UNIVERSIDAD DE CONCEPCIÓN - CHILE
FACULTAD DE INGENIERIA
DEPARTAMENTO DE INGENIERÍA INDUSTRIAL

El problema de *timetabling* para colegios chilenos. Solución mediante Algoritmos Genéticos.

por

Víctor Yamil Neira González

Profesor Guía:

Dra. Lorena Pradenas Rojas

Concepción, Noviembre de 2014

Tesis presentada a la

DIRECCIÓN DE POSTGRADO
UNIVERSIDAD DE CONCEPCION

Para optar al grado de

MAGISTER EN INGENIERÍA INDUSTRIAL

Dedicado a:

“Mis papas, mi hermano, mi amada polola

y todos los que siempre estuvieron conmigo”

RESUMEN

El problema de timetabling para colegios chilenos. Solución mediante Algoritmos Genéticos.

Víctor Yamil Neira González

"Octubre de 2014"

PROFESOR GUIA: Dra. Lorena Pradenas Rojas

PROGRAMA: Magíster en Ingeniería Industrial

En el presente trabajo se desarrolló un modelo matemático entero para el problema de asignación de horarios de asignaturas y profesores para colegios. Para la resolución del problema se utilizó el software Lingo en problemas de tamaño pequeño, y se diseñó un método heurístico basado en Algoritmos Genéticos para la resolución de problemas reales de mayor tamaño. También se diseñó una aplicación en Visual Basic para la captura de datos de instituciones educacionales de Chile, la cual hace cumplir las normas legales vigentes de educación. La aplicación fue creada para el manejo de personas no familiarizadas con optimización, por lo cual su uso es intuitivo al usuario. Se obtuvieron resultados factibles en tiempos computacionales deseados para problemas teóricos chilenos (menores a 2000 segundos) y para problemas en colegios internacionales (obtenidos de la literatura) con un total de 18 instancias.

Palabras Claves: programación de horarios, metaheristica, algoritmo genético, horarios de colegios.

ABSTRACT

A chilean school timetabling problem. Solution based on Genetic algorithm

Víctor Yamil Neira González

October 2014

THESIS SUPERVISOR: Dra. Lorena Pradenas Rojas
PROGRAM: Master in Industrial Engineering

In the present work, a mathematical model for school timetabling problem was developed. To solve this problem, the Lingo software was used in small-sized instances, and a metaheuristic method, a genethic algorithm was designed to solve real problems. This algorithm used a double mutation process for school timetabling. A Visual Basic application complying with legal norms was also designed to obtain data collection of Chilean educational institutions. This application was meant to be used by people unfamiliar with optimization; therefore its use is intuitive. Achievable results were obtained from desirable computational times for theoretical problems in Chile (less than 2000 seconds), and problems in international schools (obtained from the literature), with a total of 18 instances.

Keywords: **scheduling, metaheuristic, genetic algorithm, school timetable.**

INDICE

Tablas

Figuras

CAPITULO I Introducción

1.1. Introducción

Los problemas de programación de horarios son de alta complejidad por lo que su resolución en tiempos computacionales razonables requiere el uso de metaheurísticas en la mayoría de las situaciones reales. En este estudio, se propone un modelo matemático y un algoritmo para enfrentar este problema. La construcción del algoritmo para un establecimiento educacional es un problema de mucha dificultad debido a su alta combinatoria (Burke y Petrovic, 2002). El objetivo es minimizar una función que penaliza ciertas acciones tales como clases en horarios no deseados, varias cátedras de una misma asignatura en el mismo día, entre otras. También se considera que existen dos tipos de restricciones (Carter et al., 1996) , las restricciones fuertes y las blandas. El primer tipo corresponde a aquellas restricciones que deben satisfacerse siempre, por lo que definen un espacio de solución; mientras que las del segundo tipo pueden ser o no satisfechas, y en el caso de no satisfacerse, se tiene una penalización traducida en la función objetivo como costo.

Para la obtención de soluciones de problemas reales se hace necesario el uso de metaheurísticas, de modo de encontrar una buena solución en un tiempo razonable. Las restricciones fuertes que se considerarán son las recurrentes en la literatura (Burke y Petrovic, 2002), que en su mayoría corresponden a situaciones imposibles de cumplir en la vida real, como por ejemplo, un profesor no puede hacer clases a dos secciones en el mismo horario, en una sala no puede dictarse dos asignaturas distintas al mismo tiempo, el número de personas en una sala no puede exceder la capacidad de esta, etc. Para el caso de restricciones débiles, estas se asocian a que la carga de trabajo diaria para alumnos sea equilibrada, por lo que para un mismo grupo y en un día determinado, existe un número máximo de periodos a asignar. También se considera como restricción débil las prioridades o penalidades sobre asignaturas que pueden existir para que sean dictadas en días y periodos determinados (Lewis, 2008).

1.2. Hipótesis

Es posible proporcionar soluciones con bajos tiempos computacionales, en comparación a los métodos manuales, para la programación de horaria de un colegio chileno con un algoritmo genético y lograr una estandarización de los requerimientos y programación de horarios de colegios con la legislación vigente en Chile.

1.3. Objetivo General

Obtener una solución mediante un enfoque de algoritmos genéticos, a la programación horarios de un colegio chileno, operable por un usuario común, con soluciones en tiempos computacionales razonables y con una interfaz.

1.4. Objetivos Específicos

- Realizar revisión bibliográfica de programación de horarios.
- Proponer un modelo matemático para el problema de programación de horarios de colegios chilenos.
- Desarrollar las instancias de pruebas.
- Resolver con modelamiento propuesto y algoritmo genético.
- Desarrollar una interfaz apropiada para un usuario.

CAPITULO II Descripción del problema

2.1. Sistema educacional chileno

El sistema educacional de enseñanza primaria y secundaria de Chile divide por niveles de escolaridad, siendo los siguientes:

- 1°, 2°, 3°, 4° año del nivel básico (ciclo I)
- 5°, 6°, 7°, 8° año del nivel básico (ciclo II)
- 1° , 2° , 3° y 4° año de enseñanza media

Según la orientación de los contenidos que imparte la institución (propio de enseñanza media), las clasificaciones en la enseñanza media son las siguientes:

- Enseñanza científico-humanista
- Enseñanza técnico profesional

Según la procedencia de los recursos económicos para sostener la institución educacional, se pueden clasificar en:

- Colegios públicos: actualmente a cargo de las correspondientes municipales. Los fondos provienen del estado.
- Colegios subvencionados: actualmente los fondos provienen tanto del estado como de los apoderados de los alumnos. Estas instituciones deben cumplir una serie de metas para ser aptas de recibir la subvención.
- Colegios particulares pagados: No reciben fondos del estado, solo se sostienen con los pagos de los apoderados de sus alumnos.

Finalmente, existen colegios que poseen distintas jornadas, según el tiempo total que utilizan dentro del día:

- Jornada escolar completa (38 horas semanales en promedio)
- Jornada escolar no completa (30 horas semanales en promedio)

La legislación de Chile da una serie de asignaturas obligatorias para las instituciones de educación escolar. Cada año escolar presenta un conjunto de asignaturas a impartir, junto con los contenidos mínimos y las horas dedicadas. Estos requerimientos son anuales y semanales. Sin embargo, dependiendo de la institución, pueden existir asignaturas opcionales, que la institución puede elegir como obligatorios (adicionales a los legales) y además asignaturas opcionales, en los cuales son los alumnos los que deciden si cursarlos o no.

Por motivos de simplificación, se trabajará el caso de colegios que posean niveles educacionales desde 1° básico hasta 4° medio en régimen científico-humanista, con una jornada escolar no completa. Se consideran 40 semanas efectivas de trabajo para las clases.

Para los casos seleccionados, a continuación se muestran las asignaturas obligatorias y las cantidades de horas en la semana en el primer ciclo educaciones correspondientes a la Tabla 1 y Tabla 2:

Asignatura	**Horas anuales**	
	1° a 4° Básico	5° y 6° Básico
Lenguaje y comunicación	304	228
Inglés	-	114
Matemática	228	228
Historia, Geografía y Ciencias Sociales	114	152
Ciencias Naturales	114	114
Artes Visuales	76	38
Música	76	38
Educación Física y Salud	114	76
Orientación	19	38
Tecnología	19	38
Religión	76	76
Total tiempo escolar	1140	1140

Tabla 1: Distribución de horas 1° a 6° básico

Asignatura	7° y 8° Básico horas semanales
Lenguaje y comunicación	6
Idioma Extranjero	3
Matemática	6
Historia, Geografía y Ciencias Sociales	4
Ciencias Naturales	4
Educación Tecnológica	1
Educación Artística	2
Educación Física	2
Orientación	1
Religión	2
Libre disposición	2
Total mínimo	33

Tabla 2: Distribución de horas 7° y 8° básico

Luego, en el segundo ciclo, los alumnos poseen asignaturas de un nivel de complejidad más avanzado, como preparación para la educación superior, como se muestra en la Tabla 3 y Tabla 4, además se puede observar que la institución cuenta con la posibilidad de añadir horas de libre disposición, con un tope de 6 horas, pudiendo así complementar el aprendizaje con nuevas asignaturas o incluir más horas de las ya existentes:

Asignatura	1° y 2° medio horas semanales
Lenguaje y comunicación	6
Idioma Extranjero	3
Matemática	6
Biología	2
Física	2
Química	2
Historia, Geografía y Ciencias Sociales	4
Educación Tecnológica	1
Artes visuales o Artes musicales	2
Educación Física	2
Orientación	1
Religión	2
Libre disposición	0
Total mínimo	33

Tabla 3: Distribución horaria 1° y 2° medio

Asignatura	3° y 4° medio horas semanales
Lenguaje y comunicación	3
Idioma Extranjero	3
Matemática	3
Historia y Ciencias Sociales	4
Filosofía y Psicología	3
Biología	2
Física	-
Química	2
Artes visuales o Artes musicales	2
Educación Física	1
Consejo de Curso	2
Religión	2
Formación diferenciada	9
Libre disposición	0 a 6
Total mínimo	36 a 42

Tabla 4: Distribución horaria 3° y 4° medio

Todos los datos fueron recopilados de los informes curriculares del MINEDUC (Ministerio de educación) para todos los posibles tipos de instituciones educacionales existentes y clasificados de acuerdo al caso en el cual se escogió trabajar en el estudio (Ministerio de Educación, 2013). Además se tomaron como requisitos para el método de solución, el cumplimiento de las normas chilenas en los procesos educacionales.

CAPITULO III Revisión bibliográfica

3.1. Antecedentes

Un problema de programación horaria (*Timetabling problem*) consiste en la generación de horarios de un conjunto de tareas necesarias a realizar. Es un problema ampliamente aplicado en el mundo real. Dentro de los usados se encuentran: Programación de vuelos de aviones, horarios en transportes, programación de competencias deportivas, distribución horaria del personal de un hospital, ordenamiento de las tareas y maquinas en un taller, y por último la programación horaria en establecimientos educacionales.

Los problemas de planificación de horarios son de alta complejidad por lo que su resolución en tiempos computacionales razonables requiere el uso de heurísticas en la mayoría de las situaciones reales. La obtención de la solución horaria para un establecimiento educacional es un problema de mucha dificultad debido a su alta combinatoria (Burke y Petrovic, 2002). El objetivo es minimizar una función que penaliza ciertas acciones tales como clases en horarios no deseados, varias cátedras de una misma asignatura en el mismo día, entre otras. También se considera que existen dos tipos de restricciones (Carter et al., 1996): las restricciones fuertes y las blandas. El primer tipo corresponde a aquellas restricciones que deben satisfacerse siempre, por lo que definen un espacio de solución; mientras que las del segundo tipo pueden ser o no satisfechas, y en el caso de no satisfacerse, se tiene una penalización traducida en la función objetivo como costo.

En la mayoría de las instituciones educacionales en Chile, la solución a este problema es construida de manera manual, la construcción del horario puede tomar días o semanas de trabajo, no permite establecer restricciones que optimicen el uso del recurso humano y de infraestructura, y por último, da lugar a errores en la asignación de asignaturas y de los docentes (Hernández et al., 2008).

Lo anterior da un amplio espacio para el desarrollo de investigación en este tipo de problemas, y así lo evidencia las diversas técnicas analíticas y heurísticas a la solución del problema de programación de horarios escolares tales como: grafos, *Simulated Annealing*

(Abramson, 1991), Algoritmos Genéticos (Beligiannis et al., 2008), programación con restricciones (Valouxis and Housos, 2003), algoritmos de búsqueda local (Avella et al., 2007), *Tabu Search* (Lü and Hao, 2010), entre otros métodos.

El problema de asignación de horarios para instituciones educacionales se puede dividir de acuerdo al tipo de problema a resolver. Pueden ser problemas horarios de colegios, de universidades o de programación de exámenes para instituciones educacionales. En el caso específico del estudio actual, el problema de programación horaria para colegios es referido en la literatura como STP (*School Timetabling Problem*).

Además, el STP es catalogado como NP-hard o NP completo según las restricciones que contenga el problema. Existen estudios desde el año 1975 que indican la complejidad de éste, según sus características y espacio de soluciones existentes (Even et al., 1975).

3.2. Taxonomía

Según los autores mencionados, el problema tratado, el STP puede tener distintas restricciones a tratar, además de la clasificación antes dada de restricciones blandas y duras (Schaerf, 1999). A continuación se presenta una taxonomía de los distintos requerimientos que se utilizan como restricciones blandas o duras, extraídas de la literatura (Pillay, 2014) :

Requerimientos de la institución:

- PR1: Las clases deben ser programadas según en número requerido para cada curso.
- PR2: Los profesores deben ser programados según el requerimiento de cada clase.
- PR3: Se requiere que exista un horario de descanso o alimentación.

No existencia de choques o sobreposición de horario:

- NC1: Una clase no debe ser programada más de una vez en un mismo bloque de tiempo.
- NC2: Un profesor no debe ser programado más de una vez en un mismo bloque de tiempo.

- NC3: Una sala de clases no debe usarse más de una vez en un mismo bloque de tiempo.

