en qué semestre se asigna cada asignatura. Estos valores serán fundamentales para la generación de la malla curricular balanceada.

Se puede representar esta propuesta de solución con el siguiente diagrama:

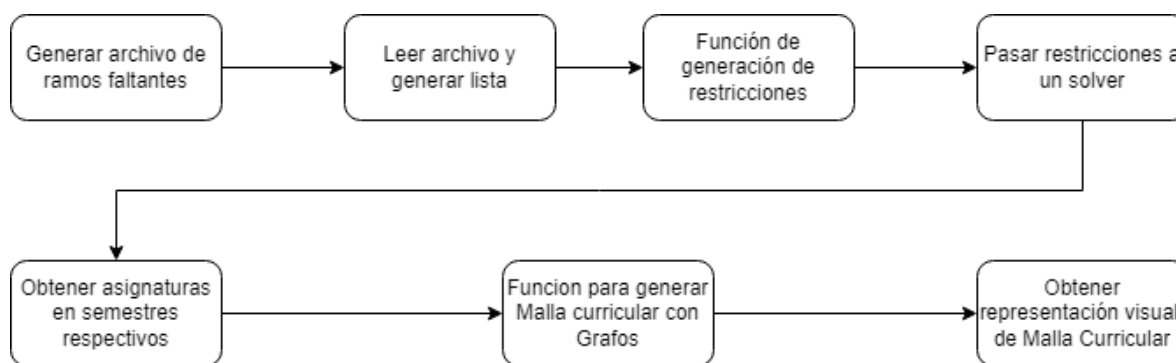


Figura 8: Flujo de propuesta de solución.

Fuente: elaboración Propia

Este flujo describe el proceso para generar y visualizar una malla curricular, tomando en cuenta ramos pendientes y ciertas restricciones, y usando técnicas de solución y representación gráfica para facilitar la planificación y visualización.

1. En este primer paso, se crea un archivo que contiene una lista de "ramos" (asignaturas o cursos) que aún no han sido completados o que están pendientes.
2. Una vez creado el archivo, se lee y procesa su contenido para generar una lista estructurada de estos ramos faltantes.

3. Con la lista de ramos faltantes, se pasa una función que genera restricciones. Estas restricciones podrían ser requisitos previos para ciertos ramos, límites de créditos por semestre, entre otros.
4. Las restricciones generadas son enviadas a un "solver", un programa o función diseñada para encontrar soluciones viables a problemas complejos, teniendo en cuenta las restricciones establecidas.
5. Después de que el solver encuentra una solución, se obtiene una lista o estructura de las asignaturas organizadas según los semestres en los que deben ser cursadas.
6. Con la organización de asignaturas por semestre, se pasa a una función que genera una representación gráfica, utilizando grafos, de la malla curricular. Esto ayuda a visualizar las relaciones entre las distintas asignaturas, como requisitos previos o secuencias recomendadas.
7. Finalmente, se obtiene una representación visual completa de la malla curricular basada en los grafos generados.

3.2. Lectura de archivo

Teniendo el documento de texto con los ramos correspondientes, se debe de leer para estructurar los datos de la siguiente manera, tomando como ejemplo la malla curricular de ingeniería civil informativa de la UTFSM:

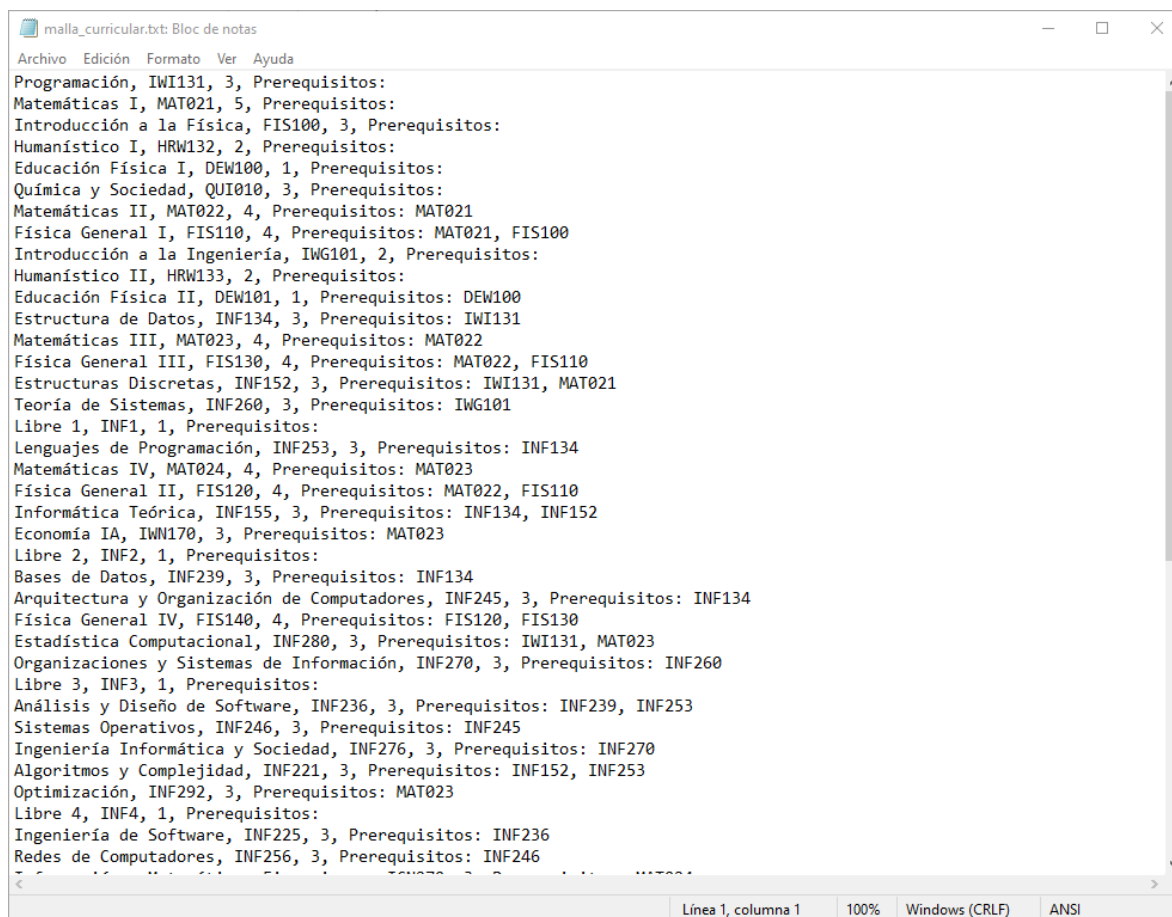


Figura 9: Estructura Malla curricular en documento de texto.

Fuente: elaboración Propia

Se puede notar como el documento enlista las asignaturas estructuradas de la siguiente manera:

- Nombre de la asignatura
- Sigla
- Créditos
- Prerrequisitos

Cabe destacar que siempre van los prerrequisitos, si una asignatura no tiene prerrequisitos se deja vacía, mientras que los que, si tienen, se añaden las siglas.

```
def leer_archivo(nombre_archivo):
    asignaturas = []

    with open(nombre_archivo, "r") as archivo:
        lineas = archivo.readlines()

        for linea in lineas:
            linea = linea.strip()
            partes = linea.split(",")

            nombre = partes[0].strip()
            sigla = partes[1].strip()
            creditos = int(partes[2].strip())

            prerequisitos = []
            if "Prerequisitos:" in linea:
                index = linea.index("Prerequisitos:")
                prerequisitos_linea = linea[index + len("Prerequisitos:"):].strip()
                if prerequisitos_linea:
                    prerequisitos = [p.strip() for p in prerequisitos_linea.split(",")]

            asignatura = [nombre, sigla, creditos, prerequisitos]
            asignaturas.append(asignatura)

    return asignaturas

asignaturas = leer_archivo("malla_curricular.txt")

restricciones = []

#Funcion para agregar restriccion a la lista de restricciones
def agregar_restriccion(restriccion):
    restricciones.append(restriccion)
```

Figura 10: función Leer archivo y agregar restricción.

Fuente: Elaboración Propia

Se define una lista restricciones y una función para agregar cada línea, o restricción, la cual se usarán para definir el modelo posteriormente.

3.3. Modelo formulado de Programación Lineal para BACP

Los parámetros del modelo formulado por [Vergara] indican lo siguiente:

- N = Cantidad de asignaturas

- K = Cantidad de semestres
- i = Índice para asignaturas
- j = Índice para semestres
- C_i = Número de créditos de la asignatura i ; $\forall i = 1, \dots, N$
- F = Carga académica semestral máxima permitida
- D = Carga académica semestral mínima permitida
- A = Cantidad máxima de asignaturas por semestre

Mientras que las variables de decisión se expresan de la siguiente manera:

Sean x_{ij} variables de decisión $\forall i = 1, \dots, N \forall j = 1, \dots, K$

Donde x_{ij} toma los valores siguientes:

- 1: Si el ramo i es asignado al semestre j
- 0: Si el ramo i no es asignado al semestre j

Sea y_j = Carga académica del semestre j ; $\forall j = 1, \dots, K$

Sea y carga académica semestral máxima

Función Objetivo

- $MIN y$

3.4 Restricciones del Modelo

En esta sección aparte de representar las restricciones matemáticas, estas vendrán acompañadas de su interpretación en Python para generarlas.

3.4.1 Restricción por carga académica

La carga académica del semestre j está definida por:

$$y_j = \sum_{i=1}^N C_i x_{ij} \quad \forall j = 1, \dots, K$$

```
def generar_restriccion_carga_academica(asignaturas, num_semestres):
    restricciones = []
    for j in range(num_semestres):
        restriccion_j = ""
        for i, asignatura in enumerate(asignaturas):
            sigla = asignatura[1]
            creditos = asignatura[2]
            restriccion_j += str(creditos)+"*" + "x" + sigla + str(j+1) + " + "
        # Borrar ultimo " + " y agregar el =
        restriccion_j = restriccion_j[:-3] + " = y" + str(j+1)
        restricciones.append(restriccion_j)

        #Agregar restriccion a la lista
        agregar_restriccion(restriccion_j)
        print(restriccion_j)

    return restricciones
```

Figura 11: Restricción carga académica.

Fuente: Elaboración Propia

Es la restricción que representa la carga académica semestral que se obtiene de todas las asignaturas cursadas en el semestre j .

