UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE QUITO

CARRERA:
INGENIERÍA DE SISTEMAS

Trabajo de titulación previo a la obtención del título de:
Ingeniero de Sistemas

TEMA:
IMPLEMENTACIÓN DE UN SISTEMA DE ASIGNACIÓN AUTOMÁTICA Y GESTIÓN DE AULAS EN EL CAMPUS SUR DE LA UNIVERSIDAD POLITÉCNICA SALESIANA.

AUTOR:
TOMÁS ALEJANDRO CORONEL CÁRDENAS

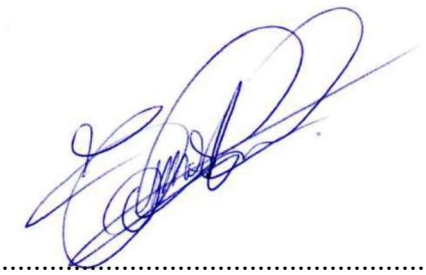
TUTOR:
JULIO RICARDO PROAÑO ORELLANA

Quito, marzo de 2021

CESIÓN DE DERECHOS DE AUTOR

Yo, Tomás Alejandro Coronel Cárdenas, con documento de identificación N° 0930706080, manifiesto mi voluntad y cedo a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que soy autor del trabajo de grado intitulado: IMPLEMENTACIÓN DE UN SISTEMA DE ASIGNACIÓN AUTOMÁTICA Y GESTIÓN DE AULAS EN EL CAMPUS SUR DE LA UNIVERSIDAD POLITÉCNICA SALESIANA., mismo que ha sido desarrollado para optar por el título de: INGENIERO DE SISTEMAS, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En aplicación a lo determinado en la Ley de Propiedad Intelectual, en mi condición de autor me reservo los derechos morales de la obra antes citada. En concordancia, suscribo este documento en el momento que hago entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.



.....

TOMÁS ALEJANDRO CORONEL CÁRDENAS

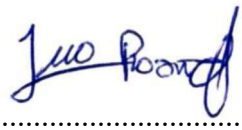
CI: 0930706080

Quito, marzo de 2021

DECLARATORIA DE COAUTORÍA DEL DOCENTE TUTOR/A

Yo declaro que bajo mi dirección y asesoría fue desarrollado el Artículo Académico, con el tema: IMPLEMENTACIÓN DE UN SISTEMA DE ASIGNACIÓN AUTOMÁTICA Y GESTIÓN DE AULAS EN EL CAMPUS SUR DE LA UNIVERSIDAD POLITÉCNICA SALESIANA. realizado por Tomás Alejandro Coronel Cárdenas, obteniendo un producto que cumple con todos los requisitos estipulados por la Universidad Politécnica Salesiana, para ser considerado como trabajo final de titulación.

Quito, marzo de 2021



.....

JULIO RICARDO PROAÑO ORELLANA

CI: 0103909412

DEDICATORIA

Este trabajo es dedicado primeramente a Dios, por haberme brindado salud y sabiduría para poder finalizar mis estudios, a mi esposa por haberme siempre apoyado en las decisiones que he tomado, a mi hijo quien es mi mayor inspiración y por quien me propongo a cumplir cada meta que trazo para mi núcleo familiar, a mis padres, quienes me brindaron el apoyo para iniciar mis estudios universitarios y haber sembrado en mí los valores que han permitido ser la persona que soy.

En especialmente a mi mami Matilde, quien a pesar de su discapacidad ha logrado ser un apoyo fundamental desde mis primeros años de vida.

Tomás Alejandro Coronel Cárdenas

AGRADECIMIENTO

Agradezco a la Universidad Politécnica Salesiana, pues me abrió sus puertas para estudiar en tan prestigiosa institución, a mi tutor Julio Proaño y a todos los integrantes de la Unidad de Titulación, por el apoyo brindado y la confianza a lo largo del desarrollo de mi trabajo de titulación.

Y a cada una de las personas dentro de mi vida profesional quienes me brindaron un apoyo para poder finalizar mis estudios.

Tomás Alejandro Coronel Cárdenas

IMPLEMENTACIÓN DE UN SISTEMA DE ASIGNACIÓN AUTOMÁTICA Y GESTIÓN DE AULAS EN EL CAMPUS SUR DE LA UNIVERSIDAD POLITÉCNICA SALESIANA

IMPLEMENTATION OF AN AUTOMATIC ASSIGNMENT AND CLASSROOM MANAGEMENT SYSTEM IN THE SOUTH CAMPUS OF THE SALESIAN POLYTECHNIC UNIVERSITY

Julio Proaño[1], Tomás Coronel 2[2]

Resumen

El presente documento tiene como objetivo desarrollar un prototipo a la dificultad que afronta hoy en día la Universidad Politécnica Salesiana Sede Quito Campus Sur, dentro del proceso de asignación de horarios y aulas, las mismas que se realizan de manera manual, lo cual puede conllevar a cometer errores dejando como consecuencia la inversión de más tiempo para la corrección de las novedades presentadas, de la misma manera puede concurrir en una falta de optimización de aulas y distancia recorrida por parte de los estudiantes. Se propone el desarrollo de un prototipo usando un algoritmo genético [1], implementado en base de datos, probando una alta eficacia en el consumo de recursos computacionales, en donde la última generación será la más óptima dentro de la asignación de horarios y aulas de clases, teniendo así una eficiencia en los tiempos de los estudiantes y espacios de la universidad, así mismo de implementar una arquitectura de datos la cual permita administrar las variables usadas por el algoritmo, los cuales permitan calibrar las variables que usa el proceso de asignación y para la ejecución del algoritmo genético.