Restricción de utilización de recursos:

- RU1: Tiempo de viaje entre salas de clases.
- RU2: Los profesores solo pueden ser programados cuando estén disponibles.
- RU3: La capacidad de las salas de clases no debe ser excedida.
- RU4: Todas las salas de clases deben ser utilizadas.
- RU5: Ciertas clases requieren el uso de salas especializadas para ellas.
- RU6: En el caso de que la institución tenga sedes en distintos lugares geográficos, las asignaciones de salas para un curso deben realizarse dentro de la misma sede.
- RU7: Las salas no deben tener asignaciones de clases mientras se encuentren en mantenimiento.
- RU8: Durante ciertos períodos de tiempo no pueden existir clases.

Restricciones de carga de trabajo

- W1: Existen cantidades de horas máximas y mínimas que un profesor puede realizar, ya sea en cada día o la semana de trabajo.
- W2: Las clases dentro de un día para un grado de curso en particular deben ser las mismas, pero pueden diferir para grado distintos. Ejemplo: cuarto medio A, B y C tienen el mismo número de horas, pero tercero medio C puede tener un numero distinto de horas respecto a los tres anteriores.
- W3: Para cada grado de curso debe existir un máximo y mínimo de horas diarias en la semana para asignación de clases.

Restricciones de distribución de periodos:

- PD1: Existencia de periodos libres de clases, dependiendo de la institución o el grado del curso.

- PD2: Existencia de periodos libres de clases para profesores. Adicionalmente se aconseja que esos periodos deben minimizarse (Birbas et al., 2009).
- PD3: Distribución de las asignaturas dentro de la semana para un curso. En algunos STP se requiere que ciertas asignaturas sean distribuidas dentro de la semana, o que sean compactadas lo más posible dentro de la misma semana.
- PD4: Distribución de los profesores dentro de la semana. Algunos STP requieren que las asignaturas realizados por cierto profesor estén esparcidos durante toda la semana, u otros que estén concentrados en lo posible.

Restricciones de preferencias:

- P1: Preferencias de clases en la semana. Algunas asignaturas preferiblemente se deben dictar en ciertos horarios, muy temprano o muy tarde dentro del día. O en un día determinado.
- P2: Preferencia de profesores en la semana. Según los tipos de STP, en algunos casos se puede preferir que el horario de un profesor sea en un conjunto de determinados días y horas de la semana, preferentemente.

Restricciones de clases:

- L1: Algunas clases ya están programadas en periodos específicos antes de iniciar la programación horaria en general.
- L2: Es necesario que algunas clases sean programadas antes o después de otras clases. Ejemplo: trabajo práctico de algunas clases de matemática debe ser programado después de una clase teórica de matemáticas.
- L3: Clases dobles o triples consecutivas según requerimiento de la institución.
- L4: La fusión o división de cursos en grupos para programación de clases.
- L5: Ciertas clases deben ser programadas simultáneamente.

Dentro de la literatura estudiada, la evaluación de las soluciones (*Fitness*) o función de evaluación es tratada de dos formas distintas. La primera es la suma de las violaciones a las restricciones, blandas y duras (Valouxis and Housos, 2003). La segunda forma de

evaluación es la suma ponderada de las mismas violaciones a las restricciones. En este segundo caso, se establecen que ciertas restricciones tienen más peso que otras, según los requerimientos del problema tratado (Wright, 1996). Ambas formas de la literatura dan la posibilidad de trabajar con soluciones infactibles del problema, incluso con más de una violación a las restricciones.

Azami plantea en el año 2005 algoritmos de *Simulated Annealing , Tabu Search , Ant Colony* y Algoritmos Genéticos para la resolución de problemas combinatorios (Naji Azimi, 2005). En su artículo verifica la capacidad de cada tipo de algoritmo por separado para luego probar la eficiencia de nuevos métodos, que consistían en combinar pares de algoritmos y hacerlos funcionar en conjunto. Para cada algoritmo, se plantea una estructura de solución tipo vector y métodos híbridos. Como resultado se obtuvo que la combinación de los algoritmos produjera que la convergencia a soluciones sea mucho más rápida en un tiempo computacional deseable.

Existe un algoritmo (Alvarez-Valdes et al., 2002) desarrollado en el año 2002 el cual se basa en el algoritmo de *tabú search*. La implementación del algoritmo se usó para resolver el *timetabling* de una universidad de Valencia y cuenta con una interfaz amigable para un usuario no especializado en materia de Investigación Operativa. El método usado es una modificación del algoritmo *tabu search* adaptado desde el modelo matemático del problema específico de la universidad. El algoritmo consta de 3 fases para construir la asignación de horarios. La primera construye una asignación inicial tomando una a una las clases secuencialmente y de acuerdo a la prioridad del usuario. La segunda mejora la asignación de la fase 1, usando 52 diferentes combinaciones de estrategias para la mejora. La tercera fase, realiza una mejora en la asignación solo de las salas, tomando lo demás como constante. El objetivo en esta fase es disminuir el número de cambios de salones por los estudiantes, minimizando así la distancia recorrida por ellos.

En el 2011 es desarrollado el primer algoritmo que resuelve problemas del tipo *timetabling* usando la metaheuristica “colonia de abejas” (Sabar et al., 2012). La justificación para el uso de este tipo de método fue la facilidad de resolución de problemas del tipo NP-HARD por parte de éste tipo de heurísticas. La heurística incorpora una serie de modificaciones para un Simulated Annealing normal, lo cual hace que sea más competitivo frente a

problemas de programación horaria. Las modificaciones del algoritmo incluyen el uso de cruzamiento en los pasos y el uso de búsqueda en el espacio de solución en base a la función de *annealing*, entre otros. Estas modificaciones no han sido utilizadas antes en la literatura para este tipo de problemas, con lo cual la comparación de la efectividad del algoritmo se hace comparando otros tipos de algoritmos en iguales instancias. El método anterior se usó para resolver el problema de horarios para los cursos y el problema de asignación de horarios para los exámenes de los cursos. Los datos fueron obtenidos de casos estudiados en la literatura anteriormente. Los resultados fueron deseables en términos de tiempos computacionales.

En el año 2011, se creó un método basado en *Branch and price*, usando una partición Dantzing-Wolfe (Kristiansen et al., 2011) para 98 colegios secundarios de Dinamarca. El modelo utiliza programación entera, mediante un modelo de programación matemática muy similar al de la mayoría de los estudios de programación horaria de la epoca. Debido a que se busca una solución óptima del problema, los colegios elegidos debían tener tamaños relativamente pequeños, para asegurar encontrar la solución en un tiempo razonable (aun así un tiempo más largo que usando metaheurísticas). Finalmente para los 98 colegios estudiados se llegó a soluciones óptimas en tiempo deseables (alrededor de 1 hora).

En el año 2010, se desarrolló un algoritmo basado en la metaheurística *Simulated Annealing* (Zhang et al., 2010) aplicado para un problema de colegio (*high school*) el cual poseía la alteración en la estructura de búsqueda en el vecindario. El algoritmo base fue modificado de tal forma que la búsqueda local fuera más extensa que lo normal y cambiando la estructura de búsqueda también. El algoritmo fue evaluado mediante comparación de dos problemas de colegios distintos, en los cuales los resultados fueron exitosos y bien evaluados. Finalmente, la ampliación de la búsqueda local no hizo que en los resultados finales se registraran aumentos de tiempos muy significativos, con lo cual el algoritmo seguía siendo efectivo.

3.3. Últimos avances

En 2013 Victor Suares y otros plantean la solución del problema de programación óptima de horarios escolares de una escuela pública Colombiana. Para esto se considera la asignación adecuada de salas, docentes y además los ritmos cognitivos que presentan los estudiantes como el factor más fuerte dentro de la optimización. Se propone como método de solución el Algoritmo Genético de clasificación no dominada. Dentro de los resultados se puede apreciar la disminución en la deserción de los distintos cursos en comparación a los cursos que fueron ordenados con otros métodos (Suárez et al., 2013).

En el año 2013, se desarrolló un algoritmo heurístico basado en operadores evolutivos, mediante un algoritmo memético. La herramienta fue probada en problema de programación de horarios de universidades de Iran. El algoritmo contiene los operadores de carácter genético y dentro de la parametrizacion se probó que se pueden obtener resultados deseados tanto para instancias medianas como grandes (reales) con un total de 2000 generaciones de conjuntos de soluciones del problema de horario (Qaurooni and Akbarzadeh-T, 2013).

Además de los métodos utilizados, es muy importante agregar la representación del espacio de los problemas de asignación horaria. En 2014, se crea un algoritmo a base de "colonia de Hormigas" (Thepphakorn et al., 2014), simula la construcción del mejor camino, formado por hormigas, desde la colonia, hasta el alimento deseado. El algoritmo toma en consideración de estructura de un cubo, formado de cubos más pequeños y que emula la asignación en 3 dimensiones significativas de conjuntos para el problema horario. La visualización en 3 dimensiones ayuda a generar un mejor método de resolución. El problema es abordado emulando rutas de hormigas que pasan por distintos nodos. Cada nodo corresponde a una asignatura que debe impartirse en un curso. El camino completo denota el horario para todos los cursos de la institución, como un vector único, empezando por el primer curso con su primera asignatura, del primer día y la primera hora disponibles.

Luego, se desarrollaron múltiples búsquedas locales, las cuales corresponden a cambios en los nodos visitados dentro del vector. Cada cambio de ordenamiento de nodo, va acompañado con una ponderación probabilística, a medida que se encuentran mejores soluciones; aumentando así la posibilidad de encontrar un óptimo global.

Finalmente, existe un estudio que recopila un conjunto de métodos para resolver el problema de la programación de horarios (Pillay, 2014).

Método	**Cantidad**
Bee algorithms	1
Constraint programming	2
Constraint satisfaction methods	2
Cyclic transfers	1
Evolucionary algorithms	23
GRASP	1
Interger programming	7
Neural networks	1
Simulated annealing	4
Tabu search	8
Threshold accepting	1
Tiling algorithms	3
Walk down Jump up algorithm	1
Hibrid approaches	20
Comparative studies	4
Distributed methods	2

Tabla 5: Estudios de Timetable Problem, Fuente: Estudio Pillay, 2014

Se ve en la Tabla 5 que los algoritmos evolutivos son las herramientas de mayor uso en este tipo de problemas. Incluso, en los híbridos, al ser altamente compatibles con otro tipo de técnicas.

CAPITULO IV Materiales y métodos

4.1. Modelamiento matemático

A continuación, y basada en la literatura (Pillay, 2014) y de acuerdo con la actual normativa vigente en Chile, se propone un modelo de programación matemática el cual es detallado a continuación:

Los supuestos considerados son:

- Actual sistema educacional de Chile hasta el año 2013.
- Niveles desde 1° Básico hasta 4° Medio.
- Salas totales suficientes para los alumnos.
- Los horarios son para grupos de alumnos (curso) y no de manera individual.
- La institución cuenta con suficientes profesores para formar horarios de cursos factibles.
- No se consideran distancias entre las salas de clases.
- Se penaliza si existen 3 o más horas de una sola asignatura dentro de un mismo día.
- Se valoriza el agrupar en pares de horas las asignaturas iguales.

Sean los siguientes conjuntos:

I: Días disponibles en la semana para impartir clases en el colegio (1…I)

J: Horas dentro del día disponible para impartir clases (1…J)

K: Profesores disponibles para realizar la clase (1…K)

M: Asignaturas disponibles a impartir en el colegio (1…M)

N: Cursos que posee el colegio (1…N)

Z: espacio de separación posible, en horas, entre dos asignaturas iguales (1…Z)

Sean los siguientes parámetros:

A_{km} { 1, si el profesor k puede dictar la asignatura m
0, en otro caso

B_{nm} : Cantidad de veces que se dicta semanalmente la asignatura m en el curso n

D_z : Puntaje asociado a asignar dos asignaturas iguales con una separación de z horas

E_{ni} : Cantidad máxima de horas que se le puede asignar al curso n en el día i

Hor_k : Cantidad máxima de horas a las cuales puede ser asignado el profesor k

Sean las siguientes variables de decisión:

X_{ijnm} { 1, si en el curso *n* se asigna la asignatura *m* en el día *i* y la hora *j*
0, en otro caso

Y_{ijknm} { 1, si en el curso *n* se asigna al profesor *k* en el día *i* y la hora *j* para la asignatura *m*
0, en otro caso

Variable auxiliar:

∂_{ijnmz} { 1, si en el curso n, día i y en las horas j y(j+z+1), existe la asignatura m
0, en otro caso

Modelo:

Se obtuvo un modelo lineal entero. Para la función objetivo se desea minimizar las penalizaciones a los escenarios no deseados. En este caso se penaliza si existen 2 o más asignaturas el mismo día, también se hace aumentar la función objetivo si existe un agrupamiento de dos asignatura iguales en pares. Luego, para el caso de los profesores, para no producir una sobreocupación de algunos docentes, se calcula como costo la varianza del total de horas que posee cada profesor (como aporte para este trabajo) de manera de distribuir las horas en todo el personal docente.