3.4.2 Restricción por carga semestral máxima

La carga académica semestral máxima está definida por:

$$y_j \leq y \quad \forall j = 1, \dots, K$$

```
def restriccion_carga_semestral_max(num_semestres):
    restricciones = []
    for j in range(num_semestres):
        restriccion_j = "y" + str(j+1) + " <= " + "Y"
        restricciones.append(restriccion_j)

        #Agregar restriccion a la lista
        agregar_restriccion(restriccion_j)

    return restricciones
```

Figura 12: Restricción carga semestral máxima.

Fuente: Elaboración Propia

Esta restricción representa la mayor carga semestral del conjunto de las cargas de todos los semestres.

3.4.3 Restricción cursar en algún semestre

Toda asignatura debe ser cursada en algún semestre:

$$\sum_{j=1}^N x_{ij} = 1 \forall i = 1, \dots, N$$

```
def restriccion_cursar_semestre(asignaturas, num_semestres):
    for asignatura in asignaturas:
        sigla = asignatura[1]
        restriccion_i = ""
        for j in range(num_semestres):
            if j == num_semestres - 1:
                restriccion_i += "x" + sigla + str(j+1) + " = 1"
            else:
                restriccion_i += "x" + sigla + str(j+1) + " + "

        #Agregar restriccion a la lista
        agregar_restriccion(restriccion_i)
```

Figura 13: Restricción cursar semestre.

Fuente: Elaboración Propia

Esta restricción asegura que toda asignatura, va a cursarse en alguno de los semestres disponibles.

3.4.4 Restricción carga semestral máxima

La carga académica del semestre j debe ser menor o igual que el máximo permitido:

$$y_j \leq F \quad \forall j = 1, \dots, K$$

```
def generar_restriccion_carga_maxima(num_semestres, F):
    restricciones = []
    for j in range(num_semestres):
        restriccion_j = "y" + str(j+1) + " <= " + str(F)
        restricciones.append(restriccion_j)

        #Agregar restriccion a la lista
        agregar_restriccion(restriccion_j)

    return restricciones
```

Figura 14: Restricción carga máxima.

Fuente: Elaboración Propia

Esta restricción define que la carga del semestre j no podrá exceder del valor F, de manera de evitar sobre exigir al estudiante.

3.4.5 Restricción carga semestral mínima

La carga académica del semestre j debe ser mayor o igual que el máximo permitido:

$$y_j \geq D \quad \forall j = 1, \dots, K$$

```
def generar_restriccion_carga_minima(num_semestres, D):
    restricciones = []
    for j in range(num_semestres):
        restriccion_j = "y" + str(j+1) + " >= " + str(D)
        restricciones.append(restriccion_j)

        #Agregar restriccion a la lista
        agregar_restriccion(restriccion_j)

    return restricciones
```

Figura 15: Restricción carga mínima.

Fuente: Elaboración Propia

Esta restricción define que la carga del semestre j deberá de ser mayor que el valor D, el opuesto del máximo, que busca evitar el retraso o semestres demasiado livianos.

3.4.6 Restricción cantidad máxima de asignaturas por semestre

La cantidad de asignaturas del semestre j debe ser menor o igual que el máximo permitido:

$$\sum_{j=1}^K x_{ij} \leq A \quad \forall j = 1, \dots, K$$

```
def generar_restriccion_max_asignaturas(asignaturas, num_semestres, max_asignaturas):
    restricciones = []
    for j in range(num_semestres):
        restriccion_j = ""
        for asignatura in asignaturas:
            sigla = asignatura[1]
            restriccion_j += "x" + sigla + str(j+1) + " + "
        restriccion_j += "A <= " + str(max_asignaturas)
        restricciones.append(restriccion_j)

        #Agregar restriccion a la lista
        agregar_restriccion(restriccion_j)
    return restricciones
```

Figura 16: Restricción máximo de asignaturas.

Fuente: Elaboración Propia

Esta restricción busca no sobrepasar la cantidad de asignaturas inscritas en un semestre. Que también busca evitar sobre exigir los tiempos del estudiante.

3.4.7 Restricción asignaturas con prerrequisitos

Algunas asignaturas tienen como prerrequisitos otras asignaturas:

Si la asignatura b tiene como prerrequisito la asignatura a, entonces

$$x_{bj} \leq \sum_{r=1}^{j-1} x_{ar} \quad \forall j = 2, \dots, K$$

```
def generar_restricciones_prerrequisitos(sigla_asignatura, asignaturas, num_semestres):
    restricciones = []
    for asignatura in asignaturas:
        sigla = asignatura[1]
        prerrequisitos = asignatura[3]
        if sigla == sigla_asignatura and prerrequisitos:
            for prerrequisito in prerrequisitos:
                for semestre in range(2, num_semestres + 1): # Comenzamos desde el segundo semestre
                    asignatura_semestre = "x" + sigla + str(semestre)
                    restriccion = f"{asignatura_semestre} <= "
                    for i in range(1, semestre):
                        prereq_semestre = "x" + prerrequisito + str(i)
                        restriccion += prereq_semestre + " + "
                    restriccion = restriccion[:-3] # Eliminar el último "+" y espacios
                    restricciones.append(restriccion)

                #Agregar restriccion a la lista
                agregar_restriccion(restriccion)
    return restricciones
```

```
def generar_restricciones_prerequisitos_Lista(asignaturas, num_semestres):
    for asignatura in asignaturas:
        sigla = asignatura[1]
        generar_restricciones_prerequisitos(sigla, asignaturas, num_semestres)

    return restricciones
```

Figura 18 y 19: Restricción prerequisitos y añadir a lista restricciones.

Fuente: Elaboración Propia

Esta restricción establece que, si una asignatura tiene otra como prerequisito, la primera debe programarse en un semestre posterior al de la asignatura que es prerequisito. Primero se define la función de la restricción y posteriormente se recorre la lista de asignaturas y se hace la búsqueda para generar las restricciones.

3.5 implementación grafica con Graphviz

La implementación en Python utilizando la biblioteca Graphviz se centrará en representar una malla curricular basada en los ramos y sus prerequisitos. A continuación, detallo las partes clave para la generación del grafo:

3.5.1 Creación de la instancia del grafo

Para iniciar, se utiliza `graphviz.Digraph()` para crear un grafo dirigido. Este grafo será donde se añaden todos los nodos y aristas. Seleccione el motor 'neato' para la visualización mediante `dot.engine = 'neato'`, que es la herramienta que da soporte a la imagen, como se puede ver en la siguiente figura:

```
In [1]: import graphviz

#Crear Grafo
dot = graphviz.Digraph(comment='Example')

#Agregar Nodos
dot.node('db1', 'input A', {'color':'aquamarine','style':'filled'})
dot.node('db2', 'input B', {'color':'aquamarine','style':'filled'})
dot.node('db3', 'input C', {'color':'aquamarine','style':'filled'})

dot.node('B', 'transformation', shape='box')
dot.node('C', 'output', shape='cylinder', style='filled')

#Agregar Conexion
for n in ['db1','db2','db3']:
    dot.edge(n, 'B')
dot.edge('B','C')
```

In [2]: dot

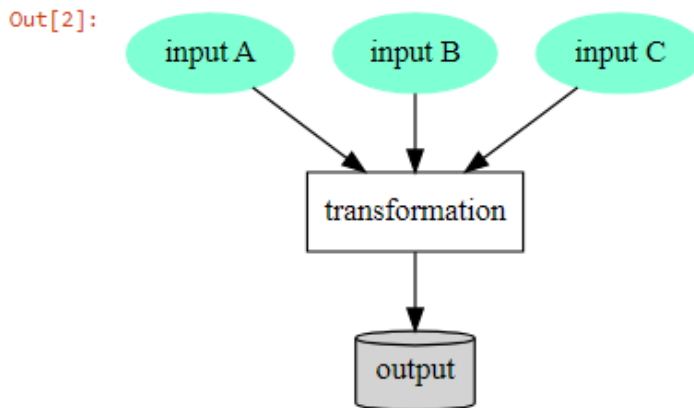


Figura 20: Ejemplo de creación de un grafo con Graphviz
Fuente: Elaboración Propia

Se puede apreciar en la Figura 20 un ejemplo de visualización, que define formas, colores y estilos en los nodos. Serán fundamentales para visualización deseada.

3.5.2 Creación de Nodos y Arcos

Se utiliza la función `dot.node()` para añadir el nodo, el primer elemento de la función equivale al nombre o identificador del nodo, el segundo elemento define lo que estará escrito adentro del nodo y posteriormente los siguientes elementos van relacionados con el diseño, estilo, posicionamiento, etc.

La función `dot.edge()` grafica la arista que une el nodo del primer elemento en la función con la segunda. Si se escribe `dot.edge('a','b')` sería lo mismo que `a -> b` ('a' apunta a 'b')

```
In [8]: #Testeando posiciones

dot2 = graphviz.Digraph(comment='Example')

#Crear Grafo
#Agregar Nodos
dot2.node('a', 'input A', {'pos': '-1,0!'})
dot2.node('b', 'input B', {'pos': '1,1!'})
dot2.node('c', 'input C', {'pos': '1,0!'})
dot2.node('d', 'input D', {'pos': '1,-1!'})

#Agregar Conexion
dot2.edge('a', 'b')
dot2.edge('a', 'c')
dot2.edge('a', 'd')
```

```
In [9]: dot2.engine = 'neato'
dot2
```

Out[9]:

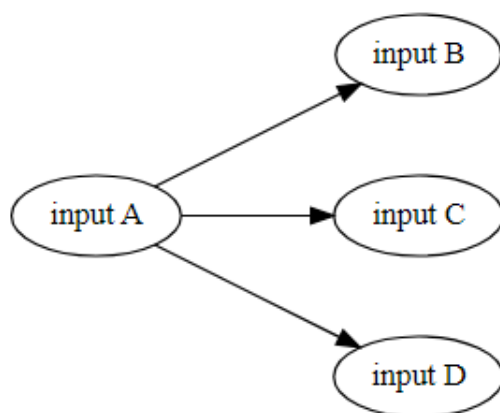


Figura 21: Ejemplo de uso función node y edge

Fuente: Elaboración Propia

Para este ejemplo se define el posicionamiento de cada nodo, dando una distancia del eje horizontal de 2 entre columnas, debido a que una distancia de 1, visualmente estaban muy cercas y no se apreciaban las conexiones.