Palabras Clave: algoritmo genético, base de datos, proceso de asignación, generación.

Abstract

The objective of this document is to develop a prototype of the difficulty faced by the Universidad Politécnica Salesiana Sede Quito Campus Sur today, within the process of assigning timetables and classrooms, which are carried out manually, which can lead to making mistakes, leaving therefore the investment of more time for the correction of the novelties presented, in the same way it can concur in a lack of optimization of classrooms and distance traveled by students. The development of a prototype is proposed using a genetic algorithm [1], implemented in a database, proving a high efficiency in the consumption of computational resources, where the latest generation will be the most optimal within the allocation of schedules and classrooms of classes, thus having an efficiency in the times of the students and spaces of the university, as well as implementing a data architecture which allows to manage the variables used by the algorithm, which allow to calibrate the variables used by the allocation process and for the execution of the genetic algorithm.

Keywords: genetic algorithm, database, allocation process, generation.

[1] Estudiante de Ingeniería de Sistemas, Universidad Politécnica Salesiana, Sede Quito Campus Sur – Ecuador.
Autor para correspondencia: tacc91@gmail.com

[2] Docente de la Carrera de Ingeniería de Sistemas, Universidad Politécnica Salesiana, Sede Quito Campus Sur – Ecuador
Autor para correspondencia: jproanoo@ups.edu.ec

1. Introducción

En el campus sur de la Universidad Politécnica Salesiana se desarrolló un proyecto de titulación en donde su principal objetivo es la asignación automática de aulas a las diferentes materias impartidas de cada una de las carreras de pregrado [6]; sin embargo, dicho proyecto hoy en día no ha sido implementado.

El algoritmo usado en el proyecto tiene un tiempo de procesamiento extendido y solo resuelve una parte del problema, la cual es la asignación de aulas [6], y no contempla la asignación de horarios.

Por lo que el personal administrativo tiene como una de sus actividades asignar las aulas que se encuentren disponibles en los diversos bloques para cada una de las materias que se van a dictar durante el semestre, las mismas que corresponden a los diferentes niveles de cada una de las carreras.

Sin embargo, esta es una actividad que se realiza de manera manual, lo cual presenta los siguientes inconvenientes:

- Una alta demanda de tiempo para poder llevar a cabo la asignación de aulas y/o laboratorios.
- No optimizar el uso de los espacios físicos.
- No considerar el desplazamiento de los estudiantes al momento de recibir una materia en otra aula diferente.

La propuesta para la solución del problema que sigue presentando la universidad es implementar un algoritmo genético [1], el cuál permita no solo la optimización de aulas, sino también incluir la distribución de horarios para las diferentes materias que se imparten para cada grupo y carrera que se dicta en la sede, así mismo de una arquitectura de datos que permita la administración de todos los parámetros que intervienen dentro del proceso de asignación, esta solución es implementada en base de datos, el cual permite una correcta distribución de recursos y aun no hay ningún algoritmo genético implementado bajo este lineamiento.

Una de las limitantes del prototipo implementado es la falta de una interfaz gráfica, sin embargo, el proceso de la ejecución del algoritmo funciona en sus máximas condiciones dejando todos los recursos del equipo para su uso exclusivo al momento de ejecutar el algoritmo.

2. Trabajos Relacionados

El objetivo principal del proyecto “Sistema para optimizar asignación de aulas UNICEN” es poder implementar el uso de algoritmos genéticos para la asignación inteligente de aulas en las materias que se dictan dentro de la UNICEN [4], incluyendo un Software en plataforma web desarrollado bajo el esquema Modelo-Vista-Controlador, el cual ayude a gestionar de manera efectiva los parámetros de ingreso para la ejecución del modelo y la reportería necesaria que permita realizar consultas a los diferentes horarios y espacios ocupados.

La implementación del algoritmo permitió una mejor optimización de las aulas de la UNICEN, así mismo el desarrollo de su propio software Web que permite una mejor administración de los parámetros iniciales [4], donde también se desataca el uso de la cantidad de reportes implementados para la visualización del estado de cada espacio físico.

El trabajo “Iterated Heuristic Algorithms for the Classroom Assignment Problem” tiene como objetivo principal la aplicación de dos algoritmos heurísticos iterados, los cuales permitan la resolución de los problemas de asignación clásicos y cuellos de botella [5], y

un tercer algoritmo llamado Búsqueda de entornos variados, para poder comparar y llegar a una conclusión de cuál es el óptimo y responda a las necesidades para la asignación de aulas.

Dentro de los resultados mostrados los algoritmos de asignación lineal o clásico y el algoritmo de búsqueda de entornos variados resultan ser mucho más eficientes en comparación a la realización manual [5], dichos modelos, a pesar de tener un número mayor de iteraciones representan una mejora considerable en cuestión de distancia entre un aula y otra, una menor cantidad de valores desfavorables y un costo total muy inferior. Los tres algoritmos fueron ejecutados con datos reales de tres periodos académicos.

Dentro del proyecto "Análisis e implementación de un algoritmo de optimización para la asignación automática de aulas clase en el Campus Sur de la Universidad Politécnica Salesiana" el objetivo principal de este trabajo es la implementación de un algoritmo metaheurístico Hill-Climbing, con reinicios aleatorios para la asignación de aulas en los diferentes cursos [6], así mismo un entorno web que permite el ingreso de los horarios, los cuales son el insumo primordial del algoritmo.