Para el modelo es necesario calcular y penalizar las desviaciones horarias de los profesores. Para esto se utilizará como penalización, la varianza de las horas asignadas a un profesor, respecto al promedio de horas por profesor. Para calcular el promedio de horas por profesor se utilizará la expresión (1) y para calcular el número de horas de un profesor determinado la expresión (2):

$$PRO = \frac{\sum_{m=1}^{M}\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{n=1}^{N}(Y_{ijknm})}{K} \quad (1)$$

$$HP_k = \sum_{m=1}^{M}\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{n=1}^{N} Y_{ijknm} \quad (2)$$

Entonces el modelo será:

$$MIN\ Z = \sum_{z=1}^{Z}\sum_{i=1}^{I}\sum_{j=1}^{J-(z+1)}\sum_{n=1}^{N}\sum_{m=1}^{M}\left(\partial_{ijnmz} \times D_z\right) + \sum_{k=1}^{K}(HP_k - PRO)^2 \quad (3)$$

$$Y_{ijknm} \leq A_{km} \quad \forall k, \forall m, \forall i, \forall j, \forall n \quad (4)$$

$$\sum_{m=1}^{M}\sum_{k=1}^{K} Y_{ijknm} \leq 1 \quad \forall i, \forall j, \forall n \quad (5)$$

$$\sum_{m=1}^{M} X_{ijnm} \leq 1 \quad \forall i, \forall j, \forall n \quad (6)$$

$$\sum_{i=1}^{I}\sum_{j=1}^{J} X_{ijnm} = B_{mn} \quad \forall m, \forall n \quad (7)$$

$$\sum_{j=1}^{J}\sum_{m=1}^{M} X_{ijnm} \leq E_{ni} \quad \forall n, \forall i \quad (8)$$

$$HP_k \leq Hor_k \quad \forall k \quad (9)$$

$$\partial_{ijnmz} \leq X_{ijnm} \quad \forall z, \forall i, \forall n, \forall m, (\forall j \leq J - z - 1) \quad (10)$$

$$\partial_{ijnmz} \leq X_{i(j+z+1)nm} \quad \forall z, \forall i, \forall n, \forall m, (\forall j \leq J - z - 1) \quad (11)$$

$$X_{ijnm} + X_{i(j+z+1)nm} - \partial_{ijnmz} \leq 1 \ \forall z, \forall i, \forall n, \forall m, (\forall j \leq J - z - 1) \quad (12)$$

$$\sum_{k=1}^{K} Y_{ijknm} = X_{ijnm} \quad \forall i, \forall j, \forall n, \forall m \quad (13)$$

Se debe garantizar que un profesor no puede realizar una asignatura que no le corresponda (4) y además, respetar su máximo número de horas a trabajar (9). Luego para todos los cursos, en cualquiera de sus horarios, solo se puede asignar a un profesor para realizar la asignatura asignada (5); también en cualquiera de sus horarios, solo se puede asignar una asignatura para ser realizada (6) y además, se deben asignar las asignaturas mínimas correspondientes en la semana según la legislación vigente de Chile (7). Ahora si bien lo anterior genera un horario requerido para los distintos cursos, es posible crear un sobre agrupamiento en unos pocos días para la semana. Por eso mismo, para todos los niveles de cursos y para cada día, no se puede exceder el horario de salida según el propuesto por el colegio (8). La expresión (10), (11), y (12) se utilizan para otorgar la linealidad al problema con la variable binaria incorporada. Finalmente, la expresión (13) hace que exista un profesor asignado en un horario si es que alguna asignatura ha sido programada para el curso.

Según las clasificaciones de problemas de *timetabling* (Pillay, 2014), la clasificación del problema tratado seria:

PR1, PR2, NC1, NC2, NC3, RU2, W1, W2, W3, PD3, PD4, L3.

Y se asumen cumplidas, de acuerdo al modelo, las siguientes restricciones:

RU3, RU4, RU5.

Para el problema de horarios, se generó un conjunto de instancias ficticias para ser probadas en software de optimización LINGO 11.0. Con lo anterior, se puede verificar que existen soluciones exactas para el problema pequeño. De esta manera, también se puede revisar el desempeño del modelo.

Las instancias creadas varían el problema dependiendo de: tamaño, número de profesores disponibles, número de cursos, cantidad de asignaturas requeridas para cada curso y días de la semana en que hay clases.

Las instancias de prueba utilizadas para el modelo de programación matemática entera se muestran en la Tabla 6:

Instancias	Profesores	Asignaturas por nivel	Cursos	Días de clases
A1	10	3	1	1
A2	15	5	2	1
B1	20	5	2	2
B2	20	5	2	4
B3	30	5	14	5
C	40	7	16	5

Tabla 6: Instancias de prueba utilizadas

Para la resolución del problema se utilizará, además del modelo de programación matemática, una metaheuristica basada en algoritmos genéticos. El modelo matemático implementado en el software es de carácter no lineal-binario.

4.2. Metaheurística utilizada

Los algoritmos genéticos usan como analogía el comportamiento evolutivo de individuos. Cada individuo representa una solución del problema tratado y tiene un valor único (fitness) dependiendo de sus características también únicas. El valor de fitness representa su capacidad para competir con otros organismos y sobrevivir. Este individuo también tiene la capacidad de reproducirse, al juntarse con otro individuo. De esta manera, las características que hacen que el fitness sea elevado se pueden ir transmitiendo de generación en generación, mediante la reproducción. La reproducción es la creación de un nuevo individuo a partir de las características de otros dos individuos. A medida que se tienen más individuos, van desapareciendo otros; este fenómeno elimina los individuos con menos fitness, haciendo que solo los que tienen mayor adaptación puedan sobrevivir. De esta manera se protegen las características buenas de los individuos a través de las nuevas generaciones de individuos. Al mantener las características de mayor contribución al fitness, es posible tener una búsqueda en áreas más prometedoras dentro de las soluciones factibles.

Para la creación de un Algoritmo Genético, se debe crear una estructura de solución del problema a tratar (individuo). La estructura debe tener la forma de cromosomas o vectores de información. Así, un vector completo representará los valores de las variables de decisión para el problema. Después, para cada individuo se debe asociar una función de sobrevivencia, la cual representa el costo o beneficio de los valores del genoma de la solución. Luego de terminada la estructura, de debe crear un método para la creación de soluciones iniciales. Las distintas soluciones se agrupan en conjuntos llamados poblaciones de individuos. Mediante métodos iterativos, las distintas soluciones deben ir alterando sus valores, para encontrar mejores valores de su función de sobrevivencia. Los operadores realizan intercambios similares al proceso evolutivo animal. Toman partes del vector y los cambian entre distintos individuos, creando nuevas soluciones. También existen operadores que no requieren un cruzamiento entre varios individuos, como lo es la mutación. La mutación altera secciones dentro de un solo cromosoma, con una cierta probabilidad. Finalmente, mediante el cumplimento de algún criterio, previamente definido, se da término a la creación de nuevas generaciones de soluciones y se obtiene la solución final en base a la mejor función de sobrevivencia.

Otra característica de los algoritmos genéticos es la capacidad de alterar un individuo en alguna generación, sin la necesidad de aplicar una reproducción; la cual es llamada mutación. La mutación altera alguna parte de un individuo, de manera probabilística y sin que necesariamente aumente el fitness.

El poder de los algoritmos genéticos proviene del hecho de que pueden tratar con éxito una gran variedad de problemas provenientes de diferentes aéreas, no quedan atrapados por máximos locales en el caso de problemas de maximización e incluso pueden utilizarse en enfoques híbridos. (Booker et al., 1989).

Se dispone de un ejemplo de pseudocódigo de un algoritmo genético general (Goldberg and Holland, 1988):

Inicializar una población de “u” individuos
Evaluar los “u” individuos según su función de sobrevivencia
Repetir
Seleccionar “p” padres a través de algún proceso
Generar “k” hijos de los “u” padres a través del
Cruzamiento con cierta probabilidad;

Mutar hijos con cierta probabilidad
Evaluar los “k” hijos
Reemplazar la población con “u” individuos padres e hijos
Hasta Criterio de término
Salida Mejor individuo o población encontrada

Además de lo anterior, dentro de los estudios revisados, se encuentran los del tipo comparativo, en donde se ve su eficiencia contra otros métodos de resolución. Estos estudios demuestran que los algoritmos genéticos son capaces de encontrar soluciones factibles en tiempo computacionales deseados (sin embargo, en un mayor tiempo que otros algoritmos). Si bien existen algoritmos que convergen más rápido a soluciones, la evidencia muestra que no siempre dan como resultado soluciones factibles (Colorni et al., 1998), pero aun así, los algoritmos genéticos conservan la cualidad de adaptarse a otros tipos de heurísticas y trabajar en metodologías hibridas para el tratamiento de problema. Esto quiere decir que a futuro, las técnicas evolutivas pueden usarse en combinación con otras para mejorar su eficiencia.

4.2.1. Estructura de la solución

Dada la literatura revisada, la estructura más utilizada para este tipo de problemas corresponde a 2 genomas de soluciones. El primero trata la programación de las asignaturas para los cursos de la institución y el segundo se refiere a la programación de los profesores en cada hora y día dentro de la semana. Cada individuo dentro del algoritmo genético posee estos dos genomas, los cuales dan su fitness correspondiente.

El primer genoma (más adelante referido como "genoma A") tendrá una longitud de λ. Éste se asocia a la variable X del modelo matemático. Dentro del genoma se establecen mediante números enteros las distintas asignaturas para cada curso. El genoma solo presenta números enteros ya que es posible la representación entera sin afectar la factibilidad ni el tiempo de exploración para la solución inicial. Esto se debe a que solo se requiere conocer las asignaturas necesarias para cada curso y luego ingresarlas en un orden aleatorio dentro del vector, teniendo una solución factible desde el punto de vista de la programación curso-asignatura. Sin embargo la factibilidad de los horarios de los profesores no está garantizada.

$$\lambda = \alpha * \beta * \gamma$$

Donde:

α: número de cursos disponibles.

β: número de dias a la semana que pueden existir clases.

γ: número de bloques dentro del dia en lo que pueden existir clases.

El segundo genoma (más adelante referido como "genoma B") tendrá una longitud de π. Éste hace referencia a las horas asignadas para cada profesor en un día, hora y curso específicos.

$$\pi = \alpha * \beta * \gamma * \varepsilon$$

Donde:

α: número de cursos disponibles.

β: número de dias a la semana que pueden existir clases.

γ: número de bloques dentro del dia en lo que pueden existir clases.

ε: número de profesores disponibles.

Cada uno de los dos genomas entrega información de la asignación tanto de las asignaturas como de los profesores. La asignación se hace para cada curso, día y hora disponibles; los cuales se pueden saber, viendo la posición del valor específico dentro del genoma.

Finalmente se creará una estructura similar a la de los profesores, pero que solo será un complemento para el cálculo de la función de fitnes. No se considerará dentro de la solución final. Para esto se creará un "*profesor imaginario*". Tal profesor tiene la particularidad de realizar cualquier asignatura, en cualquier curso. Si no existe ningún profesor real que pueda dictar una clase en un curso, es el *profesor imaginario* el encargado de realizar dicha clase. Sin embargo la utilización de este profesor tiene una penalización dentro de la función de fitnes; cada vez que se asigna una hora para éste profesor, la función de fitnes aumenta en 1 unidad.

4.2.2. Población de individuos y número de generaciones

Si bien son dos parámetros que deben calibrarse, la literatura indica posibles valores para estimarlos. Para el caso de la población, Goldberg da como tamaño propicio un número de individuos igual al tamaño de la representación (Goldberg and Holland, 1988). Luego en el año 1992, Alander confirma los datos de Goldberg e indica que el tamaño suficiente seria entre l y $2l$, donde l es el tamaño de la representación de la solución (Alander, 1992). Dado que el problema cambia, dependiendo de la institución educacional en donde se encuentre el problema de programación horaria, el tamaño de la población también irá cambiando...

Para el caso del número de generaciones. En la literatura existen dos métodos muy utilizados. El primero es dar un número definido de iteraciones hasta terminar la heurística. El segundo es crear un criterio de convergencia en donde se establece un cierto porcentaje; tal porcentaje corresponde al número de genes iguales entre si buscados dentro de las generaciones. Si se cumple el porcentaje, se dice que el algoritmo ha convergido y puede detenerse, independiente del número de generaciones actual (Jong and Alan, 1975). En este caso se usará el criterio de convergencia definido para el 95% de los individuos; esto significa que si existe un 95% de individuos que posean el mismo fitnes, se asume que se ha convergido a la solución final. Sin embargo, en este tipo de problema podemos encontrar múltiples soluciones muy diferentes que pueden alcanzar un mismo valor de fitnes, por lo cual, las demás soluciones seguirán almacenadas en el caso de requerir programaciones horarias opcionales. En caso de no tener convergencia, el algoritmo generará un numero N

de iteraciones como cota superior igual a 8000. El número escogido corresponde a una cota de máximo superior al número de iteraciones que se obtienen con convergencia en problemas de *timetable* similares. De esta manera, de no haber convergencia, el algoritmo encontrará la mejor solución, pero en un tiempo computacional más alto, asegurando así la convergencia forzada.

4.2.3. Operador de cruzamiento

El operador de cruce para este estudio será del tipo *One-point Crossover two parent crossover* (OP2). Para este tipo de cruzamiento, se eligen 2 padres y se generan 2 hijos combinando los genes de ambos padres y seleccionando aleatoriamente que genes heredan cada hijo (Poli and Langdon, 1998). Cada gen heredado por un hijo no puede ser heredado por el otro. Para el caso del estudio, son seleccionados los distintos horarios de cursos para el cruzamiento.

A medida que se va realizando el cruce y creando los hijos, se van acomodando los horarios de cada profesor. Cada vez que una asignatura del gen del padre es heredado por el hijo, también se revisa si es posible copiar el mismo horario del profesor perteneciente al padre. Si no es posible, se busca un profesor reemplazante dentro del conjunto disponible. Finalmente si ningún profesor puede tomar la asignatura heredado, se le asigna al profesor imaginario. De esta manera, no puede quedar ninguna asignatura sin un profesor asignado a ella. En el caso de que la asignación sea para el profesor imaginario, se tiene una penalización en la función de fitnes.

4.2.4. Operador de mutación

Dentro de la mutación de los individuos, existen 3 tipos usados en programación horaria para colegios. Si bien es claro con el estudio bibliográfico que la probabilidad de mutación debe ser muy pequeña, también está la posibilidad de que los valores de dicha probabilidad vayan variando a lo largo de las generaciones de un algoritmo genético. Los tipos de valores de probabilidad son:

- Valor constante: el valor p de la probabilidad para mutar permanece constante a lo largo de las generaciones.
- Agresiva inicial: el valor p de la probabilidad varia a los largo de las generaciones, teniendo una mayor probabilidad de mutar en las primeras generaciones.
- Agresiva final: el valor de la probabilidad varia a los largo de las generaciones, teniendo una mayor probabilidad de mutar en las últimas generaciones.

Sin embargo, para problemas como horarios de colegio, es aconsejable una estrategia multi-mutación, en vez que una única mutación con probabilidad variable (Boers, 2001). Luego, se crean dos tipos de mutaciones para los horarios. El primero intercambia una asignatura de un día y hora específicos y lo intercambia por otra asignatura del mismo curso, dentro del mismo día. El segundo operador realiza también un cambio de dos asignaturas dentro de un mismo curso, pero con la diferencia que el cambio se realiza entre días distintos y horas iguales.

Ambos tipos de mutaciones corresponden al tipo de mutación basado en cambios, el cual se utiliza en representaciones del tipo enteras (Banzhaf, 1990).