3.5.3 Función para generar Malla Curricular

Sabiendo como crear nodos, unirlos y editarlos, tenemos lo necesario para crear funciones de ordenamiento para generar la Malla curricular que deseamos.

```
import graphviz

#Funciones

def OrdenarListaRamos(ramos):
    lista_ordenada = sorted(ramos, key=lambda lista : lista[2]) #Ordenar lista por semestre
    lista_final = []
    lista_sememstral = []
    contador = 0
    for i in lista_ordenada: #Crear lista que agrupe ramos por semestre
        if len(lista_sememstral) == 0: #Asumiendo que el primer elemento de la lista está en el semestre mas cercano
            lista_sememstral.append(i)
        elif i[2] == lista_ordenada[contador-1][2]: #Si el ramo es del mismo semestre que el anterior, se agrega a la misma lista
            lista_sememstral.append(i)
        else: # Se agrega la lista actual a la lista final antes de cambiar de semestre
            lista_final.append(lista_sememstral)
            lista_sememstral = []
            lista_sememstral.append(i) #Aqui empieza otro semestre
            contador += 1
        if contador == len(lista_ordenada): #Si llegamos al final guardamos la lista final agregandole el ultimo semestre
            lista_final.append(lista_sememstral)
    return lista_final

def Generar_Malla(lista_final):
    for semestre in lista_final:
        contador = 0
        #dot.node(id, texto, {'pos':'x,y!','shape':'box'})
        for ramo in semestre: # Definimos colores para distintas siglas
            if( "IWI" in ramo[0]):
                color = 'darkgoldenrod1'
            elif( "FIS" in ramo[0]):
                color = 'aquamarine'
            elif( "MAT" in ramo[0]):
                color = 'firebrick2'
            elif( "HRW" in ramo[0]):
                color = 'chartreuse2'
            elif( "DEW" in ramo[0]):
                color = 'burlywood1'
            elif( "INF" in ramo[0]):
                color = 'darkgoldenrod3'
            else:
                color = 'none'
            dot.node(ramo[0],ramo[0]+'\\n'+ramo[1], {
                'pos':str(int(ramo[2])+1.2*int(ramo[2]))+'!', 'shape':'box', 'fixedsize':'shape',
                'width':'1.6', 'height':'0.8', 'fillcolor':color, 'style':'filled',})
            contador+= 1

def Asignar_Arcos(lista_test):
    for ramo in lista_test:
        if len(ramo)>3: #Si ramo tiene prerequisitos, representarlos al grafo
            for i in ramo[3]: #Por cada ramo prerequisito
                dot.edge(i,ramo[0]) #Añadir conexion

#Crear Grafo
dot = graphviz.Digraph(comment='Example')

#Variable de representacion grafica
dot.engine = 'neato'
```

Figura 22,23 y 24: Función Ordenar Ramos, Generar Malla y Asignar Arcos

Fuente: Elaboración Propia

La Figura 22 muestra la función OrdenarListaRamos, la cual es esencial para organizar las asignaturas en columnas según sus respectivos semestres. La primera columna corresponde al semestre más cercano, mientras que la última representa el semestre final.

Por su parte, la Figura 23 presenta la función Generar_Malla. Tras recibir la lista previamente ordenada, esta función se encarga de asignar cada nodo al grafo. En este

ejemplo, los nodos tienen forma de caja y se les asigna colores específicos basados en las siglas del ramo, lo que permite diferenciar distintas áreas de estudio. Además, esta función define la posición exacta de cada nodo en el grafo.

Finalmente, la Figura 24 destaca cómo se establecen las aristas que representan los prerrequisitos. Estas aristas apuntan hacia los ramos que requieren de ciertas asignaturas previas, proporcionando así una visualización clara de las dependencias entre ramos. Este enfoque es crucial para informar al estudiante sobre la importancia de completar ciertos ramos antes que otros. De esta manera, queda patente que cualquier retraso en cursar los prerrequisitos podría implicar demoras en ramos subsecuentes.

3.5.4 Testing con lista de prueba

Teniendo las funciones anteriores, probamos con la siguiente lista:

```
lista_test = [{"DEW100", "Ed Fisica", "3", ["FIS100"]}, {"FIS100", "Introduccion Fisica", "2", []}, {"IWG101", "Introduccion Fisica", "3", []}, {"IWI131", "Programacion", "1", ["MAT021", "Matematicas 1", "1", ["DEW101", "Ed fisica1", "1", ["HRW132", "Humanista 1", "1", ["HRW132", "Humanista", "2", ["IWI131"]], ["MAT022", "Matematicas 2", "2", ["MAT021"]], ["HRW133", "Humanista 2", "2", ["MAT023", "Matematicas 3", "3", ["MAT022"]], ["MAT024", "Matematicas 4", "4", ["MAT023"]], ["INF253", "LP", "4", ["INF221", "Algoritm. Comple.", "4", ["IWG101"]], ["INF285", "Comp. Cientifica 3", "5", ["MAT024", "INF221"]], []}

ramos_ordenados = OrdenarListaRamos(lista_test)

Generar_Malla(ramos_ordenados)
Asignar_Arcos(lista_test)

dot
```

Figura 25: Lista de prueba

Fuente: Elaboración Propia

Donde la estructura de la lista es [sigla, nombre, semestre, lista prerrequisitos]

El output de ejecutar las funciones resulta en la siguiente imagen:

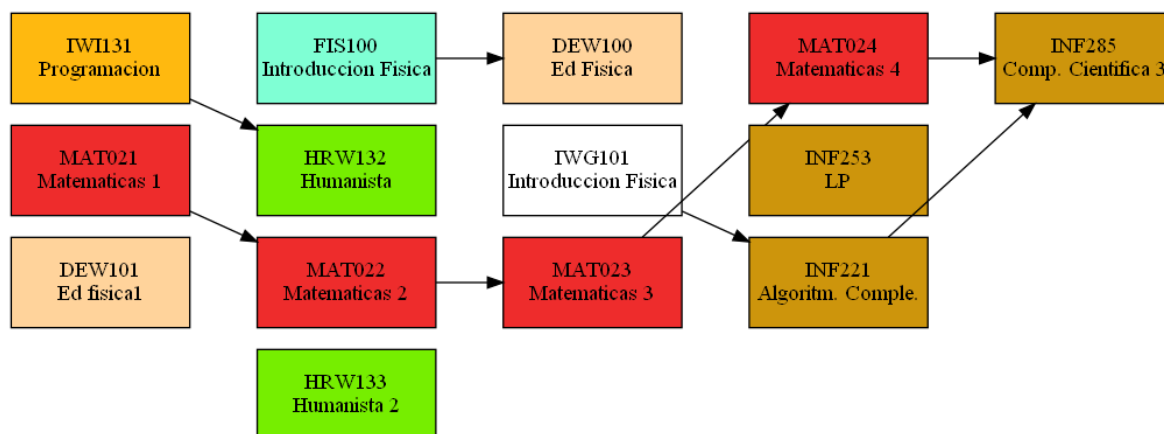


Figura 26: Malla generada a partir de lista de prueba

Fuente: Elaboración Propia

CAPÍTULO 4: VALIDACIÓN DE LA SOLUCIÓN

Para validar la propuesta de solución, en este capítulo se aplica las funciones sugeridas utilizando una malla curricular de ejemplo. Esta malla pasará por el proceso de ordenamiento, se aplicará el solver de balanceo y, finalmente, se graficará el resultado. Con esto, buscamos demostrar la validez de la solución propuesta.

4.1 Archivo utilizado

Se tendrá en mente un estudiante el cual su avance hasta el Sexto semestre a sido el siguiente:



Figura 27: Malla académica de prueba

Fuente: Elaboración Propia

Una vez que se ha recolectado y analizado la información sobre las asignaturas que el estudiante aún tiene pendientes, el siguiente paso es la creación de un documento de texto. Este documento debe ser estructurado de tal manera que su contenido pueda ser fácilmente interpretado y procesado por el código que hemos desarrollado. La estructura adecuada y ordenada del archivo es esencial para garantizar que el programa funcione de manera óptima y produzca los resultados esperados.

4.2 Generación de archivo de texto

Tomando como referencia la malla presentada en la figura 27, el formato del archivo de texto que se debe generar es el siguiente:

Tabla 1: Documento de texto de malla de ejemplo

Fuente: Elaboración Propia

Matemáticas IV, MAT024, 4, Prerequisitos:
 Física General II, FIS120, 4, Prerequisitos:
 Economía IA, IWN170, 3, Prerequisitos:
 Física General IV, FIS140, 4, Prerequisitos: FIS120
 Análisis y Diseño de Software, INF236, 3, Prerequisitos:
 Sistemas Operativos, INF246, 3, Prerequisitos:
 Ingeniería Informática y Sociedad, INF276, 3, Prerequisitos:
 Optimización, INF292, 3, Prerequisitos:
 Libre 4, INF4, 1, Prerequisitos:
 Ingeniería de Software, INF225, 3, Prerequisitos: INF236
 Redes de Computadores, INF256, 3, Prerequisitos: INF246
 Información y Matemáticas Financieras, ICN270, 3, Prerequisitos: MAT024
 Computación Científica, INF285, 3, Prerequisitos: MAT024
 Investigación de Operaciones, INF293, 3, Prerequisitos: INF292
 Diseño Interfaces Usuaris, INF322, 3, Prerequisitos: INF225
 Sistemas Distribuidos, INF343, 3, Prerequisitos: INF256
 Inteligencia Artificial, INF295, 3, Prerequisitos: INF292
 Sistemas de Gestión, INF266, 3, Prerequisitos:
 Libre 6, INF6, 1, Prerequisitos:
 Electivo Informática II, INF302, 3, Prerequisitos:
 Electivo Informática III, INF303, 3, Prerequisitos:
 Electivo I, INF311, 3, Prerequisitos:
 Electivo II, INF312, 3, Prerequisitos:
 Gestión de Proyectos de Informática, INF360, 3, Prerequisitos: INF322, INF266
 Libre 7, INF7, 1, Prerequisitos:
 Electivo Informática IV, INF304, 3, Prerequisitos:
 Electivo III, INF313, 3, Prerequisitos:
 Electivo IV, INF314, 3, Prerequisitos:
 Taller Desarrollo de Proyecto de Informática, INF228, 6, Prerequisitos: INF360
 Trabajo de Título 1, INF309, 1, Prerequisitos: INF360
 Trabajo de Título 2, INF310, 12, Prerequisitos: INF228, INF309

Una vez que hemos adecuadamente preparado y configurado el documento, el siguiente paso es proceder con su lectura. Esta acción nos permite extraer y organizar la lista de asignaturas pendientes. Una vez obtenida esta lista, es esencial aplicar las funciones diseñadas para generar las restricciones necesarias para cada asignatura.