El resultado del algoritmo fue evaluado con datos reales correspondientes al periodo 50 de la Universidad [6]. La ejecución del algoritmo tarda un lapso de 10 a 20 minutos en el cual determina la mejor solución [6]. Con los resultados se puede evidenciar que hubo una mejora significativa que van desde el tiempo de asignación de aulas hasta su optimización.

3. Fundamentación Teórica

3.1 Algoritmos Genéticos

"Los Algoritmos Genéticos fueron inspirados en la teoría Darwaniana de la evolución" [2], en donde cada individuo representa una solución a un problema, el conjunto de individuos se lo conoce como población [1], [2].

Estos algoritmos también son usados para la resolución de problemas de optimización, en donde se evalúa la aptitud de cada individuo de generación en generación, hasta encontrar al más apto el cual cumpla con las restricciones deseadas [1], [2].

Los operadores más conocidos en los algoritmos genéticos son los siguientes:

- Selección
- Cruza
- Mutación

Los cuales permiten llegar a una solución factible, el algoritmo se representa de mejor manera en la figura 1 [1].

```
1: initialize population
2: repeat
3:    repeat
4:       crossover
5:       mutation
6:       phenotype mapping
7:       fitness computation
8:    until population complete
9:    selection of parental population
10: until termination condition
```

Figura 1. Representación de un algoritmo genético. [1]

3.2 Individuo

Un individuo está representado por un cromosoma, el cual contiene todo el material genético de la solución [3].

El cromosoma está compuesto por genes [3] que a su vez estos están compuestos por alelos como se muestra en la figura 2.

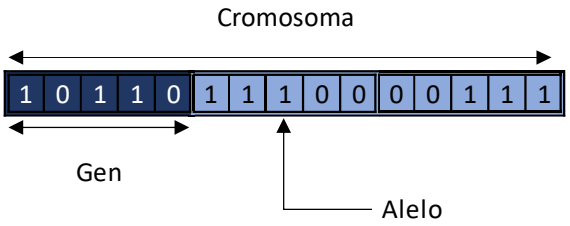


Figura 2. Representación de un cromosoma.

"Los genes pueden describir una posible solución a un problema, sin ser realmente la solución." [3], estos pueden estar representados de acuerdo con las necesidades del problema a resolver, las más comunes son, Sistema Binario, Sistema Decimal, Sistema Hexadecimal y Sistema Real.

Cada gen representa el conjunto de variables del problema a resolver, la solución está codificada en lo que se le conoce como genotipo, la cual para representar la solución es necesario realizar una decodificación lo que se le conoce como fenotipo [3].

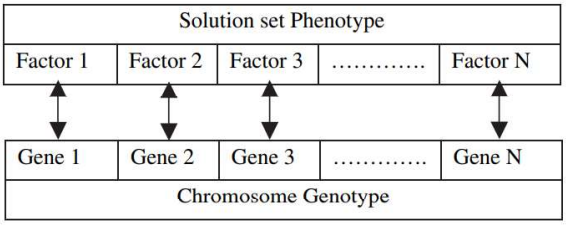


Figura 3. Representación de una decodificación de genotipo a fenotipo. [3]

3.3 Población Inicial

El conjunto de individuos o cromosomas se lo conoce como población [3], en donde cada uno de ellos son posibles soluciones al problema que se está planteando [8], como se lo muestra en la figura 4.

Population	Chromosome 1	1 1 1 0 0 0 1 0
	Chromosome 2	0 1 1 1 1 0 1 1
	Chromosome 3	1 0 1 0 1 0 1 0
	Chromosome 4	1 1 0 0 1 1 0 0

Figura 4. Representación de la población. [3]

Para crear la primera generación o población inicial, se debe primero definir la cantidad de individuos que van a ser parte de la primera generación y de las futuras, es recomendable que el tamaño sea lo más amplio posible para poder tener una gran diversidad de soluciones [3], [8]. Segundo, para definir dicha población se lo puede hacer al azar, con el fin de evitar una convergencia prematura [8], y nos permita explorar una serie de mejores individuos hasta encontrar el más apto.

3.7 Fitness

La función fitness o aptitud de un individuo se calcula a partir de la decodificación del cromosoma, es decir, a partir del genotipo calculamos el fenotipo [1].

El valor obtenido, determinará la calidad del individuo que ha sido generado por el algoritmo genético. "El diseño de la función fitness forma parte del diseño de cualquier algoritmo genético el cual cumpla con los criterios de optimización" [1], en donde el criterio de la función dependerá del problema que se vaya a optimizar.

3.4 Selección

La selección es el proceso en el cual se obtienen dos individuos al azar de una misma generación, de esta manera poder realizar la cruza o apareamiento, obteniendo nuevos individuos o hijos que formarán parte de la siguiente descendencia [3].

"El propósito de la selección es enfatizar a los individuos más aptos de la población con la esperanza de que sus descendientes tengan un mejor estado físico." [3], una idea más clara del proceso de selección lo podremos ver en la figura 5.