Luego de que se establece la mutación, la heurística ve la posibilidad de asignar al mismo profesor para las asignaturas que han sido cambiados. Si no es posible lo anterior, se busca un profesor reemplazante para dictar la asignatura en el nuevo horario. Finalmente si ningún profesor puede dictar la asignatura, luego de establecida la mutación, el *profesor imaginario* es asignado para la asignatura.

4.2.5. Evaluación de fitnes para los individuos

Para el fitnes de cada individuo, se toma en cuenta el horario de los cursos, el horario de los profesores y la utilización de profesores imaginarios.

Para los cursos, el programa revisa la existencia de más de 2 asignaturas idénticos dentro de un mismo día. Cada asignatura adicional da una penalización. Cabe destacar que si existen 2 asignaturas un mismo día, pero separados entre sí, no es penalizado. Luego se busca en

cada día la existencia de pares de asignaturas juntas (uno después del otro); para este caso, se crea un puntaje positivo para el fitnes, valorizando el hecho de que las asignatura idénticos de un día, sean dictados de forma continua, penalizando la existencia de 3 asignaturas dentro del día. Finalmente, se busca dentro de los horarios de los cursos la existencia de vacíos dentro de un mismo día, penalizándolos como infactibilidad.

Para los horarios de los profesores, se establece un puntaje que apunta a la varianza de las horas de todos los profesores. De esta manera el horario elegido no tiene profesores con horas sobrecargadas, en lo posible.

Por último, cada vez que se utiliza a un profesor imaginario, existe una penalización de infactibilidad.

$$Funcion\ de\ adaptación = F_1 - F_2 + 2 * F_3 + F_4 + 2 * F_5$$

$Donde$:

F_1: $cantidad\ de\ ramos\ superior\ a\ 2\ en\ cada\ dia\ para\ la\ solucion.$

F_2: $cantidad\ de\ ramos\ pares\ en\ un\ mismo\ dia\ dentro\ de\ la\ solución$

F_3: $cantidad\ de\ horarios\ vacios\ dentro\ de\ la\ solución$

F_4: $varianza\ de\ las\ horas\ asignadas\ a\ todos\ los\ profesores\ reales$

$$S^2 = \sum_{k=1}^{K} (Y_k - \bar{Y})^2$$

Y_k: $cantidad\ de\ horas\ totales\ asignadas\ para\ el\ profesor\ k$

$\bar{Y}$: $promedio\ de\ horas\ asignadas\ a\ todos\ los\ profesores.$

La varianza penaliza la distancia al promedio elevándola al cuadrado, con lo cual minimiza la posibilidad de que los horarios se sesguen hacia unos profesores.

F_5: $número\ total\ de\ horas\ asignadas\ al\ profesor\ imaginario$

4.2.6. Selección

Para este caso se utilizará una selección proporcional al rango del individuo (Holland, 1975). En este caso se ordenan de mayor a menor, de acuerdo al valor de su función de fitnes. Si bien en algoritmos genéticos el método más usado es el proporcional a la función de fitnes (ruleta), es posible perder la diversidad de las soluciones al tener una convergencia prematura ocasionada por la aparición de superindividuos (Bäck and Hoffmeister, 1991). El uso del método dependiente del rango da la posibilidad de crear una búsqueda más diversa en el espacio y solucionar el problema de convergencia prematura. De esta manera se seleccionan los individuos de rangos extremos, ordenados de menor a mayor según el valor de su función de fitnes.

Dado que la pérdida de soluciones buenas puede aumentar la probabilidad de encontrar solo soluciones infactibles (Pillay, 2014), se propone seleccionar a todos los genes como posibles padres. La función anterior solo selecciona la pareja la cual se creará.

4.2.7. Función de sobrevivencia

Es esta etapa se deben seleccionar los individuos o soluciones que formaran parte de la nueva solución. Para este caso, la mitad de los individuos debe ser descartada, ya que la solución ha crecido al doble de su tamaño al agregar a los descendientes de cada par de padres. Esta operación se realiza luego de las debidas mutaciones y posibles reevaluaciones de la función de adaptación de daca uno. De esta manera, el proceso de sobrevivencia que se utilizara corresponde a la reducción elitista de grado λ. Con λ=tamaño de la solución inicial. La elección se hace respecto al valor de la función de fitnes de cada individuo. Los λ primeros individuos con mejores soluciones son seleccionados para la siguiente generación, mientras que los demás son descartados.

4.2.8. Diagrama para la solución con Algoritmo Genético

Sean:

$A_{ij} = matriz\ binaria\ que\ denota\ si\ el\ profesor\ i\ puede\ realizar\ clases\ en\ el\ curso\ j$
$B_{jk} = matriz\ entera\ que\ denota\ cuantas\ veces\ el\ curso\ j\ requiere\ el\ ramo\ k\ en\ la\ semana$

$C_{ik} = matriz\ binaria\ que\ denota\ si\ el\ profesor\ i\ puede\ dictar\ el\ ramo\ k$
$X_{Mk} = matriz\ binaria\ que\ denota\ la\ asignacion\ del\ ramo\ k\ en\ el\ evento\ M$
$Y_{Mi} = matriz\ binaria\ que\ denota\ la\ asignacion\ del\ profesor\ i\ en\ el\ evento\ M$

$M = evento\ que\ indica\ el\ curso, dia\ y\ hora\ donde\ se\ realizará\ la\ asignacion.$

Solución inicial:

Sea:

$Crear\ memoria\ para\ 2\ grupos\ de\ soluciones\ X\ e\ Y.$
$Cada\ grupo\ debe\ tener\ capacidad\ de\ almacenamiento\ de\ PL\ soluciones.$

$Con\ m = (z, w, t) \quad desde\ (1,1,1)\ hasta\ (Z, W, T)\ hacer:$

(1) $X_{mk} =\ ramo\ k\ para\ el\ \ dia\ z\ de\ la\ semana, curso\ w\ , hora\ t.$

(2)$p = 0$

(3)$Buscar\ desde\ p\ hasta\ ultimo\ valor\ de\ k\ dentro\ de\ matriz\ B_{zk} \quad primer\ valor \neq 0.$

(4)$Buscar\ q\ como\ primer\ valor\ distinto\ de\ cero\ en\ C_{ip} \ \forall j$

$Si\ \forall\ z, con\ w\ y\ t\ actuales, \sum_{Z} Y_{zq} = 0$

{
$Hacer\ B_{zp} = B_{zp} - 1$
$Hacer\ X_{mp} = 1 \quad y \quad luego\ X_{mk} = 0\ \forall k \neq p$
$Hacer\ Y_{mq} = 1\ y\ luego\ Y_{mi} = 0\ \forall i \neq q$
}
$Sino, p = p + 1\ y\ volver\ a\ (3)$

$Si\ t \neq T, t = t + 1\ y\ volver\ a\ (1)$
Sino, t $= 1$, seguir con el algoritmo
$Si\ w \neq W, w = w + 1\ y\ volver\ a\ (1)$
Sino, w $= 1$, seguir con el algoritmo
$Si\ z \neq Z, z = z + 1\ y\ volver\ a\ (1)$

$Hasta\ que\ los\ grupos\ de\ almacenamiento\ no\ sean\ copados, hacer:$
{

Guardar X e Y como solucion factible dentro de los grupos de almacenamiento.

Volver a (1).

}

Los grupos de soluciones creados corresponden a la poblacion inicial del algoritmo.

Repetir el proceso aleatorio hasta completar un grupo de individuos requeridos

Una vez que se ha obtenido la solución inicial, se procede a realizar el algoritmo genético mediante los operadores ya explicados, siguiendo el siguiente esquema (ver Figura 1):

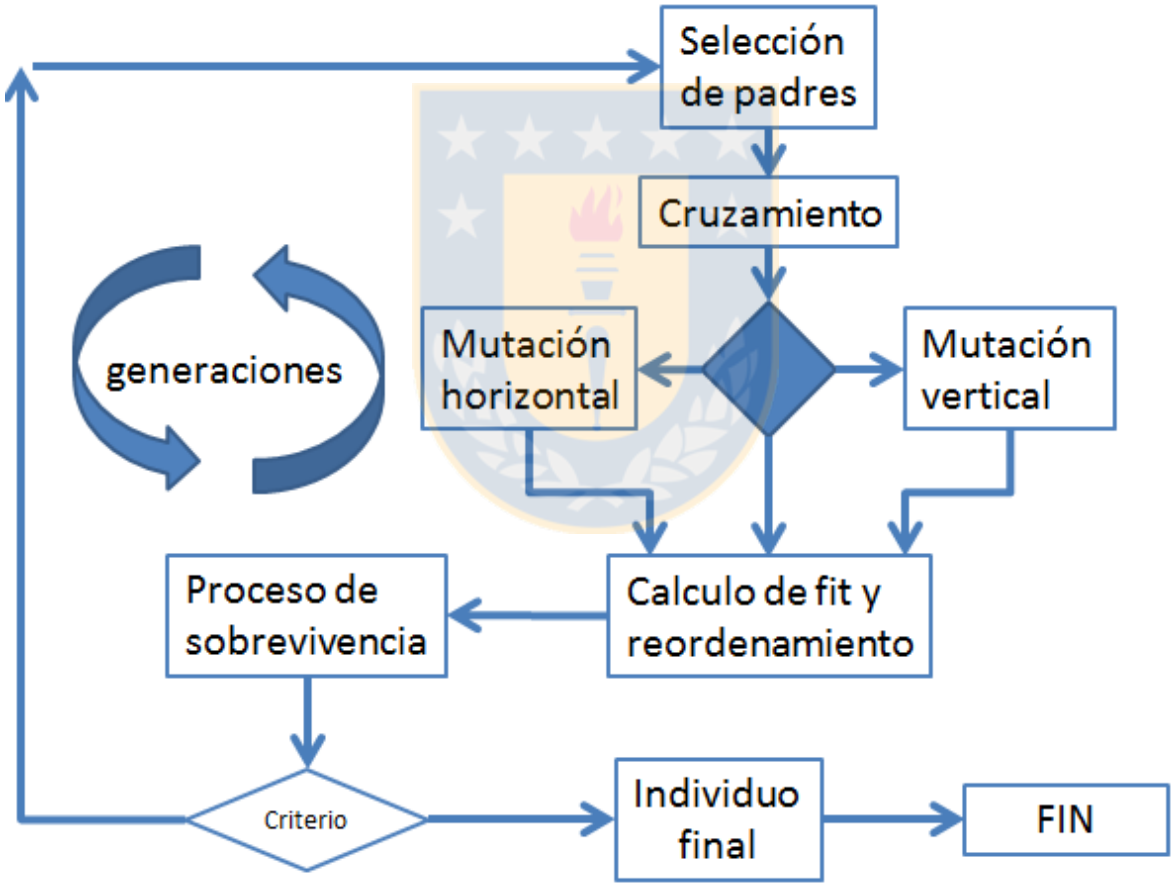

Figura 1: Esquema algoritmo genético

4.2.9. Calibración de parámetros

Dadas las características y procedimientos del algoritmo genético propuesto, es necesario realizar una calibración de los parámetros a utilizar. Esta etapa se calculará los valores de los parámetros de la heurística para los cuales se puede tener una convergencia del algoritmo adecuada.

Los parámetros del algoritmo propuestos a calibrar son:

$$N: tamaño\ de\ la\ poblacion$$
$$P: probabilidad\ de\ mutación$$
$$\beta: porcentaje\ de\ individuos\ suficientes\ para\ aceptar\ convergencia$$

Para estimar los parámetros, lo que se realizó fue un diseño de experimento, lo cual consiste en comparar y estimar el efecto de cambios realizados sobre los parámetros hacia la variable de respuesta, la función de fitnes. Se deben formar grupos de experimentos para eso, en cara uno de los parámetros y un posterior análisis estadístico de los datos. Con lo que finalmente se obtendrán los valores mínimos significativos de ellos. El mejor diseño de experimentos seria del tipo factorial, en este caso se probarían todos los posibles cambios de todos los parámetros para la función de fitnes, sin embargo también denotaría el método más costoso. Entonces si bien es conocido el mejor método, se hace necesario emplear uno que no tenga un costo elevado en tiempo y tamaño de pruebas. Para esto se hace necesario eliminar datos no requeridos como son el espacio de búsqueda no factible dentro de los parámetros, para que así los límites de búsqueda sean más pequeños y siempre dentro de datos significativos.

Este tipo de diseño de experimentos de menor costo son los del tipo exploratorio. Más específicamente, existen experimentos en los cuales solo existen un número determinado de niveles de análisis para los parámetros. Para niveles de 2 o 3 fijos, se denominan factoriales. Para el factorial 2 se utilizan dos niveles de prueba para el parámetro, uno alto y otro bajo. Más específicamente, los diseños factoriales fraccionados permiten obtener conclusiones de una fracción de experimentos, los cuales se seleccionan de manera

estratégica. De esta manera el universo de instancias es reducido drásticamente en conjunto con los límites de pruebas para los parámetros.

Desde el punto de vista de una representación de la metodología, se puede utilizar un conjunto de vectores ortogonales para la representación de los distintos niveles y parámetros en las distintas combinaciones posibles de un experimento. Una forma más condensada es la utilización de solo 2 a 3 factores principales, restando importancia a la interacción de más 3 factores en un mismo grupo de parámetros (Roy, 2010).

Para la calibración de parámetros, si existen múltiples parámetros, se aconseja utilizar un modelo del tipo factorial fraccionada abreviada como 2^{k-p}, donde k es el número de factores para el experimento, $\frac{1}{2^p}$ es el tamaño de la fracción de experimentos y p es la cantidad de interacciones del diseño. Como ejemplo, para 3 factores donde queremos la mitad de los experimentos como tamaño fraccional tendríamos que seria 2^{3-1}. El resultado 4 corresponde al total de corridas de experimentos a realizar.

Suponiendo que tenemos 3 parámetros interactuando, podemos mostrar los distintos niveles de cada uno en la Tabla 7:

Corrida	A	B	C
1	-	-	+
2	+	-	-
3	-	+	-
4	+	+	+

Tabla 7: Diseño factorial fraccionado

Los niveles indican los movimientos en cada dirección del espacio de búsqueda para los parámetros.