Es fundamental mencionar que, para este proceso, hemos establecido ciertos parámetros que guiarán el sistema. Estos parámetros son:

- Número máximo de asignaturas por semestre: 7.
- Cantidad máxima de créditos semestrales: 30.
- Cantidad mínima de créditos semestrales: 5.
- Total, de semestres disponibles para el estudiante: 6.

4.3 Generación de modelo y uso de solver

Con todas las restricciones generadas y teniendo en cuenta los parámetros previamente mencionados, alimentamos estos datos a la variable “restricciones”. Posteriormente, esta información es enviada directamente al solver para procesarla y obtener los resultados deseados

Fuente: Elaboración Propia

```

[PSolve IDE - 5.5.211 - C:\Users\jarsa\OneDrive\Documents\T2\Memorial\ValidacionTest.bj
File Edit Search Action View Options Help
Source Matrix Options Result

98 xINF2921 + xINF2922 + xINF2923 + xINF2924 + xINF2925 + xINF2926 = 1;
99 xINF41 + xINF42 + xINF43 + xINF44 + xINF45 + xINF46 = 1;
100 xINF2251 + xINF2252 + xINF2253 + xINF2254 + xINF2255 + xINF2256 = 1;
101 xINF2561 + xINF2562 + xINF2563 + xINF2564 + xINF2565 + xINF2566 = 1;
102 xICN2701 + xICN2702 + xICN2703 + xICN2704 + xICN2705 + xICN2706 = 1;
103 xINF2851 + xINF2852 + xINF2853 + xINF2854 + xINF2855 + xINF2856 = 1;
104 xINF2931 + xINF2932 + xINF2933 + xINF2934 + xINF2935 + xINF2936 = 1;
105 xINF3221 + xINF3222 + xINF3223 + xINF3224 + xINF3225 + xINF3226 = 1;
106 xINF3431 + xINF3432 + xINF3433 + xINF3434 + xINF3435 + xINF3436 = 1;
107 xINF2951 + xINF2952 + xINF2953 + xINF2954 + xINF2955 + xINF2956 = 1;
108 xINF2461 + xINF2462 + xINF2463 + xINF2464 + xINF2465 + xINF2466 = 1;
109 xINF61 + xINF62 + xINF63 + xINF64 + xINF65 + xINF66 = 1;
110 xINF3021 + xINF3022 + xINF3023 + xINF3024 + xINF3025 + xINF3026 = 1;
111 xINF3031 + xINF3032 + xINF3033 + xINF3034 + xINF3035 + xINF3036 = 1;
112 xINF3111 + xINF3112 + xINF3113 + xINF3114 + xINF3115 + xINF3116 = 1;
113 xINF3121 + xINF3122 + xINF3123 + xINF3124 + xINF3125 + xINF3126 = 1;
114 xINF3401 + xINF3402 + xINF3403 + xINF3404 + xINF3405 + xINF3406 = 1;
115 xINF71 + xINF72 + xINF73 + xINF74 + xINF75 + xINF76 = 1;
116 xINF3041 + xINF3042 + xINF3043 + xINF3044 + xINF3045 + xINF3046 = 1;
117 xINF3131 + xINF3132 + xINF3133 + xINF3134 + xINF3135 + xINF3136 = 1;
118 xINF3141 + xINF3142 + xINF3143 + xINF3144 + xINF3145 + xINF3146 = 1;
119 xINF2281 + xINF2282 + xINF2283 + xINF2284 + xINF2285 + xINF2286 = 1;
120 xINF3051 + xINF3052 + xINF3053 + xINF3054 + xINF3055 + xINF3056 = 1;
121 xINF3101 + xINF3102 + xINF3103 + xINF3104 + xINF3105 + xINF3106 = 1;
122 xMAT0241 + xMAT0242 + xMAT0243 + xMAT0244 + xMAT0245 + xMAT0246 + xFIS1201 + xFIS1202 + xFIS1203 + xFIS1204 + xFIS1205 + xFIS1206 + xIWN1701 + xIWN1702 + xIWN1703 + xIWN1704 + xIWN1705 + xIWN1706 + xINF2361 + xINF2362 + xINF2363 + xINF2364 + xINF2365 + xINF2366 + xINF2761 + xINF2762 + xINF2763 + xINF2764 + xINF2765 + xINF2766 + xINF2921 + xINF2922 + xINF2923 + xINF2924 + xINF2925 + xINF2926 + xICN2701 + xICN2702 + xICN2703 + xICN2704 + xICN2705 + xICN2706 + xINF2851 + xINF2852 + xINF2853 + xINF2854 + xINF2855 + xINF2856 + xINF2931 + xINF2932 + xINF2933 + xINF2934 + xINF2935 + xINF2936 + xINF3221 + xINF3222 + xINF3223 + xINF3224 + xINF3225 + xINF3226 + xINF3431 + xINF3432 + xINF3433 + xINF3434 + xINF3435 + xINF3436 + xINF2951 + xINF2952 + xINF2953 + xINF2954 + xINF2955 + xINF2956 + xINF2461 + xINF2462 + xINF2463 + xINF2464 + xINF2465 + xINF2466 + xINF61 + xINF62 + xINF63 + xINF64 + xINF65 + xINF66 = 1;
123 xMAT0241 + xMAT0242 + xMAT0243 + xMAT0244 + xMAT0245 + xMAT0246 + xFIS1201 + xFIS1202 + xFIS1203 + xFIS1204 + xFIS1205 + xFIS1206 + xIWN1701 + xIWN1702 + xIWN1703 + xIWN1704 + xIWN1705 + xIWN1706 + xINF2361 + xINF2362 + xINF2363 + xINF2364 + xINF2365 + xINF2366 + xINF2761 + xINF2762 + xINF2763 + xINF2764 + xINF2765 + xINF2766 + xINF2921 + xINF2922 + xINF2923 + xINF2924 + xINF2925 + xINF2926 + xICN2701 + xICN2702 + xICN2703 + xICN2704 + xICN2705 + xICN2706 + xINF2851 + xINF2852 + xINF2853 + xINF2854 + xINF2855 + xINF2856 + xINF2931 + xINF2932 + xINF2933 + xINF2934 + xINF2935 + xINF2936 + xINF3221 + xINF3222 + xINF3223 + xINF3224 + xINF3225 + xINF3226 + xINF3431 + xINF3432 + xINF3433 + xINF3434 + xINF3435 + xINF3436 + xINF2951 + xINF2952 + xINF2953 + xINF2954 + xINF2955 + xINF2956 + xINF2461 + xINF2462 + xINF2463 + xINF2464 + xINF2465 + xINF2466 + xINF61 + xINF62 + xINF63 + xINF64 + xINF65 + xINF66 = 1;
124 xMAT0242 + xFIS1202 + xIWN1702 + xINF2362 + xINF2762 + xINF2922 + xINF42 + xINF2252 + xINF2562 + xICN2702 + xINF2852 + xINF2932 + xINF3222 + xINF3432 + xINF2952 + xINF2462 + xINF62 + xINF3022 + xINF3032 + xINF3112 + xINF3122 + xINF3402 + xINF72 + xINF3042 + xINF3132 + xINF3142 + xINF2282 + xINF3052 + xINF3102 + xMAT0244 + xMAT0245 + xMAT0246 + xFIS1204 + xFIS1205 + xFIS1206 + xIWN1704 + xIWN1705 + xIWN1706 + xINF2364 + xINF2764 + xINF2924 + xINF44 + xINF2254 + xINF2564 + xICN2704 + xINF2854 + xINF2934 + xINF3224 + xINF3434 + xINF2954 + xINF2464 + xINF64 + xINF3024 + xINF3034 + xINF3114 + xINF3124 + xINF3404 + xINF74 + xINF3044 + xINF3134 + xINF3144 + xINF2284 + xINF3054 + xINF3104 + xMAT0245 + xMAT0246 + xFIS1205 + xFIS1405 + xFIS1406 + xIWN1705 + xIWN1706 + xINF2365 + xINF2765 + xINF2925 + xINF45 + xINF2255 + xINF2565 + xICN2705 + xINF2855 + xINF2935 + xINF3225 + xINF3435 + xINF2955 + xINF2465 + xINF65 + xINF3025 + xINF3035 + xINF3115 + xINF3125 + xINF3405 + xINF75 + xINF3045 + xINF3135 + xINF3145 + xINF2285 + xINF3055 + xINF3105 + xMAT0246 + xFIS1206 + xFIS1406 + xIWN1706 + xINF2366 + xINF2766 + xINF2926 + xINF46 + xINF2256 + xINF2566 + xICN2706 + xINF2856 + xINF2936 + xINF3226 + xINF3436 + xINF2956 + xINF2466 + xINF66 + xINF3026 + xINF3036 + xINF3116 + xINF3126 + xINF3406 + xINF76 + xINF3046 + xINF3136 + xINF3146 = 1;
125 xMAT0243 + xFIS1203 + xIWN1703 + xINF2363 + xINF2763 + xINF2923 + xINF43 + xINF2253 + xINF2563 + xICN2703 + xINF2853 + xINF2933 + xINF3223 + xINF3433 + xINF2953 + xINF2463 + xINF63 + xINF3023 + xINF3033 + xINF3113 + xINF3123 + xINF3403 + xINF73 + xINF3043 + xINF3133 + xINF3143 + xINF2283 + xINF3053 + xINF3103 + xMAT0244 + xMAT0245 + xMAT0246 + xFIS1204 + xFIS1205 + xFIS1206 + xIWN1704 + xIWN1705 + xIWN1706 + xINF2364 + xINF2764 + xINF2924 + xINF44 + xINF2254 + xINF2564 + xICN2704 + xINF2854 + xINF2934 + xINF3224 + xINF3434 + xINF2954 + xINF2464 + xINF64 + xINF3024 + xINF3034 + xINF3114 + xINF3124 + xINF3404 + xINF74 + xINF3044 + xINF3134 + xINF3144 + xINF2284 + xINF3054 + xINF3104 + xMAT0245 + xMAT0246 + xFIS1205 + xFIS1405 + xFIS1406 + xIWN1705 + xIWN1706 + xINF2365 + xINF2765 + xINF2925 + xINF45 + xINF2255 + xINF2565 + xICN2705 + xINF2855 + xINF2935 + xINF3225 + xINF3435 + xINF2955 + xINF2465 + xINF65 + xINF3025 + xINF3035 + xINF3115 + xINF3125 + xINF3405 + xINF75 + xINF3045 + xINF3135 + xINF3145 + xINF2285 + xINF3055 + xINF3105 + xMAT0246 + xFIS1206 + xFIS1406 + xIWN1706 + xINF2366 + xINF2766 + xINF2926 + xINF46 + xINF2256 + xINF2566 + xICN2706 + xINF2856 + xINF2936 + xINF3226 + xINF3436 + xINF2956 + xINF2466 + xINF66 + xINF3026 + xINF3036 + xINF3116 + xINF3126 + xINF3406 + xINF76 + xINF3046 + xINF3136 + xINF3146 = 1;
126 xMAT0244 + xFIS1204 + xIWN1704 + xINF2364 + xINF2764 + xINF2924 + xINF44 + xINF2254 + xINF2564 + xICN2704 + xINF2854 + xINF2934 + xINF3224 + xINF3434 + xINF2954 + xINF2464 + xINF64 + xINF3024 + xINF3034 + xINF3114 + xINF3124 + xINF3404 + xINF74 + xINF3044 + xINF3134 + xINF3144 + xINF2284 + xINF3054 + xINF3104 + xMAT0245 + xMAT0246 + xFIS1205 + xFIS1405 + xFIS1406 + xIWN1705 + xIWN1706 + xINF2365 + xINF2765 + xINF2925 + xINF45 + xINF2255 + xINF2565 + xICN2705 + xINF2855 + xINF2935 + xINF3225 + xINF3435 + xINF2955 + xINF2465 + xINF65 + xINF3025 + xINF3035 + xINF3115 + xINF3125 + xINF3405 + xINF75 + xINF3045 + xINF3135 + xINF3145 + xINF2285 + xINF3055 + xINF3105 + xMAT0246 + xFIS1206 + xFIS1406 + xIWN1706 + xINF2366 + xINF2766 + xINF2926 + xINF46 + xINF2256 + xINF2566 + xICN2706 + xINF2856 + xINF2936 + xINF3226 + xINF3436 + xINF2956 + xINF2466 + xINF66 + xINF3026 + xINF3036 + xINF3116 + xINF3126 + xINF3406 + xINF76 + xINF3046 + xINF3136 + xINF3146 = 1;
127 xMAT0245 + xFIS1205 + xIWN1705 + xINF2365 + xINF2765 + xINF2925 + xINF45 + xINF2255 + xINF2565 + xICN2705 + xINF2855 + xINF2935 + xINF3225 + xINF3435 + xINF2955 + xINF2465 + xINF65 + xINF3025 + xINF3035 + xINF3115 + xINF3125 + xINF3405 + xINF75 + xINF3045 + xINF3135 + xINF3145 + xINF2285 + xINF3055 + xINF3105 + xMAT0246 + xFIS1206 + xFIS1406 + xIWN1706 + xINF2366 + xINF2766 + xINF2926 + xINF46 + xINF2256 + xINF2566 + xICN2706 + xINF2856 + xINF2936 + xINF3226 + xINF3436 + xINF2956 + xINF2466 + xINF66 + xINF3026 + xINF3036 + xINF3116 + xINF3126 + xINF3406 + xINF76 + xINF3046 + xINF3136 + xINF3146 = 1;
128 xMAT0246 + xFIS1206 + xIWN1706 + xINF2366 + xINF2766 + xINF2926 + xINF46 + xINF2256 + xINF2566 + xICN2706 + xINF2856 + xINF2936 + xINF3226 + xINF3436 + xINF2956 + xINF2466 + xINF66 + xINF3026 + xINF3036 + xINF3116 + xINF3126 + xINF3406 + xINF76 + xINF3046 + xINF3136 + xINF3146 = 1;
129 y1 <= 30;
130 y2 <= 30;
131 y3 <= 30;
132 y4 <= 30;
133 y5 <= 30;
134 y6 <= 30;
135 y1 >= 5;
136 y2 >= 5;
137 y3 >= 5;
138 y4 >= 5;
139 y5 >= 5;
140 y6 >= 5;
141 y1 <= 30;
142 y2 <= 30;

File Edit Search Action View Options Help
Source Matrix Options Result

147
148 /* Definición de variables binarias */
149 bin xMAT0241;
150 bin xMAT0242;
151 bin xMAT0243;
152 bin xMAT0244;
153 bin xMAT0245;
154 bin xMAT0246;
155 bin xFIS1201;
156 bin xFIS1202;
157 bin xFIS1203;
158 bin xFIS1204;
159 bin xFIS1205;
160 bin xFIS1206;
161 bin xIWN1701;
162 bin xIWN1702;
163 bin xIWN1703;
164 bin xIWN1704;
165 bin xIWN1705;
166 bin xIWN1706;
167 bin xFIS1401;
168 bin xFIS1402;
169 bin xFIS1403;
170 bin xFIS1404;
171 bin xFIS1405;
172 bin xFIS1406;
173 bin xINF2361;
174 bin xINF2362;
175 bin xINF2363;
176 bin xINF2364;
177 bin xINF2365;
178 bin xINF2366;
179 bin xINF2461;
180 bin xINF2462;
181 bin xINF2463;
182 bin xINF2464;
183 bin xINF2465;
184 bin xINF2466;
185 bin xINF2761;
186 bin xINF2762;
187 bin xINF2763;
188 bin xINF2764;
189 bin xINF2765;
190 bin xINF2766;
191 bin xINF2921;
192 bin xINF2922;
193 bin xINF2923;