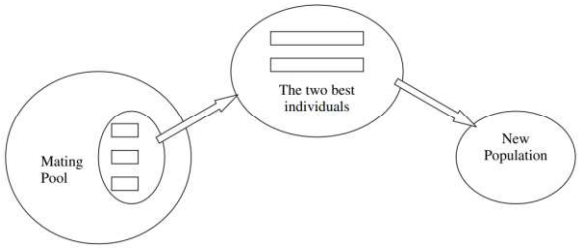


Figura 5. Ejemplo del proceso de selección. [3]

Para el proceso de selección existen diversas técnicas, de la cuales tenemos las siguientes Selección por Ruleta, Selección Aleatoria, Selección por Rank, Selección por Torneo, Selección de Boltzmann y Muestreo Estocástico Universal.

Sin embargo, para efectos de este documento nos enfocaremos en la selección por ruleta.

3.4.1 Selección por Ruleta

La selección por ruleta es la técnica tradicional dentro de los algoritmos genéticos [3], su principio fundamental es poder obtener de manera proporcional el porcentaje de participación del valor de aptitud de cada individuo con respecto a la aptitud de la generación actual, y de esa manera realizar un recorrido mediante una ruleta, en donde los individuos más aptos tendrán una mayor probabilidad de ser seleccionados.

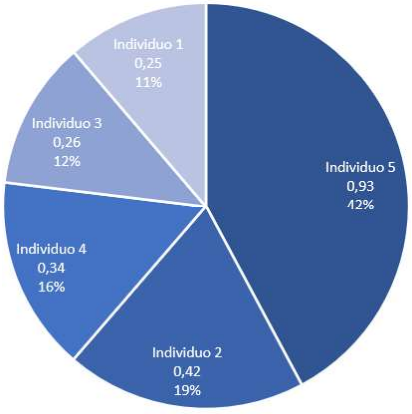


Figura 6. Ejemplo de una selección por ruleta.

3.5 Cruza

La cruza es el método que le permite a dos individuos poder reproducirse y obtener uno o varios hijos, combinando el material genético de cada padre [1], [3].

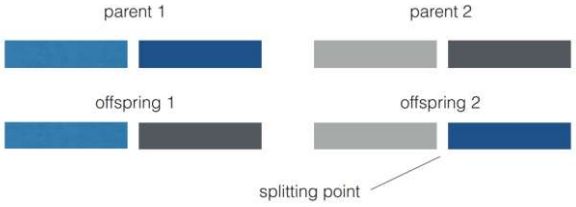


Figura 7. Ejemplo de cruce de dos individuos [1].

Dentro del método de cruce existen varias estrategias, como las siguientes Cruce de un punto, Cruce de dos puntos, Cruce de N puntos y Cruce uniforme.

Para el desarrollo de este proyecto, se utilizará el cruce de un punto [3], siendo muy similar a lo que se muestra en la figura 7.

3.6 Mutación

La mutación permite alterar parte del material genético de un individuo, debido a que un alelo ha mutado, conlleva a cambiar su fenotipo, lo que tiene como consecuencia alterar el valor de la aptitud del individuo [2], [3].

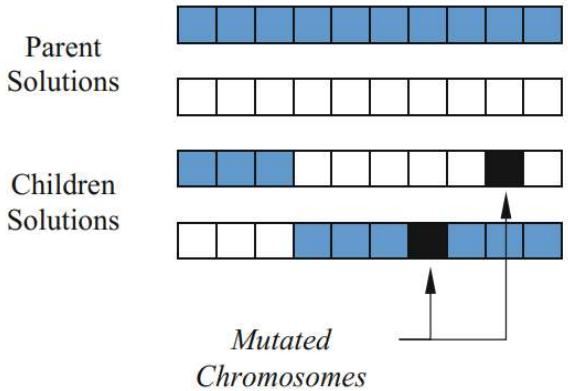


Figura 8. Ejemplo de mutación [2].

Como se observa en la figura 8 es un claro ejemplo de cómo funciona el proceso de mutación, así mismo existen varias estrategias, las cuales tenemos las siguientes

Mutación uniforme, Mutación no-uniforme, Mutación de potencia, Mutación supervisada y Mutación Gaussiana.

Para efectos de este proyecto se hará uso de la mutación uniforme, la cual obtiene un valor aleatorio, dentro del rango de posibles valores que la variable vaya a mutar [7].

4. Materiales y Métodos

Considerando el inconveniente que presenta la Universidad Politécnica Salesiana Campus Sur con respecto a la asignación de aulas, en donde hoy en día todavía lo sigue realizando de manera manual, se plantea implementar un algoritmo genético desarrollado en Transact-SQL dentro de una base de datos SQL Server.

El algoritmo ha sido implementado siguiendo las características básicas conceptuales que lo conforman, cada una de ellas están representadas mediante procedimientos almacenados y funciones, las cuales se las dará a conocer a detalle a medida se vaya avanzando en el documento.

Para obtener un mejor detalle de los procedimientos almacenados implementados y del proceso en general, se recomienda la refencia [9].

4.1 Variables a resolver

El algoritmo plantea resolver las variables presentadas en la tabla 1.

Tabla 1: Variables a resolver

Variable	Descripción
Generar y optimizar horarios de clase	Generar los horarios académicos de las materias que se imparten para cada una de las carreras dentro de la Universidad Politécnica Salesiana Campus Sur.
Asignación de aulas	Asignar las aulas para cada uno de los horarios de manera óptima y satisfaga las restricciones establecidas.
Optimización del desplazamiento entre aulas	Optimizar el desplazamiento de los estudiantes, durante cada clase que reciban en los diferentes días de la semana.