Luego podemos dar una representación geométrica de los distintos experimentos (Coy et al., 2001)dependiendo de los valores de los parámetros en cada experimento como se ve en la Figura 2:

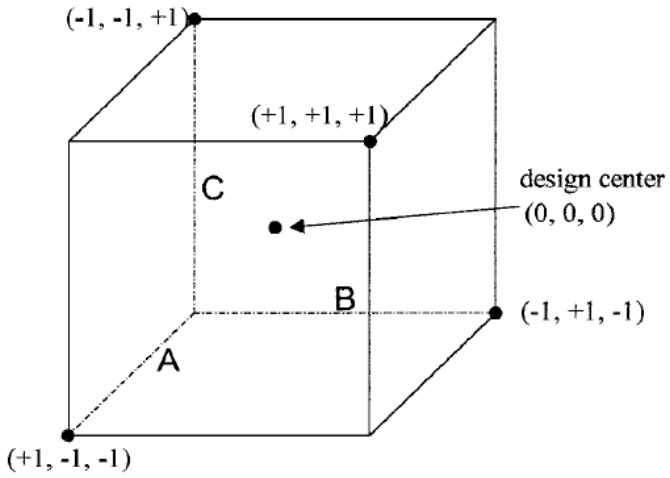

Figura 2: Representación gráfica de un diseño factorial fraccionado

Coy, en 2001, plantea un procedimiento de 4 pasos para llevar a cabo la calibración de los parámetros. Se deben seleccionar los parámetros a calibrar. Luego, del espacio completo de posibles valores de los parámetros, solo se seleccionan los vectores direccionales, es decir, solo un valor para cada dimensión (cada dimensión representa un valor de parámetros). Una vez que se obtienen los vectores direccionales (Figura 2), se debe calcular la magnitud de cada uno. Esto quiere decir que cada número 1 en la matriz, no da el total de movimiento a lo largo de su correspondiente dimensión. Para el cálculo de las amplitudes, se hace un experimento exploratorio para determinar qué valores son los que afectan el tiempo computacional del método utilizado. Una vez obtenidos los vectores y amplitudes, se calcula el gradiente direccional. El gradiente da el verdadero movimiento a lo largo del espacio de valores de los parámetros, para explorar un universo menor al total. Con el gradiente calculado, se puede ir calculando los distintos tiempos computacionales que tendrá el método a utilizar, y determinar cuál valor debe tomar cada parámetro para optimizar el tiempo de ejecución. Los pasos detallados se encuentran a continuación:

Paso 1: se selecciona una fracción del set completo de instancias del problema para llevar a cabo los experimentos.

Paso 2: se selecciona el nivel de inicio de cada parámetro a calibrar, el rango de variación y el aumento o disminución de cada uno.

Paso 3: Seleccionar los buenos parámetros para cada instancia del problema utilizando el diseño de experimentos.

Paso 4: Combinar los valores obtenidos en el paso 3 para obtener valores de parámetros con las mejores calidades.

Dentro de los pasos, el 3 es el que contiene un mayor análisis estadístico, por lo cual se desglosa en los pasos que se indican a continuación en la Figura 3 :

Paso 3.1	Generar un diseño de experimentos factorial
Paso 3.2	Para cada problema en el grupo de análisis, repetir paso 3.3 a 3.9
Paso 3.3	Calcular el vector de parámetros asociado con cada fila del diseño de experimentos factorial
Paso 3.4	Realizar 5 ejecusiones del algoritmo, para cada vector de parámetros calculados en el paso 3.3
Paso 3.5	Ajustar un modelo lineal usando el promedio de la función de costos para cada grupo de 5 ejecusiones, como variable dependiente
Paso 3.6	Encontrar el camino de descenso sobre la superficie de respuesta determinada en el paso 3.5
Paso 3.7	Hacer hasta que todos los parámetros estadísticamente significativos hayan alcanzado el límite de la región experimental o un nuevo mínimo no ha sido encontrado en dos pasos completos
Paso 3.8	Calcular el vector de parámetros asociado con un cuarto de paso sobre el camino de descenso
Paso 3.9	Realizar 5 ejecusiones a partir de la combianación de parámetros obtenida en el paso 3.8 y determinar el promedio

Figura 3: Desglose paso 3 (Coy el al. 2001)

CAPITULO V Resultados

5.1. Resultados de calibración de parámetros

Paso 1

Para la selección de las instancias de prueba se debe tener en cuenta que mientras mayor es el número de instancias elegidas, mayor será la precisión del método, sin embargo el costo aumenta en conjunto. Lo recomendable es utilizar instancias con diversas estructuras del problema. De esta manera no será necesario requerir un número elevado de experimentos.

Teniendo en cuenta lo anterior, se utilizaron 5 instancias distintas teóricas (ver Tabla 8), de la literatura y también del caso de requerimientos de Chile en sus instituciones.

Instancia	Descripción
hdtt4	Instancia de base de datos de problemas tipo timetable no pertenecientes a chile. 4 profesores, 20 cursos, 4 clases, 4 salas.
C-chile	Instancia de prueba para Lingo de tamaño grande
Op-2-chile	Simulación de colegio real, 2 cursos por grado y 50 profesores
Op-3-chile	Simulación de colegio real, 3 cursos por grado y 60 profesores
Alemán-114	Instancia de un colegio alemán adaptada. 114 profesores, 43 clases.

Tabla 8: Instancias usadas para calibración

Paso 2

Para la selección del punto de inicio es recomendable poseer experiencia en el comportamiento de los problemas. En la literatura podemos encontrar información que apunta a las aproximaciones de los puntos de inicios de cada parámetro, sabiendo que se puso obtener una buena solución con tales valores. Sin embargo, los datos de parámetros de la literatura son estáticos, por lo que determinar los puntos de movimiento de ellos no es posible con solo tener sus valores. Para lo anterior, se realizó un análisis de prueba y error.

Para cada valor estático de los parámetros, se procedió a variar los demás hasta ver cuando se presentaba un comportamiento no apropiado para la metaheurística; de esta manera se puede obtener un valor aproximado de la variación de tales parámetros.

La Tabla 9 muestra los valores centrales tomados de la literatura y los limites, los cuales son los valores máximos y mínimos que deben tomar sin que se afecte a la metaheuristica. El valor significativo para afectar el resultado se tomara como 2 minutos de retraso en comparación a la ejecución normal del programa. Esto quiere decir que si el tiempo del algoritmo sufre un cambio de 2 minutos en su tiempo de ejecución, luego de realizar un cambio en algún parámetro, ese cambio realizado se considera significativo.

Parámetro	Mínimo	Máximo	Valor central	Incremento
P	0,005	0,07	0,01	0,005
β	0,04	0,16	0,05	0,1
N	20	∞	50	10

Tabla 9: Límites e incremento de parámetros

Paso 3

Para el paso 3 se requiere calcular el tamaño del diseño de experimento. En este caso se tienen 3 parámetros a calibrar y se desea la mitad de los experimentos que requiere un diseño factorial completo. Para este caso se tendría que serian $2^{3-1} = 4$. Se necesitan corridas de experimentos. A continuación se muestra la Tabla 10, que muestra el espacio de búsqueda necesario para los experimentos. Dependiendo del valor en la tabla, se podrá realizar un aumento o disminución en una amplitud igual al incremento (calculado anteriormente) para cada experimento.

Corrida	P	β	N
1	-1	-1	1
2	1	-1	-1
3	-1	1	-1
4	1	1	1

Tabla 10: Diseño del experimento para calibración

El estudio fue realizado utilizando el software de análisis estadístico de datos Statgraphics Centurion versión 15.2.06. La Tabla 111 muestra los valores de cada parámetro en cada instancia. Los valores ceros son para los parámetros que no son significativos a un nivel de confianza del 95%. Es decir, cada parámetro que tenga asociado un valor p superior a 0,05 no es significativo.

Instancia	R^2 ajustado	Constante	P	β	N	Valor -p
hdtt4	0,49351	58	4,6	97,42	0	0,015
C-chile	0,58262	49,039	0	-7,632	-8,755	0,019
Op-2-chile	0,53514	-18,425	-18,294	-7,61	-25,689	0,0296
Op-3-chile	0,53183	33,261	-16,462	0	0	0,0081
Alemán-114	0,50374	46,703	-20,946	-9,754	0	0,0022

Tabla 11: Resultados de los experimentos para los parámetros

A continuación se debe calcular el gradiente de descenso, ya que es un problema de mínimo. Para el cálculo del gradiente, se deben estimar todos los parámetros del vector de movimiento.

Sea b_j *el gradiante de la forma* (b_1, b_2, b_3)*ya que se tienen solo* 3 *parámetros*.

Para cada problema se calcula el gradiente de la siguiente forma:

- Se selecciona el parámetro de mayor valor absoluto b_x
- Cada parámetro es dividido por b_x y multiplicado por su correspondiente incremento.
- Se obtiene un nuevo vector b_j el cual es el gradiente de movimiento.
- Desde los valores centrales de todos los parámetros, se empiezan a cambiar los valores de parámetros, calculando el costo y tiempo en cada paso.

- En caso de alcanzar los límites establecidos, se mantienen constantes.

Los valores de los 5 gradientes se muestran a continuación en la Tabla 12:

Instancia	P	β	N
hdtt4	0,04721823	1	0
C-chile	0	0,87173044	1
Op-2-chile	0,7121336	0,29623574	1
Op-3-chile	1	0	0
Alemán-114	1	0,46567364	0

Tabla 12: Gradientes de avance para las instancias

En la Figura 4 se puede ver que es posible alcanzar una solución de costo cero (infactibilidades) para la instancia de tamaño pequeño. Sin embargo, el costo en tiempo para alcanzar dicha solución se eleva incluso casi de manera exponencial en los tramos centrales.

Para la Figura 5, si bien no es un caso que asemeje la realizas completamente, se observa que en instancias de tamaño pequeño, la función de costos es altamente sensible a los cambios del tamaño de la población, con lo cual se podría realizar algún enfoque distinto frente a colegios pequeños o de número de cursos reducidos, en comparación a los de tamaño grande.

Tanto la Figura 6, como la Figura 7, representan instancias muy parecidas, ya que son los horarios generados de colegios con normativa chilena. Se puede apreciar que ambos casos presentan cambios mínimos de tiempo una vez que se encuentran los pasos para los parámetros ideales. Ambos son altamente sensibles cuando el parámetro de mutación va en aumento, lo cual hace pensar que para casos de Chile, el usar una mutación adecuada puede incluso suplir un criterio de termino no definido con exactitud. Es posible afirmar también que el parámetro de población no ha tenido mucho efecto en la disminución del costo, ya que haciendo gradientes distintos para la población en ambas instancias, la tendencia de disminuir los costos significativamente se mantienen.

En la Figura 8 se observa una convergencia mucho más rápida que los demás casos. En parte se debe al tamaño pequeño de la instancia