```

Figura 30 y 31: Restricciones malla de *testing* y definición de variables

Fuente: Elaboración Propia

Tras la construcción meticulosa del modelo, avanzamos con la ejecución del solver. El resultado que se nos presenta es el siguiente:

LPSolve IDE - 5.5.2.11 - C:\Users\janac\OneDrive\Documentos\TT2\Memoria\ValidationTest.lp

File Edit Search Action View Options Help

Source Matrix Options Result

Objective	Constraints	Sensitivity
Variables	M...	result
y2	19	19
y4	18	18
y3	16	16
y5	14	14
y6	11	11
xFIS1202	1	1
xFIS1403	1	1
xCN2704	1	1
xINF2252	1	1
xINF2286	1	1
xINF2361	1	1
xINF2462	1	1
xINF2565	1	1
xINF2663	1	1
xINF2764	1	1
xINF2854	1	1
xINF2921	1	1
xINF2932	1	1
xINF2953	1	1
xINF3025	1	1
xINF3034	1	1
xINF3043	1	1
xINF3096	1	1
xINF3101	1	1
xINF3114	1	1
xINF3123	1	1
xINF3132	1	1
xINF3142	1	1
xINF3224	1	1
xINF3436	1	1
xINF3605	1	1
xINF45	1	1
xINF66	1	1
xINF75	1	1
xW/N1705	1	1
xMAT0241	1	1

Figura 32: Resultado del solver

Fuente: Elaboración Propia

Es esencial recordar que la notación x_{ij} representa el valor 1 si, y solo si, la asignatura 'i' se asigna al semestre 'j'. Con esta información en mano, procedemos a alimentar la función encargada de generar la representación gráfica de la malla curricular.

4.4 Generación grafica de la malla curricular

```
def Generar_Malla(lista_final):
    for semestre in lista_final:
        contador = 0
        #dot.node(id, texto, {'pos':'x,y!','shape':'box'})
        for ramo in semestre: # Definimos colores para distintas siglas
            if( "IWN" in ramo[0]):
                color = 'darkgoldenrod1'
            elif( "FIS" in ramo[0]):
                color = 'aquamarine'
            elif( "MAT" in ramo[0]):
                color = 'firebrick2'
            elif( "ICN" in ramo[0]):
                color = 'chartreuse2'
            elif( "DEW" in ramo[0]):
                color = 'burlywood1'
            elif( "INF" in ramo[0]):
                color = 'darkgoldenrod3'
            else:
                color = 'none'
            dot.node(ramo[0],ramo[0]+'\\n'+ramo[1], {
                'pos':str(int(ramo[2])+1.2*int(ramo[2]))+', '+str(contador+2)+'!', 'shape':'box', 'fixedsize':'shape',
                'width':'1.6', 'height':'0.8', 'fillcolor':color, 'style':'filled',})
            contador+= 1
```

Figura 33: Modificación al coloreo en función Generar Malla

Fuente: Elaboración Propia

Procedemos a realizar una configuración personalizada de la función `Generar_Malla()`. Esta configuración permite asignar un color distintivo para cada sigla presente en nuestro conjunto de datos ejemplar. Una vez asegurado que las asignaturas están distribuidas en sus semestres respectivos, estructuramos la lista siguiendo el formato:

[sigla, nombre de la asignatura, semestre, lista de prerrequisitos]:

```
lista_test = [['MAT024', 'Matemáticas IV', 1, []],
['FIS120', 'Física General II', 2, []],
['IWN170', 'Economía IA', 5, []],
['FIS140', 'Física General IV', 3, ['FIS120']],
['INF236', 'Análi. y Dis. de Soft.', 1, []],
['INF246', 'Sist. Operativos', 2, []],
['INF276', 'Ing. Info. y Soc.', 4, []],
['INF292', 'Optimización', 1, []],
['INF4', 'Libre 4', 5, []],
['INF225', 'Ingeniería de Software', 2, ['INF236']],
['INF256', 'Red. de Comp.', 5, ['INF246']],
['ICN270', 'Info. y Mat. Financ.', 4, ['MAT024']],
['INF285', 'Compu. Científica', 4, ['MAT024']],
['INF293', 'Invest. de Oper.', 2, ['INF292']],
['INF322', 'Diseño Interfaces Usuarías', 4, ['INF225']],
['INF343', 'Sistemas Distribuidos', 6, ['INF256']],
['INF295', 'Int. Artificial', 3, ['INF292']],
['INF266', 'Sistemas de Gestión', 3, []],
['INF6', 'Libre 6', 6, []],
['INF302', 'Electivo Info. II', 5, []],
['INF303', 'Electivo Info.a III', 4, []],
['INF311', 'Electivo I', 4, []],
['INF312', 'Electivo II', 4, []],
['INF360', 'Gest. Proyectos Info', 1, ['INF322', 'INF266']],
['INF7', 'Libre 7', 5, []],
['INF304', 'Electivo Info. IV', 3, []],
['INF313', 'Electivo III', 2, []],
['INF314', 'Electivo IV', 2, []],
['INF228', 'Taller Des. Proy. Info.', 6, ['INF360']],
['INF309', 'TT 1', 6, ['INF360']],
['INF310', 'TT 2', 1, ['INF228', 'INF309']]
ramos_ordenados = OrdenarListaRamos(lista_test)

Generar_Malla(ramos_ordenados)
Asignar_Arcos(lista_test)
```