Para lograr el objetivo del algoritmo, los datos más relevantes tienen que estar activos con el fin de que el proceso lo considere para poder obtener una solución óptima, la mismas son aulas, días de semana, horas, materias, grupos, mención y carrera.

4.2 Características del algoritmo

El proceso comienza obteniendo las variables importantes para la ejecución de cualquier algoritmo genético, las mismas son tamaño de la población, probabilidad de cruza, probabilidad de mutación y cantidad total de generaciones.

De igual manera se obtienen variables de trabajo que van a ayudar a la correcta ejecución del algoritmo.

Tabla 2: Variables de trabajo

Variables de trabajo	Descripción
Contador	Variable que permite controlar la creación de nuevos individuos.
Suma del valor de fitness	Variable que obtiene el valor de aptitud de la población.

Obtenido los valores iniciales y declaradas las variables de trabajo el proceso está listo para comenzar a generar soluciones hasta encontrar la óptima, sin antes primero limpiar las tablas de trabajo para lograr una correcta ejecución del algoritmo.

Tabla 3: Tablas de trabajo

Tablas de trabajo	Descripción
tmp_poblacion	Tabla donde se almacena a los individuos por cada generación.
tmp_ruleta	Tabla donde se almacenan a los individuos con su aptitud y su participación.
tmp_hijos	Tabla donde se almacena la descendencia de la generación actual.
tmp_mejores_individuos	Tabla donde se almacenan los mejores individuos de todas las generaciones.

La estructura de la tabla tmp_poblacion y tmp_hijos tienen la misma estructura, con el fin de cuando se pase a la siguiente generación los datos de tmp_poblacion se eliminen y se inserten los datos que contiene la tabla tmp_hijos, una vez realizado dicho paso se eliminan los datos de tmp_hijos para calcular la siguiente descendencia, la estructura de los objetos se muestra en las siguientes tablas:

Tabla 4: tmp_poblacion

Campo	Descripción
i_individuo	Identificador del individuo.
i_alelo	Número del alelo por cada gen.
i_id_gru_carrera	Identificador del grupo y la carrera.
i_id_materia	Identificador de la materia
i_id_dia_fenotipo	Identificador del día
s_id_dia_genotipo	Codificación binaria del día
i_id_hora_fenotipo	Identificador de la hora
s_id_hora_genotipo	Codificación binaria de la hora
i_id_aula_fenotipo	Identificación del aula
s_id_aula_genotipo	Codificación binaria del aula.
f_fitness	Valor de Aptitud de
i_parent1	Identificador del padre 1 de la generación anterior
i_parent2	Identificador del padre 2 de la generación anterior
f_guia_mutacion	Identificador el cual permitirá la mutación de las variables que faltan optimizar.

Tabla 5: tmp_ruleta

Campo	Descripción
i_individuo	Identificador del individuo.
f_probabilidad	Valor con la probabilidad de ser seleccionado de un individuo.
f_fitness	Valor de aptitud del individuo

Tabla 6: tmp_hijos

Campo	Descripción
i_individuo	Identificador del individuo.
i_alelo	Número del alelo por cada gen.
i_id_gru_carrera	Identificador del grupo y la carrera.
i_id_materia	Identificador de la materia
i_id_dia_fenotipo	Identificador del día
s_id_dia_genotipo	Codificación binaria del día
i_id_hora_fenotipo	Identificador de la hora
s_id_hora_genotipo	Codificación binaria de la hora
i_id_aula_fenotipo	Identificación del aula
s_id_aula_genotipo	Codificación binaria del aula.
f_fitness	Valor de Aptitud de
i_parent1	Identificador del padre 1 de la generación anterior
i_parent2	Identificador del padre 2 de la generación anterior
f_guia_mutacion	Identificador el cual permitirá la mutación de las variables que faltan optimizar.

Una vez listo el escenario se procede a ejecutar el algoritmo siguiendo los siguientes pasos:

1. Generar la población inicial.
2. Actualizar la ruleta para comenzar a generar la siguiente descendencia.
3. Crear la nueva generación.

4.2.1 Generar la población inicial

Para poder generar la población inicial se ejecuta la cantidad de veces que sea necesario el procedimiento almacenado spr_poblacion_inicial, el cual va a ir crear un individuo a la vez hasta lograr completar el número total de individuos de acuerdo con el valor almacenado en la variable @TamanioPoblacion.

Por cada combinación de materia y grupo de carrera, el procedimiento procederá a crear valores aleatorios para las variables de día, hora y aula, siempre y cuando estén dentro del rango permitido de cada una de las variables, al final de esta ejecución se habrá obtenido un nuevo individuo o solución que formará parte de la primera generación.

Creado el individuo, el último paso dentro del procedimiento será obtener su valor de aptitud y establecer la guía de mutación cuyos valores podrán ser:

- **4:** Cualquier variable puede mutar.
- **3:** Las variables hora y aula pueden mutar.
- **2:** La variable aula puede mutar.
- **1:** Ninguna variable puede mutar, por lo general esto es cuando se ha encontrado la solución óptima del problema.

4.2.2 Actualizar la ruleta

A partir de este punto ya entramos en el ciclo para proceder a crear nuevas generaciones, sin embargo, el primer paso que se realiza, una vez creada la primera generación, es limpiar los valores de la ruleta.