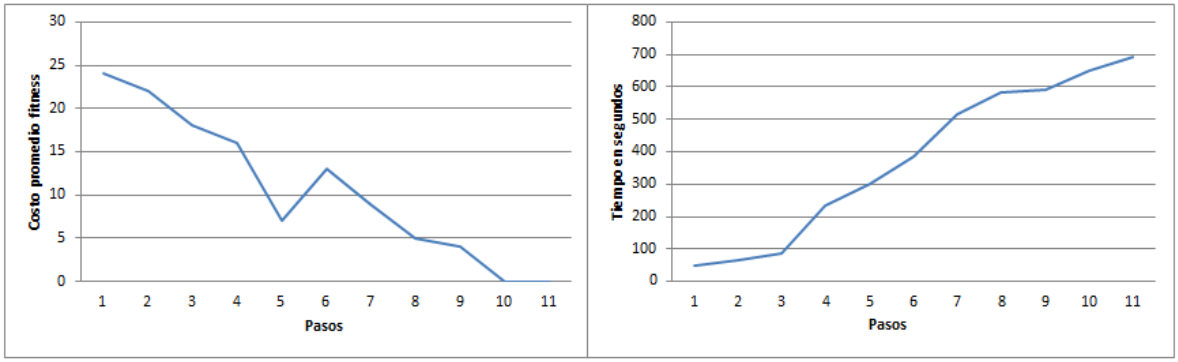

Figura 4: Pasos para instancia 1

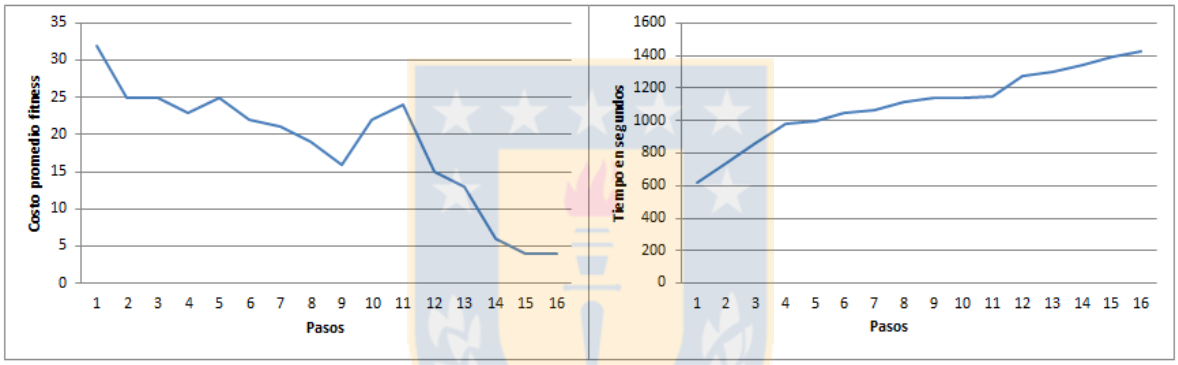

Figura 5: Pasos para instancia 2

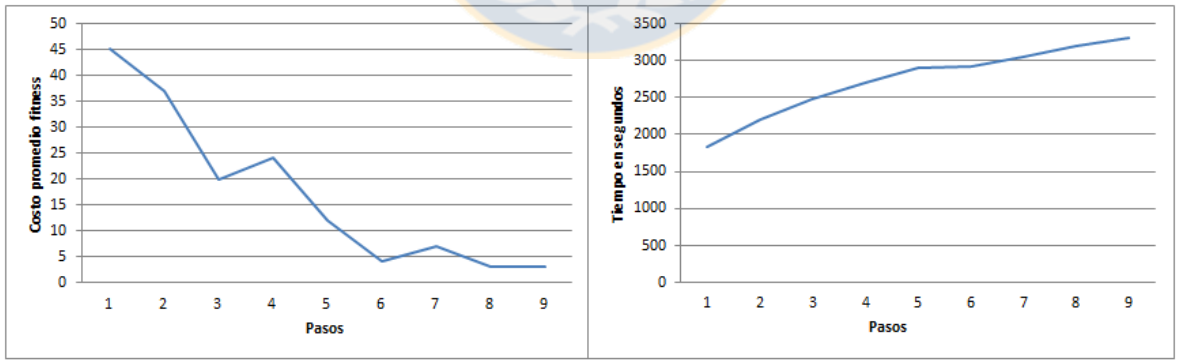

Figura 6: Pasos para instancia 3

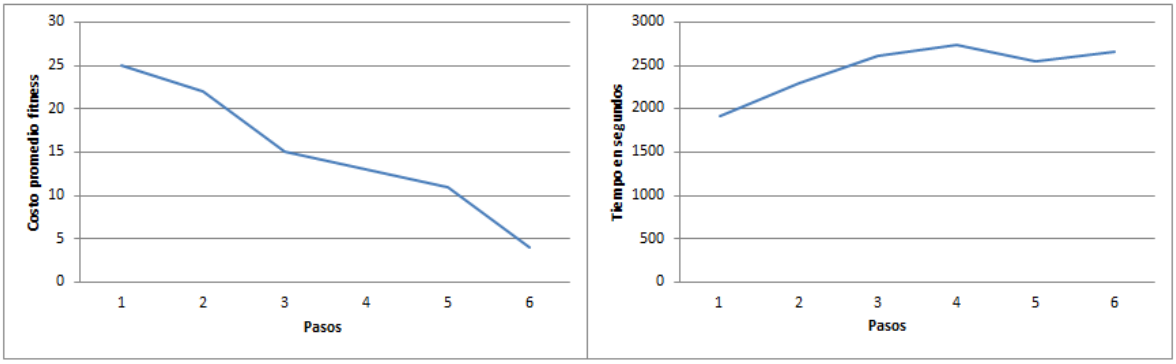

Figura 7: Pasos para instancia 4

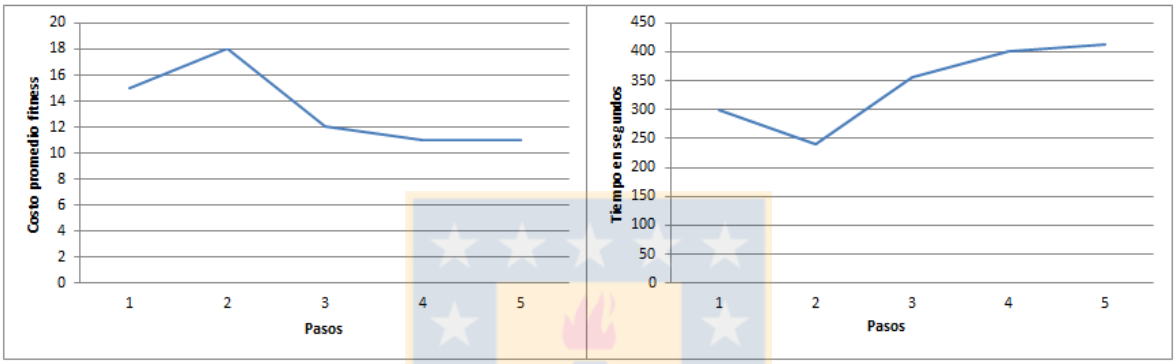

Figura 8: Pasos para instancia 5

Paso 4

En el último paso se deben calcular los promedios de los parámetros obtenidos para cada instancia tratada. Los valores finales son lo que se utilizarán para ejecutar la metaheurística. A continuación se muestran en la Tabla 13 los valores para cada uno:

P	β	N
1,63%	7,31%	130

Tabla 13: Parámetros calibrados

5.2. Interfaz de usuario

Para la creación de instancias ficticias y la recopilación de datos reales; además, adecuando la herramienta para usuarios que no tienen conocimiento sobre optimización, se diseñó una herramienta computacional con instrucciones detalladas y una interfaz agradable al usuario, para poder capturar los datos necesarios para resolver problemas de tamaño reducido mediante solver LINGO y problemas de tamaño grande, utilizando la heurística diseñada. A continuación se detallan los pasos para la obtención de los datos.

- Creación de cursos existentes:

  En esta etapa, el usuario puede elegir qué cursos existen en la institución educacional. Siguiendo la legislación de Chile y para el estudio objetivo, se dan las opciones de elegir los grados desde primero básico hasta cuarto medio de educación (ver Figura 9). Adicionalmente, si para cada grado existen cursos en paralelo, es posible ingresas los datos, separando los cursos del mismo grado, asignándoles una letra de identificación. Además, es posible saber la cantidad de asignaturas utilizados con el botón de comando visto en la figura. Los datos son guardados en una planilla Excel en forma de vector.

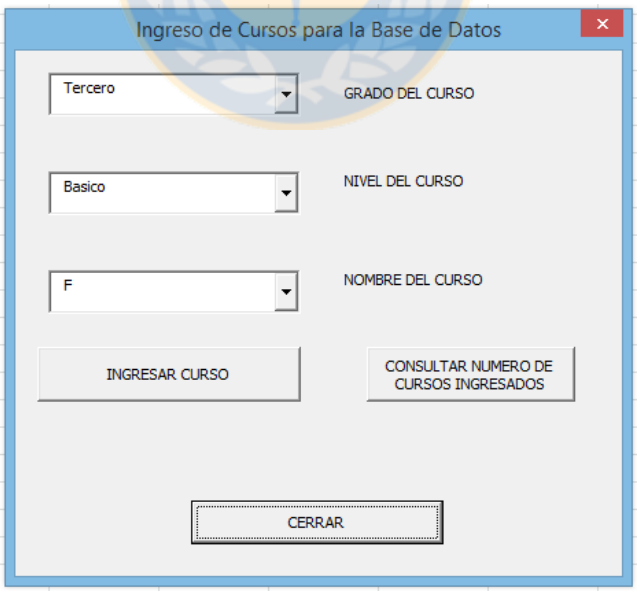

Figura 9: Ventana creación de cursos

- Recopilación de asignaturas que pueden ser dictados en la institución:

En esta sección, el usuario ingresa el nombre de las asignaturas existentes en la institución. Para esto hay que tener en cuenta que no interesa si la asignatura tiene un mismo nombre y es operable para más de algún grado. Ejemplo: Si existe matemática, y es utilizada para todos los cursos de la institución, solo es necesario agregar la asignatura una sola vez. Más adelante se hacen distinciones de asignaturas según sus grados. Además el usuario debe ingresar los posibles formatos de bloques de horarios en los cuales se puede dictar la asignatura ingresada. La Figura 10 muestra la pantalla para la captura de los datos.

Figura 10: Ventana de creación de asignaturas

- Asignación de asignaturas para cada curso existente:

En esta etapa se seleccionan las asignaturas que requieren cada curso dentro de la institución. El usuario selecciona un curso en particular, luego la asignatura que requieren y a continuación se indican las horas semanales que requiere dicha asignatura en el curso seleccionado. Opcionalmente, el usuario puede dar una carga horaria igual a la de algún curso ingresado con anterioridad, evitando así las repeticiones excesivas (ver Figura 11).

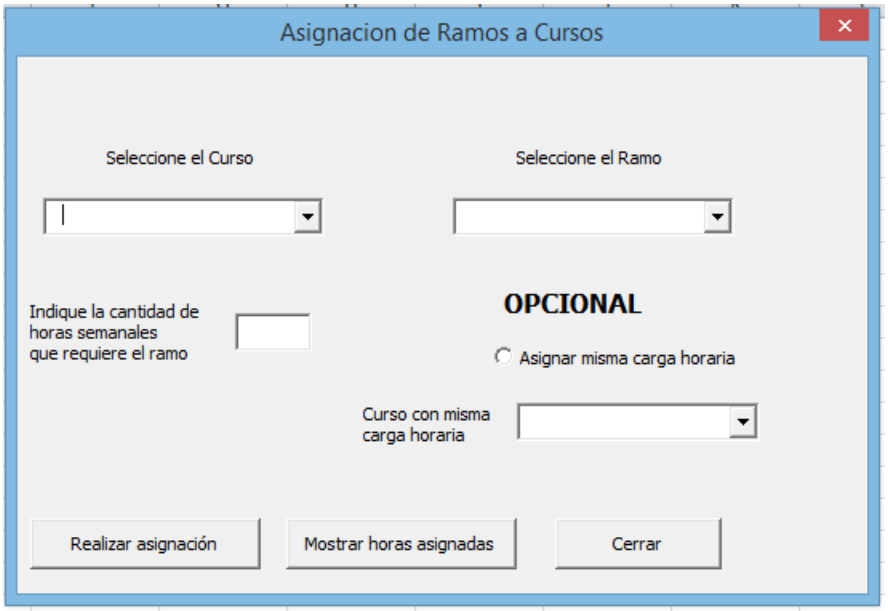

Figura 11: Ventana asignación asignaturas curso

- Ingreso de profesores disponibles para hacer clases por la institución:

En esta ventana (ver Figura 12) se pueden ingresar los profesores que existen en la institución. Los nombres ingresados serán utilizados para establecer el horario. Si así lo desea el usuario, puede ingresar solo un número o seudónimo para proteger la identidad de los profesionales. Adicionalmente se debe ingresar el número de horas por la cual se ha contratado el profesor. El software tomara en cuenta tal número de horas como el máximo de tiempo que puede realizar clases un profesor dentro de la institución.

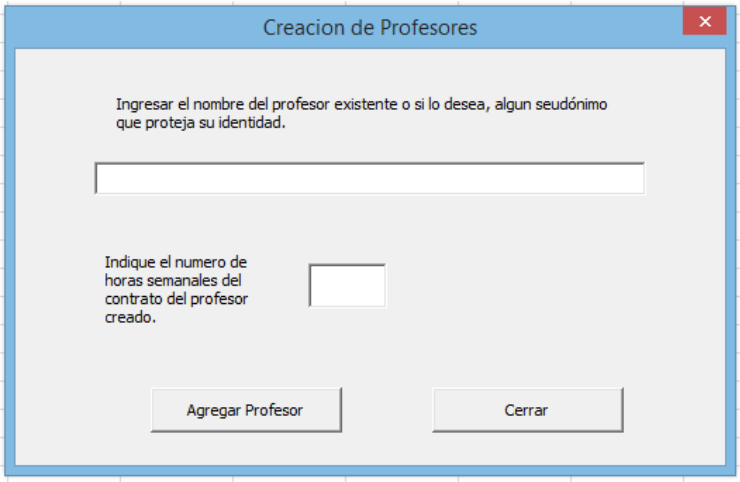

Figura 12: Ventana creación de profesores

- Asignación de asignaturas que puede dictar cada profesor:

La Figura 13 que se muestra a continuación es la última ventana encargada de ingresar datos de la institución educacional. En esta etapa se realizaran las asignaciones de las asignaturas que pueden dictar cada profesor y los grados en los cuales puede tomar dicha asignatura. Para lo anterior, el usuario debe seleccionar un profesor y una asignatura para realizar la conexión. Una vez ingresados los dos datos, se debe elegir como ingresar los grados en los cuales se den dicha conexión entre el profesor y la asignatura; para esto se tienen dos opciones. En la primera opción el usuario puede dar un conjunto de grados en los cuales el profesor puede dictar la asignatura elegida. En la segunda opción solo se elige un grado en el cual el profesor puede dictar la asignatura elegida.

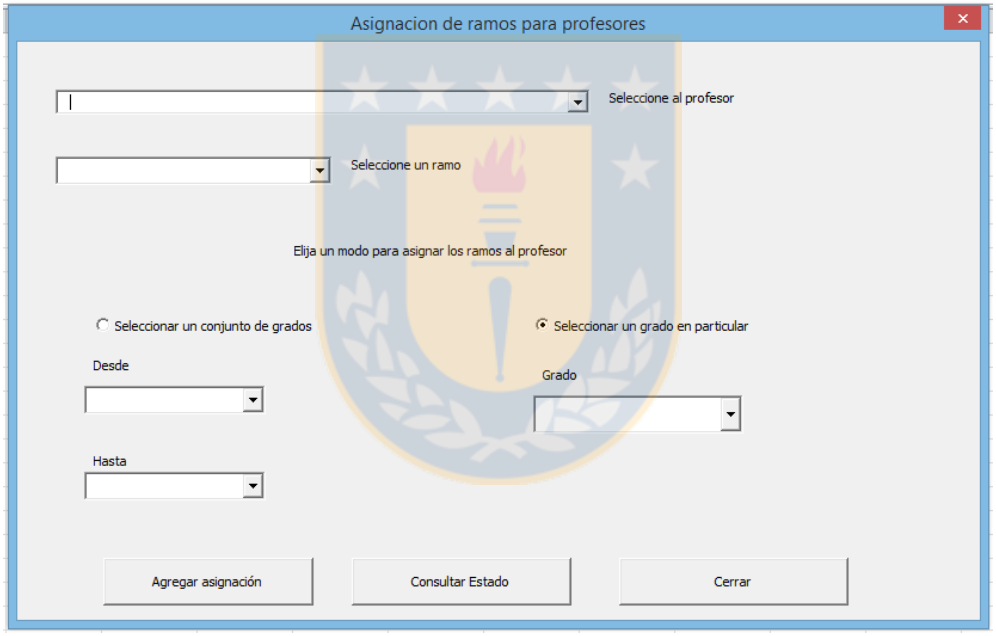

Figura 13: Ventana asignación profesor con asignatura

- Preparación de datos para Lingo y Metaheurística

Finalmente, cuando se han introducido todos los datos necesarios, Se debe decidir que método de resolución usar, dependiendo del tamaño del problema.

Para problemas pequeños, la solución puede ser exacta y resolverse mediante LINGO. En este caso, el programa de visual basic dará nombre a los distintos rangos necesarios para

que sean identificables por Lingo como los parámetros del problema de horarios. Luego, desde la ventana de Lingo, mediante el archivo con el modelo ya programado, se debe resolver el problema. Lingo dará la solución a la que converge y exportará dicha solución a Excel, para que pueda ser leída correctamente como una tabla de horario.