Figura 34: Lista basada en resultado del solver

Fuente: Elaboración Propia

Tras la ejecución de las funciones encargadas del ordenamiento, generación de mallas y asignación de arcos, obtenemos como resultado la malla curricular que se presenta a continuación:

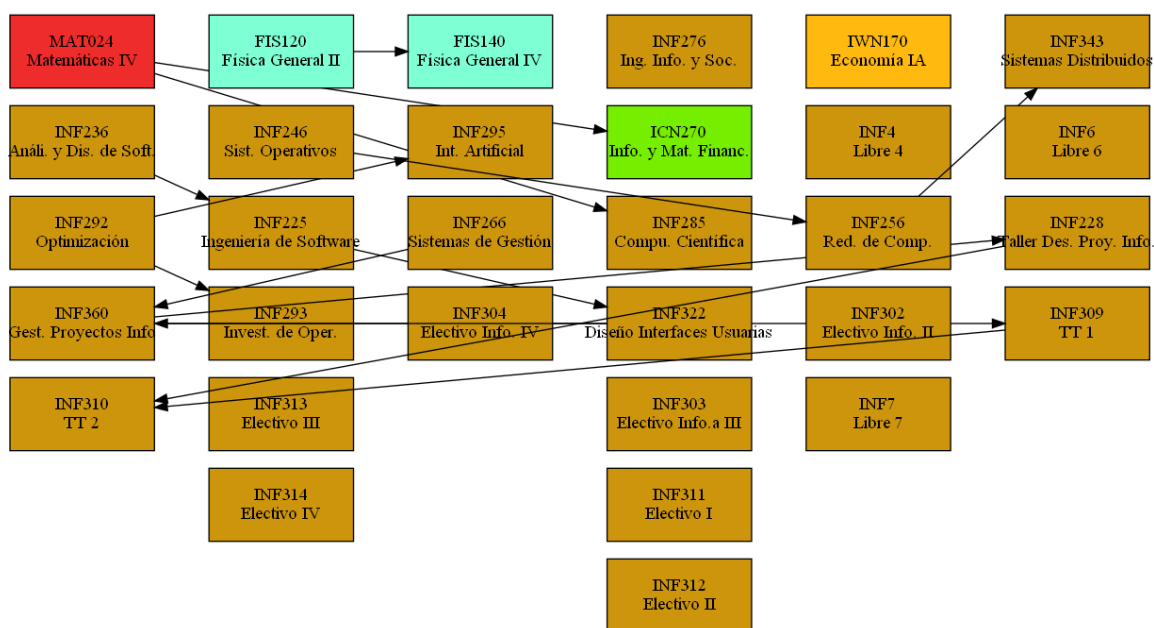


Figura 35: Malla curricular generada resultante

Fuente: Elaboración Propia

Es evidente que la disposición de los nodos recuerda a una malla curricular tradicional, con semestres organizados en columnas y asignaturas distribuidas en filas. Sin embargo, es notable que algunas asignaturas presentan prerequisites ubicados en semestres posteriores. Esto sugiere que pudo haber un inconveniente con las restricciones establecidas. Es posible que, durante el proceso de optimización, se hayan hecho ciertas concesiones para asegurar que todas las restricciones se cumplieran.

4.5 Forzar restricciones fuertes

Como es fácilmente observable, en el contexto de la UTFSM, resulta atípico, por no decir improbable, que el curso "Trabajo de Título 2" se imparta en el primer semestre disponible. Esta anomalía indica que se deben hacer ajustes en el modelo para que se adecúe al marco y contexto específico de esa malla de estudio. Por lo tanto, se define:

$$x_{INF3106} = 1$$

Adicionalmente, es esencial asegurarse de que ninguna asignatura que posea prerequisites se contemple en el inicio del primer semestre disponible. Para abordar esta situación, también se establece como restricción fuerte:

$$x_{INF3101} = 0$$

$$x_{FIS1401} = 0$$

$$x_{INF2251} = 0$$

$$x_{INF2561} = 0$$

$$x_{ICN2701} = 0$$

$$x_{INF2931} = 0$$

$$x_{INF3221} = 0$$

$$x_{INF2951} = 0$$

$$x_{INF3091} = 0$$

$$x_{INF2281} = 0$$

$$x_{INF3431} = 0$$

$$x_{INF2851} = 0$$

Se posiciona también en el solver, las restricciones de prerequisites primero debido a que en pruebas que se han hecho iterativamente, estos afectan al resultado final en las asignaciones de las variables binarias:

```

82 xFIS1402 <= xFIS1201;
83 xFIS1403 <= xFIS1201 + xFIS1202;
84 xFIS1404 <= xFIS1201 + xFIS1202 + xFIS1203;
85 xFIS1405 <= xFIS1201 + xFIS1202 + xFIS1203 + xFIS1204;
86 xFIS1406 <= xFIS1201 + xFIS1202 + xFIS1203 + xFIS1204 + xFIS1205;
87 xINF2252 <= xINF2361;
88 xINF2253 <= xINF2361 + xINF2362;
89 xINF2254 <= xINF2361 + xINF2362 + xINF2363;
90 xINF2255 <= xINF2361 + xINF2362 + xINF2363 + xINF2364;
91 xINF2256 <= xINF2361 + xINF2362 + xINF2363 + xINF2364 + xINF2365;
92 xINF2562 <= xINF2461;
93 xINF2563 <= xINF2461 + xINF2462;
94 xINF2564 <= xINF2461 + xINF2462 + xINF2463;
95 xINF2565 <= xINF2461 + xINF2462 + xINF2463 + xINF2464;
96 xINF2566 <= xINF2461 + xINF2462 + xINF2463 + xINF2464 + xINF2465;
97 xICN2702 <= xMAT0241;
98 xICN2703 <= xMAT0241 + xMAT0242;
99 xICN2704 <= xMAT0241 + xMAT0242 + xMAT0243;
100 xICN2705 <= xMAT0241 + xMAT0242 + xMAT0243 + xMAT0244;
101 xICN2706 <= xMAT0241 + xMAT0242 + xMAT0243 + xMAT0244 + xMAT0245;
102 xINF2852 <= xMAT0241;
103 xINF2853 <= xMAT0241 + xMAT0242;
104 xINF2854 <= xMAT0241 + xMAT0242 + xMAT0243;
105 xINF2855 <= xMAT0241 + xMAT0242 + xMAT0243 + xMAT0244;
106 xINF2856 <= xMAT0241 + xMAT0242 + xMAT0243 + xMAT0244 + xMAT0245;
107 xINF2921 <= xINF2921;
108 xINF2922 <= xINF2921 + xINF2922;
109 xINF2923 <= xINF2921 + xINF2922 + xINF2923;
110 xINF2924 <= xINF2921 + xINF2922 + xINF2923 + xINF2924;
111 xINF2925 <= xINF2921 + xINF2922 + xINF2923 + xINF2924 + xINF2925;
112 xINF3221 <= xINF2251;
113 xINF3222 <= xINF2251 + xINF2252;
114 xINF3223 <= xINF2251 + xINF2252 + xINF2253;
115 xINF3224 <= xINF2251 + xINF2252 + xINF2253 + xINF2254;
116 xINF3225 <= xINF2251 + xINF2252 + xINF2253 + xINF2254 + xINF2255;
117 xINF3432 <= xINF2561;
118 xINF3433 <= xINF2561 + xINF2562;
119 xINF3434 <= xINF2561 + xINF2562 + xINF2563;
120 xINF3435 <= xINF2561 + xINF2562 + xINF2563 + xINF2564;
121 xINF3436 <= xINF2561 + xINF2562 + xINF2563 + xINF2564 + xINF2565;
122 xINF2952 <= xINF2921;
123 xINF2953 <= xINF2921 + xINF2922;
124 xINF2954 <= xINF2921 + xINF2922 + xINF2923;
125 xINF2955 <= xINF2921 + xINF2922 + xINF2923 + xINF2924;
126 xINF2956 <= xINF2921 + xINF2922 + xINF2923 + xINF2924 + xINF2925;
127 xINF3602 <= xINF3221;
128 xINF3603 <= xINF3221 + xINF3222;

```

Figura 36: Reposicionamiento restricciones en solver

Fuente: Elaboración Propia

Dándonos así la nueva lista una vez pasando los valores:

```
lista_test = [['MAT024', 'Matemáticas IV', 1, []],
['FIS120', 'Física General II', 2, []],
['IWN170', 'Economía IA', 5, []],
['FIS140', 'Física General IV', 3, ['FIS120']],
['INF236', 'Análi. y Dis. de Soft.', 1, []],
['INF246', 'Sist. Operativos', 2, []],
['INF276', 'Ing. Info. y Soc.', 4, []],
['INF292', 'Optimización', 1, []],
['INF4', 'Libre 4', 3, []],
['INF225', 'Ing. de Software', 2, ['INF236']],
['INF256', 'Red. de Comp.', 4, ['INF246']],
['ICN270', 'Info. y Mat. Financ.', 4, ['MAT024']],
['INF285', 'Compu. Científica', 4, ['MAT024']],
['INF293', 'Invest. de Oper.', 2, ['INF292']],
['INF322', 'Diseño Interf. Usua.', 3, ['INF225']],
['INF343', 'Sist. Distribuidos', 6, ['INF256']],
['INF295', 'Int. Artificial', 3, ['INF292']],
['INF266', 'Sistemas de Gestión', 1, []],
['INF6', 'Libre 6', 6, []],
['INF302', 'Electivo Info. II', 1, []],
['INF303', 'Electivo Info.a III', 4, []],
['INF311', 'Electivo I', 3, []],
['INF312', 'Electivo II', 5, []],
['INF360', 'Gest. Proyectos Info', 4, ['INF322', 'INF266']],
['INF7', 'Libre 7', 5, []],
['INF304', 'Electivo Info. IV', 3, []],
['INF313', 'Electivo III', 2, []],
['INF314', 'Electivo IV', 2, []],
['INF228', 'Taller Des. Proy. Info.', 5, ['INF360']],
['INF309', 'TT 1', 5, ['INF360']],
['INF310', 'TT 2', 6, ['INF228', 'INF309']]]
```

Figura 37: Lista de asignaturas de restricciones forzadas

Fuente: Elaboración Propia

La cual genera el siguiente grafo al aplicar las funciones establecidas:

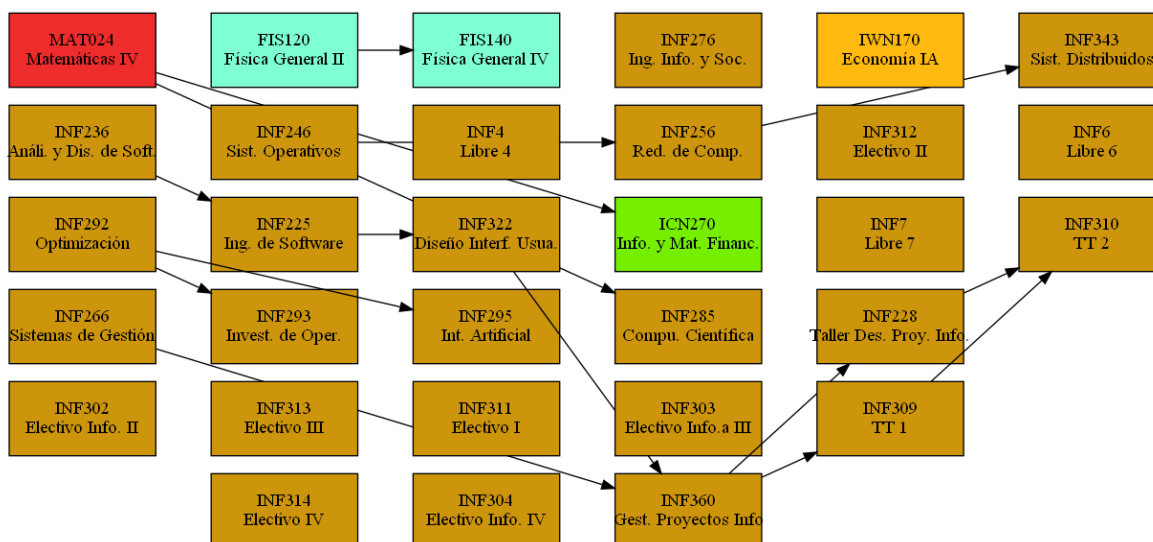


Figura 38: Malla generada restricciones forzadas

Fuente: Elaboración Propia

4.6 Resultados y análisis

Tras analizar el resultado podemos ver que esta vez se tiene una mejor representación visual ya que los prerrequisitos hacen sentido que vayan dirigidos mientras avanzan los semestres, podemos notar que:

- Se puede generar mallas curriculares utilizando esta herramienta
- Se puede seguir modificando el código para dar más personalización a las asignaturas
- Hay que revisar bien la construcción del modelo del solver para que hagan sentido con lo visualizado
- Se ha de tener una comunicación entre estos actores de flujo, desde la generación del archivo, generar la información, generar el modelo, resolverlo y con el resultado poder crear el grafo que representa la malla curricular balanceada

Después de analizar el resultado obtenido, es evidente que la herramienta desarrollada ofrece una representación visual más coherente y precisa de la malla curricular. Los prerrequisitos ahora tienen una lógica secuencial, respetando la progresión natural de los semestres. A partir de esta experiencia, podemos destacar varios puntos importantes

- La herramienta demostró su capacidad para crear mallas curriculares ajustadas a las necesidades educativas y restricciones impuestas.
- Una de las ventajas es la posibilidad de continuar adaptando y modificando el código para proporcionar mayor personalización en función de las asignaturas y las particularidades de cada plan de estudios.
- A pesar de los avances, es fundamental revisar con detalle la construcción del modelo del solver. Esto garantizará que los resultados obtenidos se alineen adecuadamente con la representación visual deseada.
- El proceso llevado a cabo reveló la importancia de mantener una comunicación fluida entre las diferentes etapas: desde la generación del archivo inicial, pasando por la extracción de información, la creación del modelo, su resolución, hasta llegar a la visualización final del grafo que representa la malla curricular balanceada.

Con este conocimiento, es evidente que, aunque se han hecho avances significativos, todavía existen áreas de oportunidad para mejorar y optimizar aún más el proceso y los resultados.

CAPÍTULO 5: CONCLUSIONES

5.1 Conclusiones generales

La tarea de ofrecer una herramienta que permita la generación de mallas curriculares balanceadas usando grafos, centrada en el avance parcial de estudiantes, se ha abordado con seriedad en este trabajo. A través de este esfuerzo, se ha logrado presentar un recurso valioso que cumple con su propósito inicial, aunque aún existe un margen para su perfeccionamiento.

5.1.1 Metodología y Herramientas de Trabajo

A lo largo del desarrollo de esta memoria, se adoptaron metodologías y herramientas específicas que facilitaron el proceso de investigación, desarrollo y prueba. Estas herramientas no sólo proveyeron un marco de trabajo efectivo, sino que también permitieron una iteración rápida y una adaptabilidad frente a los desafíos que surgieron.

Una parte crucial en nuestra metodología fue la elección de utilizar documentos de texto, específicamente en formato Word, como medio para introducir y procesar información de las mallas curriculares. Esta decisión no fue arbitraria. Los documentos de texto son ampliamente utilizados en contextos académicos y administrativos, lo que facilita su adopción y uso. Además, ofrecen flexibilidad al permitir la edición y revisión de información de manera sencilla con los lenguajes de programación utilizados. Al utilizar documentos de texto, se puede garantizar que la herramienta sea compatible con los métodos tradicionales de gestión de información en varias instituciones educativas

Jupyter Notebook fue la herramienta que se convirtió en una pieza central de este trabajo, ofreció un entorno interactivo que facilitó la experimentación directa con el código. Gracias a sus características, fue posible escribir, testear y modificar el código en Python de manera dinámica, visualizando los resultados al instante. Además, su naturaleza modular permitió organizar el trabajo en celdas individuales, lo que facilitó las pruebas, la depuración y la optimización de cada segmento de código de forma independiente. Al final, la ejecución secuencial de todas las celdas resultaba en el flujo completo del programa, con todas las restricciones y lógicas implementadas. Esta metodología modular facilitó una organización clara del código, promovió una rápida identificación de errores y permitió iteraciones ágiles durante las fases de desarrollo y validación.

Para la parte de programación lineal y resolución de problemas de optimización, se optó por utilizar LPSolver IDE. Esta herramienta proporcionó un espacio dedicado para definir, resolver y analizar problemas lineales, asegurando que los modelos desarrollados fueran

consistentes y robustos.

5.1.2 Desarrollo

El proceso de desarrollo de esta memoria resultó ser un desafío mayor al esperado, tanto en esfuerzo como en tiempo invertido. Desde el inicio, la inmersión en una profunda búsqueda de información y la necesidad de familiarizarme con diversas fuentes documentales fueron necesarias. Esta exploración se vio intensificada al tener que comprender la documentación de diversas librerías que se utilizaron para el desarrollo de funciones específicas. Si bien intenté implementar la programación lineal con librerías de Python, la falta de experiencia combinada con una documentación limitada me llevó a optar por un solver externo, con el que también tuve que invertir tiempo para familiarizarme. A su vez, el aprendizaje y adaptación de Graphviz presentó sus propios retos. A pesar de contar con buena documentación, la transición de su código base a funciones específicas en Python fue un proceso de continua iteración. Dedicar tiempo para dibujar nodos, establecer conexiones, posicionarlos adecuadamente y explorar combinaciones de personalización, con el fin de dar orden mediante la entrada de datos, reafirmó el nivel de detalle y precisión necesario para este proyecto.

5.1.3 Visualización y Representación Gráfica

La representación visual es una parte integral de la herramienta desarrollada, ya que brinda al usuario una comprensión clara y concisa de la malla curricular y la relación entre las distintas asignaturas. La imagen presentada es un claro testimonio de la capacidad de la herramienta para generar visualizaciones coherentes y ordenadas que reflejen la estructura y la lógica de la malla curricular. Como fue posible observar, cada asignatura se representa mediante un bloque de color, etiquetado con un código y el nombre de la materia. Las líneas o arcos que conectaban diferentes bloques representando las relaciones de prerrequisito entre las asignaturas, y una de las características más notables de esta visualización es su claridad y coherencia. A pesar de la complejidad inherente a una malla curricular, la herramienta logra presentar la información de manera ordenada y fácil de entender.

5.2 Logros alcanzados

5.2.1 Objetivos específicos

- Poder tomar el output de un solver que resuelva el problema del balanceo de la carga académica y representar su resultado

En cuanto al traspaso directo de información desde un solver externo, se encontraron limitaciones. El reto de tomar el output de un solver y plasmarlo de manera directa en una representación aún se mantiene y logra su objetivo. A pesar de ello, el trabajo realizado sienta las bases para futuras adaptaciones y mejoras en este aspecto.

- Representar su avance visualmente utilizando grafos conectando los ramos que son prerequisites de otros.

Se ha logrado una representación visual efectiva utilizando grafos, donde las aristas del grafo claramente indican las relaciones de prerequisites entre las asignaturas. Esto contribuye en gran medida a la comprensión intuitiva del avance académico y las interdependencias curriculares que es importante destacar.

- Ordenar los grafos de tal forma que luzca como una malla curricular, donde el eje x represente los semestres y los ramos se posicionen en el eje y

El formato deseado, con un orden tradicional que simula una malla curricular, se ha conseguido satisfactoriamente. La distribución de semestres en columnas y la disposición de asignaturas en las filas, ofrecen una perspectiva clara y ordenada del currículo.

- Validar el mismo procedimiento con distintas cargas y avances académicos.

Por último, se ha demostrado la versatilidad de la herramienta al aplicar diferentes escenarios y avances académicos. El hecho de que se puedan incorporar y visualizar distintas restricciones y avances valida la robustez y adaptabilidad del sistema propuesto

5.2.2 Objetivo General

- Ofrecer una herramienta para la generación de mallas curriculares balanceadas utilizando grafos, considerando avance parcial de los estudiantes para una representación visual de los ramos que debe tomar en cada semestre.

El objetivo principal de proporcionar una herramienta visual para la generación de mallas curriculares balanceadas se ha cumplido. Esta herramienta no solo toma en cuenta la estructura y contenido de los programas de estudio, sino que también incorpora una consideración vital: el avance parcial de los estudiantes. Es decir, se centra no solo en el diseño académico per se, sino en la experiencia y trayectoria individual de cada estudiante. Este enfoque dual ha permitido generar una representación visual que es tanto comprensiva como personalizada. Al integrar el avance parcial de los estudiantes en la herramienta, se logra una visualización que indica qué ramos o asignaturas deberían ser tomados en cada semestre. Esto orienta y facilita la toma de decisiones académicas, proporcionando claridad y dirección tanto para estudiantes como para administradores educativos.