Para esta sección los valores de aptitud de los individuos ya fueron calculados, entonces se procede a obtener la suma de aptitudes de la población actual para calcular el ponderado de cada individuo, el cual será la probabilidad de selección de cada uno de ellos, para ello se hace uso de la siguiente formula:

$$Ponderado\ del\ Individuo_i = \frac{fitness_i}{\sum_{i=1}^{n} fitness_i} \quad (1)$$

4.2.3 Crear la nueva generación

A partir de este punto se va a lograr a obtener las futuras generaciones, en donde se hace uso de los operadores genéticos como es la selección, la cruza y la mutación.

El procedimiento almacenado que se convoca es spr_nueva_generacion, el cual recibe como parámetros el tamaño de la población, la probabilidad de cruza y la probabilidad de mutación, los mismos se obtienen al inicio de la ejecución del algoritmo.

Dentro del procedimiento almacenado, como se lo indicó anteriormente se hace el llamado de los operadores genéticos los cuales van a ayudar a la creación de la nueva descendencia.

El primer paso que se realiza es seleccionar a dos individuos aptos, esto es posible ya que los datos de la ruleta fueron previamente actualizados con los valores de la generación actual, para esto hacemos uso de la función fn_SeleccionRuleta, la cual devolverá un número indicando al individuo que fue seleccionado.

Obteniendo los dos padres, el siguiente paso es realizar la cruza o apareamiento entre estos individuos para poder obtener a dos nuevos hijos, para esto, se ejecuta el procedimiento almacenado spr_crossover, el cual va a recibir los siguientes parámetros:

- Padre 1.
- Padre 2.
- Probabilidad de cruza.

Dentro de este procedimiento almacenado, como primer paso se realiza un llamado a la función fn_flip, la cual recibe como parámetro la probabilidad de cruza, la función flip simula el lanzamiento de la moneda, los valores posibles que se pueden obtener son 0 o 1.

Si el valor de flip es 1, se procede a generar los dos descendientes, para esto se obtiene el punto de cruza, mediante un aleatorio entre 1 y el tamaño del cromosoma; ya disponible dicho valor se procede a generar la nueva descendencia. Si el valor de flip es 0, el material genético de los padres pasa a la siguiente generación sin realizar ninguna cruza.

Esta nueva generación se almacena dentro de la tabla tmp_hijos, sobre estos nuevos hijos se procede a realizar la mutación, para esto se llama al procedimiento almacenado spr_mutation, el cuál recibe como parámetros

al individuo que va a mutar y la probabilidad de mutación.

Dentro de este procedimiento, primero se obtiene el valor de la aptitud antes de mutar, para conocer sobre que variables se puede realizar la mutación, realizado este paso, se procede a llamar a la función flip la cual recibe como parámetro la probabilidad de mutación.

Si el valor de flip es igual 1, se obtiene el valor del alelo de manera aleatoria que va a mutar, sobre este alelo se actualizan los valores de las variables que van a poder mutar, de acuerdo con lo obtenido en la aptitud del individuo, estas pueden ser día, hora y aula, para lo cual el valor se actualiza con un valor aleatorio entre 1 y el rango máximo permitido de dichas variables. Luego de esto se procede a obtener su nuevo valor de aptitud.

Finalizada la cruza, se procede a ejecutar nuevamente todos los pasos anteriores hasta completar la cantidad de individuos para la nueva descendencia.

Una vez completos todos los individuos para la siguiente generación, se procede a realizar el elitismo, el cual consiste en obtener a los dos peores hijos y al mejor padre de la generación actual, con el fin de que estos dos peores hijos hereden el material genético del mejor padre, esto se logra ejecutando el procedimiento almacenado spr_elitism, finalmente, se desecha a la generación actual, la cuál es reemplaza por la nueva generación.

Todo esto se ejecuta tantas veces sea necesario hasta haber alcanzado el máximo de generaciones o haber encontrado una solución óptima que cumpla con todas las restricciones.

4.3 Función Fitness

La función fitness, es la que va a permitir obtener el valor de cada uno de los individuos, de acuerdo con el cumplimiento de las siguientes restricciones:

- El día de clases sea laborable.
- El número de horas por materia sea igual a 2 por día.
- Las 2 horas para recibir cada materia sean contiguas.
- Se reciban máximo dos materias por día.
- El estado de la hora a recibir una materia debe ser activo.
- Las horas para recibir una materia deben ser contiguas.
- Si se reciben dos materias en un día ambas materias deben tener horarios contiguos.
- El estado del aula en donde se va a recibir una materia debe ser activo.
- El tipo de aula en donde se va a recibir una materia debe ser la adecuada, por ejemplo, si es una materia que se debe impartir en laboratorio, el aula debe ser un laboratorio.
- Las materias que se reciben en un mismo día deben ser en la misma aula.

Conocidas las restricciones que el algoritmo debe cumplir, la forma de realizar el cálculo de aptitud de cada individuo será contabilizar los errores, en donde si un individuo tiene la menor cantidad de errores el valor de aptitud va a ser superior en comparación con toda la población.

Para esto, se categorizan los errores de acuerdo con lo que se muestra en la siguiente tabla:

Tabla 7: Categorización de errores

Error	Descripción
Errores de tipo 1	Asociados a la variable de día.
Errores de tipo 2	Asociados a la variable de hora.
Errores de tipo 3	Asociados a la variable de aula.

Teniendo distribuidos los errores, se debe considerar cuales deben ser corregidos primero, asignándoles un peso mayor en comparación con los otros tipos de errores para ellos se establece el siguiente criterio para resolver los errores y encontrar la solución óptima:

- Los errores de tipo 1 son los primeros que deben ser resueltos.
- Los errores de tipo 2 son los segundos en resolverse.
- Por último, los errores de tipo 3, asociados a las aulas son los últimos en ser resueltos.