Para el caso de utilizar la metaheurística, el archivo debe guardarse de tal manera de tener las matrices necesarias para el algoritmo, como archivo texto delimitado por tabulaciones (de esta manera la lectura es aceptable para C++). Finalmente, mediante el programa creado, se debe poner en marcha el algoritmo y esperar que de la solución. La solución puede exportarse a Excel para su correcta lectura como tabla de horario.

5.3. Resultados de LINGO y metaheurística

La Tabla 14 muestra los resultados ordenados según los tipos de instancias que se utilizaron para la prueba. Todos los resultados a continuación se dieron con parámetros calibrados. Para los resultados obtenidos, tanto en Lingo como en la heurística utilizada, los valores de la varianza son eliminados después de obtener el resultado final. Lo anterior es necesario para crear una adecuada comparación con los resultados de los otros algoritmos, ya que no consideran varianza, solo infactibilidades.

Grupo 1: Instancias obtenidas de internet de la librería de problemas J.E. Beasley. Fuente: http://people.brunel.ac.uk/~mastjjb/jeb/jeb.html

Grupo 2: Instancias teóricas creadas para la prueba de Lingo v/s Metaheurística. Todas las instancias cumplen con la norma chilena de educación en requerimiento de clases y tiempo máximo asignado.

Grupo 3: Instancias teóricas creadas para las pruebas de la metaheurística. Todas cumplen con la norma chilena de educación en requerimiento de clases y tiempo máximo asignado. Las instancias emulan el tamaño de un colegio real.

Grupo 4: instancias obtenidas de la librería de la Universidad de Twente. Todas pertenecen a colegios reales fuera de Chile. Fuente http://www.utwente.nl/ctit/hstt/

		Características			Mejor solución			Tiempos en segundos	
	Instancia	Profesores	Cursos	Clases	Heurística	Lingo	Optimo	Heurística	Lingo
Grupo 1	hdtt4	4	20	120	3	-	0	300	-
	hdtt5	5	20	150	2	-	0	353	-
	hdtt6	6	20	180	0	-	0	401	-
	hdtt7	7	20	210	3	-	0	415	-
	hdtt8	8	20	240	15	-	0	459	-
Grupo 2	a1	10	1	5	0	0	0	204	4
	a2	15	2	10	0	0	0	217	4
	b1	20	2	10	0	0	0	233	6
	b2	20	2	15	0	0	0	245	15
	b3	30	14	24	2	-	-	742	-
	c	40	16	48	2	-	-	806	-
Grupo 3	op-2-chile	40	24	376	2	-	-	1200	-
	op-3-chile	44	48	568	2	-	-	1575	-
	op-4-chile	44	48	890	3	-	-	1703	-
Grupo 4	BGHS98	56	30	1564	510	-	386	1984	-
	BR-SM-00	23	12	300	72	-	51	2077	-
	AU TES99	37	13	806	200	-	125	2131	-
	FI-PB-98	46	31	854	2	-	0	2403	-

Tabla 14: Instancias utilizadas divididas por grupos

CAPITULO VI Conclusiones

En comparación con otros tipos de problemas de optimización tratado con metaheuristicas, el tiempo computacional para la resolución es mucho más alto. Sin embargo esto ocurre porque la representación del problema es de un tamaño mayor al de los problemas normales.

Si bien Lingo es una herramienta confiable debido a su gran potencial de resolución de problemas de optimización y su interacción con bases de datos, tiene sus limitaciones a la hora de resolver problemas de tamaño real de programación de horarios. Para este tipo de problemas se hace necesario el uso de herramientas heurísticas debido a su complejidad (cabe recordar que este tipo de problemas es definido como NP-Hard según la literatura).

Si bien, no todas las instancias fueron resueltas encontrando los valores óptimos; con este estudio se pudo demostrar que es posible encontrar soluciones factibles para el problema de programación de horarios, en contraste con los estudios comparativos de metaheuristicas. Además es posible resolver el problema en tiempo computacionales deseables.

Es posible resolver problemas reales dentro de colegios de Chile con el método obtenido, ya que se ajusta a las normas y leyes vigentes. El cambio de institución, al igual que el cambio en instancias de la literatura, solo requiere agregar o quitar parámetros de costos en la función de fitnes.

Es una buena estrategia utilizar métodos de resolución que utilicen poblaciones o grupos de soluciones, ya que el problema puede poseer múltiples puntos de buenas soluciones que satisfacen las restricciones. Esto da también paso a dar como solución, un conjunto de horarios que satisfacen los requerimientos, lo cual aumenta la probabilidad de aceptación por parte de los usuarios en caso de no haber incluido parámetros dentro del problema.

Para el caso de la configuración de parámetros para la resolución de problemas del tipo *timetabling*, es necesario hacer un estudio del tamaño del problema, ya que para tamaños pequeños es el tamaño de población la que afecta más a la solución, mientras que para colegios grandes, el parámetro de mutación afecta en mayor medida el resultado final.

Como trabajo futuro es posible crear instancias de colegios reales mediante la interfaz creada. De esta manera puede hacerse posible la incursión de investigaciones internacionales en los casos nacionales.

REFERENCIAS

1. Abramson, D. (1991). Constructing School Timetables Using Simulated Annealing: Sequential and Parallel Algorithms. Manag. Sci. *37*, 98–113.

2. Alander, J.T. (1992). On optimal population size of genetic algorithms. In CompEuro '92 . "Computer Systems and Software Engineering",Proceedings., pp. 65–70.

3. Alvarez-Valdes, R., Crespo, E., and Tamarit, J.M. (2002). Design and implementation of a course scheduling system using Tabu Search. Eur. J. Oper. Res. *137*, 512–523.

4. Avella, P., D'Auria, B., Salerno, S., and Vasil'ev, I. (2007). A computational study of local search algorithms for Italian high-school timetabling. J. Heuristics *13*, 543–556.

5. Bäck, T., and Hoffmeister, F. (1991). Extended Selection Mechanisms in Genetic Algorithms. (Morgan Kaufmann), pp. 92–99.

6. Banzhaf, D.W. (1990). The "molecular" traveling salesman. Biol. Cybern. *64*, 7–14.

7. Beligiannis, G.N., Moschopoulos, C.N., Kaperonis, G.P., and Likothanassis, S.D. (2008). Applying evolutionary computation to the school timetabling problem: The Greek case. Comput. Oper. Res. *35*, 1265–1280.

8. Birbas, T., Daskalaki, S., and Housos, E. (2009). School timetabling for quality student and teacher schedules. J. Sched. *12*, 177–197.

9. Boers, E.J.W. (2001). Applications of Evolutionary Computing: EvoWorkshops 2001: EvoCOP, EvoFlight, EvoIASP, EvoLearn, and EvoSTIM, Como, Italy, April 18-20, 2001 Proceedings (Springer Science & Business Media).

10. Booker, L.B., Goldberg, D.E., and Holland, J.H. (1989). Classifier systems and genetic algorithms. Artif. Intell. *40*, 235–282.

11. Burke, E.K., and Petrovic, S. (2002). Recent research directions in automated timetabling. Eur. J. Oper. Res. *140*, 266–280.

12. Carrasco, M.P., and Pato, M.V. (2004). A comparison of discrete and continuous neural network approaches to solve the class/teacher timetabling problem. Eur. J. Oper. Res. *153*, 65–79.

13. Carter, M.W., Laporte, G., and Lee, S.Y. (1996). Examination Timetabling: Algorithmic Strategies and Applications. J. Oper. Res. Soc. *47*, 373.

14. Colorni, A., Dorigo, M., and Maniezzo, V. (1998). Metaheuristics for High School Timetabling. Comput. Optim. Appl. *9*, 275–298.

15. Coy, S.P., Golden, B.L., Runger, G.C., and Wasil, E.A. (2001). Using Experimental Design to Find Effective Parameter Settings for Heuristics. J. Heuristics *7*, 77–97.

16. Even, S., Itai, A., and Shamir, A. (1975). On the complexity of time table and multi-commodity flow problems. In , 16th Annual Symposium on Foundations of Computer Science, 1975, pp. 184–193.

17. Goldberg, D.E., and Holland, J.H. (1988). Genetic Algorithms and. Mach. Learn. *3*, 95–99.

18. Hernandez, R., Miranda, J., Rey, P. (2008). Programación de horarios de clases y asignación de salas para la facultad de ingeniería de la Universidad Diego Portales mediante un enfoque de programación entera. Revista de Ingenieria de Sistemas. XXII. 121-141.

19. Holland, J.H. (1975). Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence (Oxford, England: U Michigan Press).

20. Jong, D., and Alan, K. (1975). Analysis of the behavior of a class of genetic adaptive systems.

21. Kristiansen, S., Sørensen, M., and Stidsen, T.R. (2011). Elective course planning. Eur. J. Oper. Res. *215*, 713–720.

22. Lewis, R. (2008). A survey of metaheuristic-based techniques for University Timetabling problems. Spectr. *30*, 167–190.

23. Lü, Z., and Hao, J.-K. (2010). Adaptive Tabu Search for course timetabling. Eur. J. Oper. Res. *200*, 235–244.

24. Ministerio de Educación. (2013). Currículum nacional de instituciones educacionales de educación básica y media. Recuperado de: http://www.mineduc.cl/index5_int.php?id_portal=47&id_contenido=17116&id_seccion=3264&c=346

25. Naji Azimi, Z. (2005). Hybrid heuristics for examination timetabling problem. Appl. Math. Comput. *163*, 705–733.

26. Pillay, N. (2014). A survey of school timetabling research. Ann. Oper. Res. *218*, 261–293.

27. Poli, R., and Langdon, W.B. (1998). Schema Theory for Genetic Programming with One-Point Crossover and Point Mutation. Evol. Comput. *6*, 231–252.

28. Qaurooni, D., and Akbarzadeh-T, M.-R. (2013). Course timetabling using evolutionary operators. Appl. Soft Comput. *13*, 2504–2514.

29. Roy, R.K. (2010). A Primer on the Taguchi Method, Second Edition (Society of Manufacturing Engineers).

30. Sabar, N.R., Ayob, M., Kendall, G., and Qu, R. (2012). A honey-bee mating optimization algorithm for educational timetabling problems. Eur. J. Oper. Res. *216*, 533–543.

31. Schaerf, A. (1999). A Survey of Automated Timetabling. Artif. Intell. Rev. *13*, 87–127.

32. Suárez, V.F., Guerrero, Á., and Castrillón, O.D. (2013). Programación de Horarios Escolares basados en Ritmos Cognitivos usando un Algoritmo Genético de Clasificación No-dominada, NSGA-II. Inf. Tecnológica *24*, 103–114.

33. Thepphakorn, T., Pongcharoen, P., and Hicks, C. (2014). An ant colony based timetabling tool. Int. J. Prod. Econ. *149*, 131–144.

34. Valouxis, C., and Housos, E. (2003). Constraint programming approach for school timetabling. Comput. Oper. Res. *30*, 1555–1572.

35. Wright, M. (1996). School Timetabling Using Heuristic Search. J. Oper. Res. Soc. *47*, 347.

36. Zhang, D., Liu, Y., M'Hallah, R., and Leung, S.C.H. (2010). A simulated annealing with a new neighborhood structure based algorithm for high school timetabling problems. Eur. J. Oper. Res. *203*, 550–558.

ANEXOS

Anexo 1: Resultados de ANOVA del diseño de experimento

hdtt4		Error	Estadístico	
Parámetro	Estimación	Estándar	T	Valor-P
Cte	58	4,73743	-0,283	0,001
mutacion	4,6	3,18350	0,9314	0,037
beta	97,42	82,44591	1,779	0,139
poblacion	-17,342	-13,53814	0,0929	0,533

Fuente	Suma de Cuadrados	Cuadrado Medio	Razón-F	Valor-P
Modelo	6,88212976	2,29404325	6,31372823	0,015
Residuo	1,09002629	0,3633421		
Total	7,97215605			

R cuadrado	0,6728
R cuadrado ajustado	0,49351
Durbin-Wats	1,82390

Tabla 15: MANOVA para instancia 1

C-chile		Error	Estadístico	
Parámetro	Estimación	Estándar	T	Valor-P
Cte	49,039	22,46709	-3,0431	0,016
mutacion	0,339	0,17943	-2,2146	0,0541
beta	-7,632	-5,86556	1,5657	0,03
poblacion	-8,755	-3,29868	1,7545	0,0483

Fuente	Suma de Cuadrados	Cuadrado Medio	Razón-F	Valor-P
Modelo	54,7855778	18,2618593	5,32257788	0,019
Residuo	10,2930533	3,43101777		
Total	65,0786311			

R cuadrado	0,74092
R cuadrado ajustado	0,58262
Durbin-Wats	2,1078

Tabla 16: MANOVA para instancia 2

Op-2-chile		Error	Estadístico	
Parámetro	Estimación	Estándar	T	Valor-P
Cte	-18,425	-10,97159	2,1438	0,0278
mutacion	-18,294	-15,06892	-3,9635	0,0495
beta	-7,61	-5,83948	3,3567	0,0159
poblacion	-25,689	13,62476	-3,5987	0,0215

Fuente	Suma de Cuadrados	Cuadrado Medio	Razón-F	Valor-P
Modelo	-8,28376849	-2,76125616	6,70609376	0,0296
Residuo	-1,23525987	-0,41175329		
Total	-9,51902836			

R cuadrado	0,71016
R cuadrado ajustado	0,53514
Durbin-Wats	2,7677

Tabla 17: MANOVA para instancia 3

Op-3-chile		Error	Estadístico	
Parámetro	Estimación	Estándar	T	Valor-P
Cte	33,261	17,70376	-0,9674	0,0447
mutacion	-16,462	-7,25026	0,176	0,0461
beta	77,431	4,09301	0,7658	0,0532
poblacion	-20,139	-6,27829	-0,0072	0,1146

Fuente	Suma de Cuadrados	Cuadrado Medio	Razón-F	Valor-P
Modelo	24,5012839	8,16709463	7,69898248	0,0081
Residuo	3,18240546	1,06080182		
Total	27,6836893			

R cuadrado	0,70873
R cuadrado ajustado	0,54183
Durbin-Wats	1,9317

Tabla 18: MANOVA para instancia 4

Alemán-114		Error	Estadístico	
Parámetro	Estimación	Estándar	T	Valor-P
Cte	46,703	15,09131	-2,28	0,0215
mutacion	-20,946	13,72097	3,7248	0,0418
beta	-9,754	28,59297	2,7044	0,0239
poblacion	37,526	35,21274	-0,0092	0,0245

Fuente	Suma de Cuadrados	Cuadrado Medio	Razón-F	Valor-P
Modelo	52,0993971	17,3664657	7,23300295	0,0022
Residuo	7,20301062	2,40100354		
Total	59,3024077			

R cuadrado	0,66832
R cuadrado ajustado	0,50374
Durbin-Wats	2,4862

Tabla 19: MANOVA para instancia 5

Anexo 2: Tablas del diseño de experimento

Tablas de valores del diseño de experimentos para los distintos pasos de las 5 instancias.

paso	costo	tiempo
1	24	51
2	22	67
3	18	87
4	16	235
5	7	303
6	13	387
7	9	516
8	5	583
9	4	590
10	0	649
11	0	693

Tabla 20: Pasos para instancia 1

paso	costo	tiempo
1	32	623
2	25	739
3	25	860
4	23	982
5	25	999
6	22	1048
7	21	1065
8	19	1119
9	16	1137
10	22	1140
11	24	1148
12	15	1276
13	13	1299
14	6	1340
15	4	1394
16	4	1423

Tabla 21: Pasos para instancia 2

paso	costo	tiempo
1	45	1840
2	37	2194
3	20	2471
4	24	2705
5	12	2894
6	4	2909
7	7	3048
8	3	3193
9	3	3300

Tabla 22: Pasos para instancia 3

paso	costo	tiempo
1	25	1910
2	22	2301
3	15	2604
4	13	2740
5	11	2554
6	4	2651

Tabla 23: Pasos para instancia 4

paso	costo	tiempo
1	15	300
2	18	240
3	12	356
4	11	401
5	11	412

Tabla 24: Pasos para instancia 5

Anexo 3: Estudios de problemas de horarios

Tabla de estudios de problemas de horarios. Fuente (Pillay, 2014).