5.3 Trabajo futuro

El desarrollo y aplicación de esta herramienta ha desvelado un abanico de oportunidades y áreas de mejora para el futuro. Una de estas áreas es la optimización del solver. A pesar de que la herramienta actual utiliza eficazmente un solver externo, hay un gran potencial en explorar y adaptar solvers más sofisticados o integrar librerías especializadas de Python. Una investigación más exhaustiva sobre las capacidades y peculiaridades de diversos solvers podría conducir a una implementación más ágil y a resultados más precisos, asegurando así un balanceo más efectivo.

Las librerías de Python presentan otra área rica en oportunidades, ya que brindan un extenso rango de funcionalidades y herramientas aún sin explorar a fondo en este proyecto. Un análisis más detenido de dichas librerías podría descubrir técnicas y estrategias que refinan y simplifican el proceso de generación y visualización de mallas curriculares.

La visualización de datos mediante Graphviz es fundamental para la comprensión y análisis de las mallas curriculares. Aunque esta herramienta ya ofrece una amplia gama de opciones de personalización, hay margen para un entendimiento más profundo y una implementación más avanzada de sus capacidades. Es esencial abordar y mejorar aspectos específicos, como la ocasional superposición de arcos y nodos. Dedicar tiempo a estudiar a fondo las características y funcionalidades de Graphviz permitirá optimizar la presentación visual de los grafos, garantizando que los nodos y conexiones se muestren de manera clara, intuitiva y estéticamente agradable.

Otro posible desarrollo a futuro sería convertir esta herramienta en una plataforma web accesible que permitiría a estudiantes, administradores y educadores interactuar con ella

de manera más sencilla y dinámica. Un sitio web dedicado proporcionaría una interfaz amigable y permitiría una integración más fluida con sistemas académicos existentes, además de facilitar la actualización y adaptación constante de la herramienta. Este desarrollo implicaría también otro aspecto importante es la retroalimentación y mejora continua. Sería esencial recopilar feedback de los usuarios y hacer ajustes basados en su experiencia. Esta retroalimentación continuará impulsando las mejoras y asegurará que la herramienta se mantenga relevante y efectiva para las necesidades cambiantes del ámbito académico.

En resumen, el camino hacia adelante es prometedor y lleno de oportunidades para expandir y perfeccionar la herramienta. A medida que la educación sigue evolucionando, es imperativo que las herramientas y recursos disponibles para estudiantes y educadores también se adapten y mejoren. La base establecida por este proyecto proporciona un punto de partida sólido para estos futuros avances y mejoras.

REFERENCIAS BIBLIOGRÁFICAS

[Alvarez13] Álvarez, M. y Parra, J. (2013) *Teoría de grafos. Seminario para optar al título de profesor de enseñanza media en educación matemática. Universidad del Bio-Bio*

[Castro01] Castro, C y Manzano, S. (June 2001) "Variable and value ordering when solving balanced academic curriculum problems," in *Proceedings of the 6th Workshop of the ERCIM WG on Constraints*

[Castro07] Castro, C. Crawford, B. y Monfroy, E. (2007) "A quantitative approach for the design of academic curricula," in *Human Interface and the Management of Information. Interacting in Information Environments*, vol. 4558 of *Lecture Notes in Computer Science*, pp. 279–288.

[Chiarandini12] Chiarandini, M, Gaspero, L. D, Gualandi, S. y Schaerf, A. (2012) "The balanced academic curriculum problem revisited," *Journal of Heuristics*, vol. 18, no. 1, pp. 119–148.

[Hnich02] Hnich, B. Kiziltan, Z. y Walsh, T. (2002) "Modelling a balanced academic curriculum problem," in *Proceedings of the CP-AI-OR*.

[Lambert05] Lambert, T. Castro, C. Monfroy, E. Riff, M. C. y Saubion, F. (October 2005) "Hybridization of genetic algorithms and constraint propagation for the BACP," in *Proceedings of the 21st International Conference on Logic Programming (ICLP '05)*, pp. 421–423.

[Lambert06] Lambert, T. Castro, C. Monfroy, E. y Saubion, F. (2006) "Solving the balanced academic curriculum problem with an hybridization of genetic algorithm and constraint propagation," in *Artificial Intelligence and Soft Computing—ICAISC 2006*, vol. 4029 of *Lecture Notes in Computer Science*, pp. 410–419.

[Monette07] Monette, J. N. Schaus, P. Zampelli, S. Deville, Y. y Dupont, P. (2007) "A cp approach to the balanced academic curriculum problem," in *Proceedings of the 7th International Workshop on Symmetry and Constraint Satisfaction Problems*.

[Rubio13] Rubio, J. M., Palma, W. Rodriguez, N. Soto, R. Crawford, B. Paredes, F. y Cabrera, G. (2013) "Solving the Balanced Academic Curriculum Problem Using the ACO Metaheuristic", *Mathematical Problems in Engineering*, vol. 2013, Article ID 793671, 8 pages.

[Tayo18] Tayo E. (2018) "El uso de Organizadores Gráficos para mejorar la destreza lectora en el aprendizaje del idioma inglés". *Universidad Técnica de Ambato*.

[Valenzuela20] Valenzuela Ibarra, M. I. (2020) Modelo de evaluación de la carga real de trabajo de las asignaturas del plan de estudio de la carrera de ingeniería civil informática en la UTFSM. Memoria para optar al título de Ingeniera Civil Industrial. Universidad Técnica Federico Santa María.

[Vergara] Vergara Claudia A. (1994) Desarrollo de un modelo de Programación Lineal para la Generación de Mallas Curriculares Balanceadas

ANEXOS

Documentación de herramientas utilizadas:

- Documentación Graphviz: <https://graphviz.readthedocs.io/en/stable/index.html>
- Documentación Lenguaje DOT: <https://graphviz.org/doc/info/lang.html>
- Documentación LPSolveIDE: <https://lpsolve.sourceforge.net/5.1/IDE.htm>

Complemento en pruebas de uso de Graphviz para generar nodos:

```
In [29]: dot3 = graphviz.Digraph(comment='Example')

#Crear Grafo
#Agregar Nodos primera columna
x1=-1
x1=str(x1)
dot3.node('a', 'input A', {'pos':x1+',0!', 'shape':'box', 'color':'darkgoldenrod1', 'style':'filled'})
dot3.node('a2', 'input A2', {'pos':x1+',1!', 'shape':'box'})
dot3.node('a3', 'input A3', {'pos':x1+',-1!', 'shape':'box'})
dot3.node('a4', 'input A4', {'pos':x1+',-2!', 'shape':'box'})

#segunda columna
x2=1
x2 = str(x2)
dot3.node('b', 'input \n B', {'pos':x2+',1!', 'shape':'box'})
dot3.node('c', 'input C', {'pos':x2+',0!', 'shape':'box'})
dot3.node('d', 'input D', {'pos':x2+',-1!', 'shape':'box'})
dot3.node('d2', 'input D2', {'pos':x2+',-2!', 'shape':'box'})

#tercera columna
x3=3
x3=str(x3)
dot3.node('e', 'E', {'pos':x3+',1!', 'shape':'box'})
dot3.node('f', 'F', {'pos':x3+',0!', 'shape':'box'})
dot3.node('g', 'G', {'pos':x3+',-1!', 'shape':'box'})

#cuarta columna
x4=5
x4=str(x4)
dot3.node('h', 'H', {'pos':x4+',1!', 'shape':'box'})
dot3.node('i', 'I', {'pos':x4+',0!', 'shape':'box'})
dot3.node('j', 'J', {'pos':x4+',-1!', 'shape':'box'})
dot3.node('k', 'K', {'pos':x4+',-2!', 'shape':'box'})

#Agregar Conexion
dot3.edge('a2', 'b')
dot3.edge('a', 'c')
dot3.edge('a4', 'd')

dot3.edge('b', 'e')
dot3.edge('c', 'g')

dot3.edge('e', 'h')
dot3.edge('g', 'j')
dot3.edge('f', 'i')

dot3.edge('a2', 'f')
```

Figura 39: Ejemplo código de generación de grafo
Fuente: Elaboración Propia

```
In [30]: dot3.engine = 'neato'
dot3
```

```
Out[30]:
```

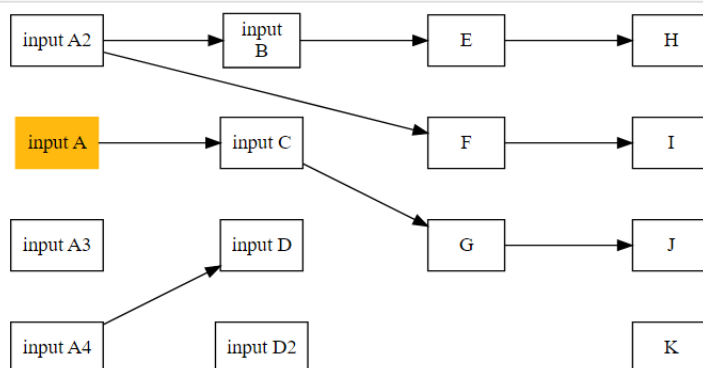


Figura 40: Resultado de generación de grafo Ejemplo
Fuente: Elaboración Propia

```
In [9]: print(dot3.source)
```

```
// Example
digraph {
  a [label="input A" color=darkgoldenrod1 pos="-1,0!" shape=box style=filled]
  a2 [label="input A2" pos="-1,1!" shape=box]
  a3 [label="input A3" pos="-1,-1!" shape=box]
  a4 [label="input A4" pos="-1,-2!" shape=box]
  b [label="input B" pos="1,1!" shape=box]
  c [label="input C" pos="1,0!" shape=box]
  d [label="input D" pos="1,-1!" shape=box]
  d2 [label="input D2" pos="1,-2!" shape=box]
  e [label=E pos="3,1!" shape=box]
  f [label=F pos="3,0!" shape=box]
  g [label=G pos="3,-1!" shape=box]
  h [label=H pos="5,1!" shape=box]
  i [label=I pos="5,0!" shape=box]
  j [label=J pos="5,-1!" shape=box]
  k [label=K pos="5,-2!" shape=box]
  a2 -> b
  a2 -> c
  a4 -> d
  b -> e
  c -> g
  e -> h
  g -> j
  f -> i
  a2 -> f
}
```

Figura 41: Código fuente en lenguaje DOT
Fuente: Elaboración Propia