Para ello, se plantea la siguiente formula:

$$fitness = \frac{1}{\{[x * (y + z)] + (y * z) + z\} + 1} \quad (2)$$

Donde las variables representan lo siguiente:

- x: Cantidad de errores de tipo 1.
- y: Cantidad de errores de tipo 2.
- z: Cantidad de errores de tipo 3.

5. Resultados y Discusión

Con el fin de evaluar el algoritmo desarrollado, se calibran los valores importantes que el algoritmo recibe, lo cual se puede observar en la siguiente tabla:

Tabla 8: Valores para la ejecución del algoritmo

Variable	Valor
Tamaño de la población	40
Número de generaciones	10,000
Probabilidad de cruza	95%
Probabilidad de mutación	30%
Tipos de aulas excluidas	Laboratorios

La asignación de estos valores fue al azar, sin embargo, se cumplen los principios sugeridos en varios artículos y libros referentes a algoritmos genéticos.

Así mismo con el fin de obtener resultados del funcionamiento del algoritmo se limitan las pruebas para el primer grupo del primer semestre de la carrera de Ingeniería en Sistemas.

Una vez habiendo delimitado las variables para las pruebas, se procede a ejecutar el algoritmo, teniendo una distribución de aptitudes de la primera generación representadas en la figura 9.

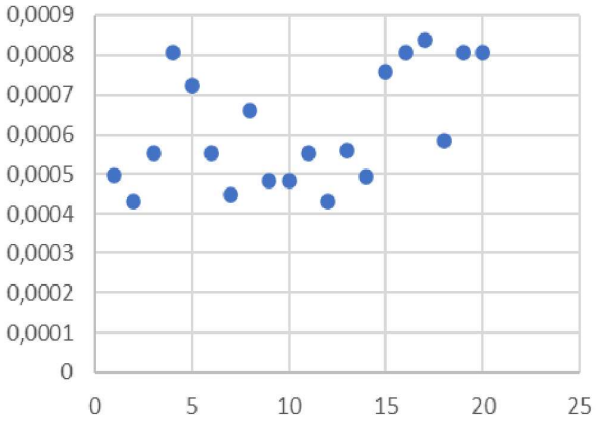


Figura 9. Aptitudes de los individuos de la primera generación.

Como se puede observar, la aptitud de los individuos es extremadamente baja, sin embargo, hay tres individuos que destacan sobre los demás, los cuales al momento de la selección ellos tendrán mayor probabilidad de ser elegidos.

Finalizada la ejecución del algoritmo, se tiene una solución viable, la cual cumple con todas las restricciones establecidas en el proyecto, siendo el individuo más óptimo.

De la misma se logra ver una evolución de todos los individuos como se muestra en la figura 10, en donde hay un claro constraste en comparación a la figura 9 donde tenemos los valores de las aptitudes de la primera generación.

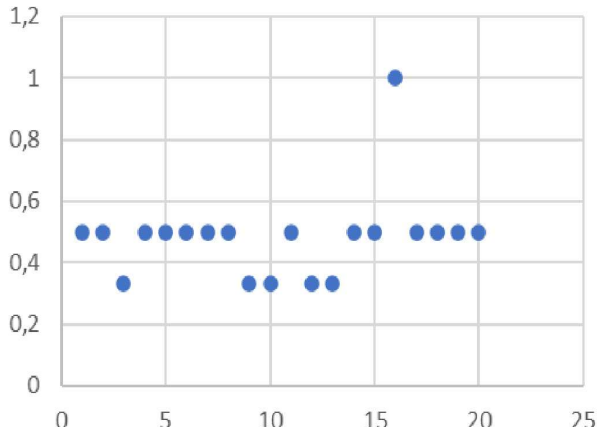


Figura 10. Aptitudes de los individuos de la última generación.

En la figura 11 se puede visualizar la evolución del individuo elitista, el cual a través de generaciones su valor de aptitud va evolucionando, obteniendo siempre una solución óptima hasta llegar al punto de la convergencia, la cual se da en la última generación.

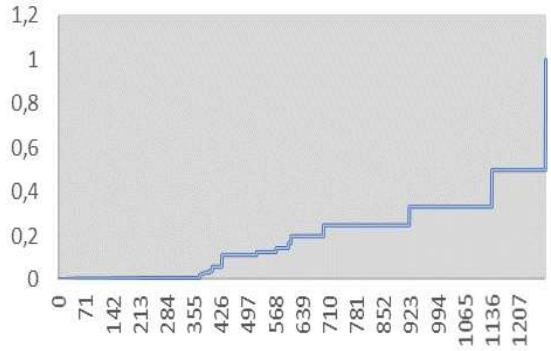


Figura 11. Evolución del individuo elitista a través de sus generaciones.