Métodos	Estudio	Problema	Restricciones
Bee algorithms	Lara et al. (2008)	Two real-world high school problems	PR1, PR2, NC1, NC2
Constraint programming	Marte (2002)	German gymnasium school	PR1, PR2, NC1, NC2, RU2, W1, W3, PD1, PD3, L1, L3
	Valouxis and Housos (2003)	Greek high school	PR1, PR2, NC1, NC2, NC3, PD1, PD3, PD4, P2
Constraint satisfaction methods	Meisels et al. (1997)		PR1, PR2, NC1, NC2
	Abbas and Tsang (2001)		PR1, PR2, NC1, NC2
Cyclic transfers	Post et al. (2010a)	German and English secondary schools	PR1, PR2, NC1, NC2, RU2, RU8, W1, W3, PD1, PD2, L5
Evolutionary algorithms	Colorni et al. (1990)	Italian high school	PR1, PR2, NC1, NC2, NC3
	Abramson and Abela (1991)	Generated problems (Beasley 2010)	PR1, PR2, NC1, NC2, NC3
	Calderia and Ross (1997)	Generated problem	PR1, PR2, PR3, NC1, NC2, NC3, RU2, PD1, PD2, PD3, L1
	Fernandes et al. (1999a)	Portuguese high school	PR1, PR2, NC1, NC2, NC3, PD3
	Fernandes et al. (1999b)	Portuguese high school	PR1, PR2, NC1, NC2, NC3, PD3
	Bufe et al. (2001)	German high school	PR1, PR2, NC1, NC2
	Di Stephano and Tettamanzi (2001)	Italian school	PR1, PR2, NC1, NC2, RU1, RU2, RU4, RU6, RU7, L1, L2, PD1, PD2, PD3, L1, L3
	Filho and Lorena (2001)	Brazilian high school	PR1, PR2, NC1, NC2, NC3, RU5, PD2, P2
	Wilke et al. (2002)	German school	PR1, PR2, PR3, PR4, NC1, NC2, RU2, PD1, P2, L1, L4
	Bedoya and Santos (2003)		PR1, PR2, NC1, NC2
	Ciscon et al. (2006)	Brazilian school	PR1, PR2, NC1, NC2, PD1, PD3
	Nurmi and Kyngas (2007)	Finnish schools	PR1, PR2, NC1, NC2, NC3, W1, W2, W3, PD1, PD2, PD4, L3

	Yigit (2007)	Technical and vocational Turkish high schools	PR1, PR2, NC1, NC2
	Beligiannis et al. (2008)	Greek high school	PR1, PR2, NC1, NC2, NC3, RU2, PD1, PD2, PD4, L2, L4
	Mohammadi and Lucas (2008)		PR1, PR2, PR3, NC1, NC2
	Nurmi and Kyngas (2008)	ITC'07 data set	PR1, PR2, NC1, NC2, NC3, RU2, PD1
	Raghavjee and Pillay (2008)	Generated problems (Beasley 2010)	PR1, PR2, NC1, NC2, NC3
	Beligiannis et al. (2009)	Greek high school	PR1, PR2, NC1, NC2, NC3, RU2, PD1, PD2, PD4, L2, L4
	Moschopoulos et al. (2009)	Greek high school	PR1, PR2, NC1, NC2, NC3, RU2, PD1, PD2, PD4, L2, L4
	Raghavjee and Pillay (2009)	Greek high school	PR1, PR2, NC1, NC2, NC3, RU2, PD1, PD2, PD4, L2, L4
	Srndic et al. (2009)		PR1, PR2, NC1, NC2
	Raghavjee and Pillay (2010a)	South African high school	PR1, PR2, NC1, NC2, P2, L1, L4
	Raghavjee and Pillay (2010b)	South African primary school	PR1, PR2, NC1, NC2, P2, L1, L4
		South African high school	PR1, PR2, NC1, NC2, RU5, PD3, L1, L3
GRASP	Moura and Scaraficci (2010)	Brazilian high school	PR1, PR2, NC1, NC2
Integer programming	Lawrie (1969)		PR1, PR2, NC1, NC2
	Birbas et al. (1997)	Greek high school	PR1, PR2, NC1, NC2, W1, W3, RU2, PD1, PD2, PD3, L1
	Papoutsis et al. (2003)	Greek high school	PR1, PR2, NC1, NC2, W1, PD1, PD2, PD3, PD4, P2
	Boland et al. (2008)	Australian high school	PR1, PR2, NC1, NC2, RU2, RU4, PD1
	Santos et al. (2008)	Brazilian high school	PR1, PR2, NC1, NC2, RU2, PD1, PD2, L3
	Birbas et al. (2009)	Greek high school	PR1, PR2, NC1, NC2, RU2, PD1, PD2, PD3, P2, L1, L3, L5
	Ribic and Konjicija (2010)		PR1, PR2, PR3, NC1, NC2, RU2, RU5, RU8, PD1, L1
Neural networks	Carrassco and Pato (2004)	Generated problems	PR1, PR2, NC1, NC2
Simulated annealing	Abramson and Abela (1991)	Australian school	PR1, PR2, NC1, NC2, NC3, RU2, W3, L3

	Abramson et al. (1999)	Generated problems	PR1, PR2, NC1, NC2, NC3
	Melicio et al. (2006)	Portuguese school	PR1, PR2, NC1, NC2
	Yongkai et al. (2009)	Two real-world high school problems	PR1, PR2, NC1, NC2
Tabu search	Wright (1996)	English comprehensive school	PR1, PR2, NC1, NC2, RU6, PD1, PD2, L1, L2, L3
	Alvarez-Valdes et al. (2002)	Spanish high school	PR1, PR2, NC1, NC2
	Santos et al. (2004)	Brazilian high school	PR1, PR2, NC1, NC2, RU2, PD1, PD2, PD4, L3
	Santos et al. (2005)	Brazilian high school	PR1, PR2, NC1, NC2, RU2, PD1, PD2, PD4, L3
	Jacobsen et al. (2006)	German high school	PR1, PR2, NC1, NC2, NC3, RU2, RU7, RU8, W1, W2, PD1, PD3, PD4, P1, L1, L3
	Ohtsubo et al. (2006)	Junior high school	PR1, PR2, NC1, NC2
	Bello et al. (2008)	Brazilian high school	PR1, PR2, NC1, NC2, RU2, PD2, PD3, PD4, W1, L3
	Minh et al. (2010)	Vietnam high school	PR1, PR2, NC1, NC2
Threshold accepting	Abboud et al. (1998)	Generated problems	PR1, PR2, NC1, NC2, RU2, PD3, PD4, P1
Tiling algorithms	Kingston (2004)	Australian high school	PR1, PR2, NC1, NC2
	Kingston (2006)	Australian high school	PR1, PR2, NC1, NC2, NC3, RU5, W1, PD2, PD3, L3, L4
	Kingston (2008)	Australian high school	
Walk Down Jump Up Algorithm	Wilke and Killer (2010b)	German high school	PR1, PR2, NC1, NC2, NC3, PD1
Hybrid approaches	Yoshikawa et al. (1994) • Arc consistency • Hill climbing		PR1, PR2, NC1, NC2, PD1, PD3, L5
	Yoshikawa et al. (1996) • Arc consistency • Hill climbing		PR1, PR2, NC1, NC2, RU2, RU3, W1, PD3
	Schaerf (1996) • Tabu search • Randomize non ascendant	Italian schools	PR1, PR2, NC1, NC2, RU1, W1, PD2, PD3, P2, L3, L4
	Alvarez-Valdes et al. (1996)		PR1, PR2, NC1, NC2, RU2, RU7, RU8, PD1, PD2, PD3, L1, L4

• Tabu search • Floyd-Warshall algorithm	Spanish schools	
Monfroglio (1996) • Genetic algorithms • Constrained heuristic search		PR1, PR2, NC1, NC2
Drexl and Salewski (1997) • Greedy randomized algorithms • Genetic algorithms	German high school	PR1, PR2, PR3, NC1, NC2, NC3, RU2, PD1, L1, L5
Schaerf (1999b) • Tabu search • Randomize non ascendant	Italian high school	PR1, PR2, NC1, NC2, RU1, W1, PD2, PD3, P2, L3, L4
Lohnertz (2002) • Tabu search • Graph coloring matching algorithm	German gymnasium school	PR1, PR2, NC1, NC2, RU2, L3
Willemen (2002) • Tree search algorithm • Tabu search	Dutch school	PR1, PR2, NC1, NC2, NC3, RU2, RU5, PD3, L3
Kwan et al. (2003) • Constraint technology • Heuristics • Local search with a tabu list		PR1, PR2, NC1, NC2
de Haan (2004) • Beam search • Branch and bound algorithm	Netherlands secondary school	PR1, PR2, NC1, NC2, NC3, RU2, RU5, PD1, PD2, PD3, P2, L1, L2, L4, L5
Landman (2005) • Beam search • Branch and bound algorithm • Shifting algorithm • Re-coloring algorithm	Dutch school	PR1, PR2, NC1, NC2, NC3, RU2, RU5, PD1, PD2, PD3, PD4, P2, L1, L4

	Post and Ruizenaar (2004) • Clustering algorithm • Branch and bound algorithm	Netherlands secondary school	PR1, PR2, NC1, NC2, NC3, RU2, RU4, RU6, RU8, PD1, PD2, PD3, P2, L1, L2, L4, L5
	Souza (2004) • GRASP • Tabu search	Brazilian high school	PR1, PR2, NC1, NC2, RU2, W1, PD3, L3
	Avella et al. (2007) • Very large neighborhood search • Simulated annealing	Generated problems	PR1, PR2, NC1, NC2, RU2, W3, PD1, PD2, PD3, PD4, P2, L1
	de Haan et al. (2007a) • Clustering algorithm • Tabu search	Netherlands secondary school	PR1, PR2, NC1, NC2, RU2, PD1, PD2, PD3, PD4, L3
	De Haan et al. (2007b) • Branch and bound algorithm • Dynamic priority rule • First-fit heuristic • Tabu search	Netherlands secondary school	PR1, PR2, NC1, NC2, RU2, PD1, PD2, PD3, PD4, L3
	Zuters (2007) • Genetic algorithms • Neural networks		PR1, PR2, NC1, NC2, PD1, PD3, L2
	Cedeira-Pena et al. (2008) • Random ascent method • Genetic algorithms		PR1, PR2, NC1, NC2
	Liu et al. (2009) • Simulated annealing • Tabu search		PR1, PR2, NC1, NC2, RU2, PD1, PD2, PD3, PD4, P2
Comparative studies	Colorni et al. (1998) • Simulated annealing • Tabu search • Genetic algorithms	Italian high school	PR1, PR2, NC1, NC2, PD1, PD2
	Smith et al. (2003)		

	• Hopfield neural network • Greedy search • Simulated annealing • Tabu search	Generated problems	PR1, PR2, NC1, NC2, NC3
	Wilke and Ostler (2008) • Tabu search • Simulated annealing • Genetic algorithms • Branch and bound algorithms	German high school	PR1, PR2, NC1, NC2, NC3, RU4
	Wilke and Killer (2010b) • Genetic algorithms • Immune systems • Harmony search • Tabu search • Simulated annealing • Great deluge • Walk down jump up	German high school	PR1, PR2, NC1, NC2, NC3, RU4
Distributed methods	Slechta (2005)		PR1, PR2, NC1, NC2
	Kingston (2010)		PR1, PR2, NC1, NC2, NC3, RU5, W1, PD2, PD3, L3, L4

Anexo 4: Regresión lineal múltiple

La regresión lineal es un método matemático que modela la relación entre una variable dependiente *Y*, mediante variables independientes X_j y un término aleatorio ε. El modelo se puede expresar de la siguiente manera:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \varepsilon$$

Donde β_0 es la intersección o término constante, las β_i ($i > 0$) son los parámetros respectivos a cada variable independiente, y p es el número de parámetros independientes a tener en cuenta en la regresión.

Para poder crear un modelo de regresión lineal, es necesario que se cumpla con los siguientes supuestos:

- El modelo de regresión es lineal en los parámetros
- La variable X no es aleatoria
- $E(\varepsilon_i | X_i) = 0$
- Homocedasticidad
- Errores aleatorios sin autocorrelación
- $\text{Cov}(\varepsilon_i | X_i) = 0$
- Var(*X*) es un número finito (no todos los *X* son iguales)
- El modelo está correctamente especificado
- No hay multicolinealidad perfecta
- Parsimonia