Con la solución obtenida en la ejecución del algoritmo genético, vemos como se obtiene un horario prestablecido y asignado a un aula para las clases, como se muestra en la tabla 4:

Tabla 9: Horario de Clases Resultante

Materia	Día	Hora	Aula
Algebra lineal	Lunes	17:00	Aula 02
Algebra lineal	Lunes	18:00	Aula 02
Algebra lineal	Martes	09:00	Aula 08
Algebra lineal	Martes	10:00	Aula 08
Calculo diferencial	Martes	11:00	Aula 08
Calculo diferencial	Martes	12:00	Aula 08
Técnicas de investigación	Miércoles	09:00	Aula 13
Técnicas de investigación	Miércoles	10:00	Aula 13
Calculo diferencial	Jueves	19:00	Aula 15
Calculo diferencial	Jueves	20:00	Aula 15
Antropología cristiana	Viernes	20:00	Aula 07
Antropología cristiana	Viernes	21:00	Aula 07
Algebra lineal	Sábado	11:00	Aula 01
Algebra lineal	Sábado	12:00	Aula 01

La solución obtenida muestra una optimización sustancial en varios ejes, los cuales se menciona a continuación:

- Optimización de tiempo de asignación de día, hora y aula para las diferentes materias, pues el tiempo de ejecución del algoritmo tiene un tiempo entre 3 a 8 minutos.
- La asignación correcta de aulas activas, y disponibles minimizando los posibles errores que pudiesen presentarse en una asignación manual.
- Corta distancia de desplazamiento de los estudiantes para recibir una segunda materia en un mismo día.
- Horas contiguas entre materias lo cual le permita al estudiante optimizar su tiempo libre en otras actividades académicas.

La optimización de los recursos usados por el algoritmo también tiene una mejora, los mismos se muestran en la siguiente tabla:

Tabla 10: Valores para la ejecución del algoritmo

Recurso	Valor Inicial	Valor Ejecución
Uso del CPU	5%	35%
Uso de RAM	42%	43%
Lectura del Disco Duro	0%	25%

Como se observa en la tabla 10, los únicos recursos que se usan son CPU y Disco Duro, sin embargo, el uso de la memoria RAM es mínimo, se consideraron los valores antes de comenzar la ejecución del algoritmo y la media mientras finalizaba el proceso.

Con estos resultados obtenidos podemos notar una mejora en el tiempo de ejecución con respecto al proyecto desarrollado anteriormente en donde se hace uso de un algoritmo Hill-Climbing [6] en donde el tiempo de ejecución de ejecución está en un rango de 10 a 20 minutos aproximadamente baja a un rango de 3 a 8 minutos con la solución propuesta, adicional a esto, este proyecto contempla no solo la asignación de aulas, sino la generación de los horarios de clases de manera automática.

De la misma en comparación al proceso manual de asignación en donde la inversión de tiempo es extremadamente alta, debido a que, para completar con el proceso, pueden pasar días por la intervención de varios actores en el proceso [6], los resultados del prototipo dan una solución mucho más factible y rápida, optimizando los tiempos y recursos.

6. Conclusiones

La implementación de este prototipo cumple con los objetivos planteados, el cual es, reducir el tiempo de asignación de aulas y horarios realizándose de manera automática, optimizando los recursos de la Universidad Politécnica Salesiana Sede Quito Campus Sur y reduciendo de manera significativa el desplazamiento de los estudiantes, ya que el algoritmo da como resultado final que todas las materias que se imparten en un mismo día se desarrollen en un mismo curso y las horas sean contiguas.

El algoritmo también se optimiza el uso de los recursos del computador, ya que, mediante la implementación en un sistema robusto de base de datos de SQL Server, el procesamiento se distribuye entre la memoria del sistema, el disco duro y el CPU.

Como trabajo futuro se podría optimizar aún más los tiempos, se podría analizar todos los procedimientos almacenados y funciones de la base de datos mediante el plan de ejecución con el fin de encontrar puntos de mejora y de esa manera lograr una efectividad y menor consumo de recursos dentro del sistema operativo, así mismo, la implementación de una interfaz gráfica la cuál permita administrar los parámetros al usuario final, y la ejecución del algoritmo implementado para la asignación automática de horarios y aulas.

Referencias

Libros:

[1] O. Kramer, Genetic Algorithm Essentials, Springer International Publishing, 2017.

[2] S. Mirjalili, «Evolutionary Algorithms and Neural Networks,» Springer International Publishing, 2019.

[3] S. Sivanandam y S. N. Deepa, Introduction to Genetic Algorithms, Springer-Verlag Berlin Heidelberg, 2008.

Artículos Académicos:

[4] M. A. Antunez, «Sistema para optimizar asignación de aulas UNICEN,» UNIVERSIDAD NACIONAL DEL CENTRO DE LA PROVINCIA DE

BUENOS AIRES, Buenos Aires, 2015.

[5] A. A. Constantino, W. M. Filho y D. Landa-Silva, «Iterated Heuristic Algorithms for the Classroom Assignment Problem,» University of Nottingham, 2010.

[6] P. Naranjo, J. Paspuel, R. Tufiño y H. Ortega, «Análisis e implementación de un algoritmo de optimización para la asignación automática de aulas clase en el Campus Sur de la Universidad Politécnica Salesiana,» Universidad Politécnica Salesiana, Quito, 2017.

[7] B. R. Rajakumar, «Static and adaptive mutation techniques for genetic algorithm: a systematic comparative analysis,» International Journal of Computational Science and Engineering, 2013.

Tesis / trabajos de titulación:

[8] J. A. A. AHUMADA, «GENERACIÓN DE HORARIOS ACADÉMICOS EN INACAP UTILIZANDO ALGORITMOS GENÉTICOS,» UNIVERSIDAD DE CHILE, SANTIAGO DE CHILE, 2014.

Sitios Web:

[9] T. Coronel, «Github,» 18 febrero 2020. [En línea]. Available: https://github.com/tacc91/Repositorio.git. [Último acceso: 18 febrero 2020].