UNIVERSIDAD CIENTÍFICA DEL SUR

Facultad de Ciencias Empresariales

Carrera Profesional de Ingeniería de Sistemas Empresariales

"ALGORITMO GENÉTICO PARA LA ELABORACIÓN DEL HORARIO DE EXÁMENES EN UNIVERSIDADES"

Tesis para optar el Título Profesional de

Ingeniero de Sistemas Empresariales

Presentado por:

BACH. HERNÁN ENRIQUE ZELADA CABEZUDO

(Lima Perú)

2018

ANEXO 13
ACTA DE SUSTENTACIÓN DE TESIS

Siendo las 17:00 horas del día 10 de diciembre del 2018, en la sala de conferencias se reunieron los miembros del Jurado:

Presidente	Mg. Chacón Cursack, Horacio Alfonso Jesús
Miembro 1	Mg. Ramos Muñoz, Alfredo Marino
Miembro 2	Dra. Meza Balvin, Sandra Jeannet

Para evaluar la tesis titulada: "ALGORITMO GENÉTICO PARA LA ELABORACIÓN DE HORARIOS DE EXÁMENES EN UNIVERSIDADES", presentada por el tesista Hernan Enrique Zelada Cabezudo para optar el Título Profesional de Ingeniero de Sistemas Empresariales.

Terminada la sustentación, el Jurado luego de deliberar concluyen de manera unánime (X) por mayoría simple ( ) que la tesis es:

Aprobado	( )
Aprobado - Muy buena	(X)
Aprobado - Sobresaliente	( )
Desaprobado	( )

Calificándola con nota de: 17 en letras (Diecisiete)

En fe de lo actuado los miembros de Jurado suscriben la presente Acta en señal de conformidad.

Presidente del Jurado
Mg. Chacón Cursack, Horacio Alfonso Jesús

Miembro del Jurado
Mg. Ramos Muñoz, Alfredo Marino

Miembro del Jurado
Dra. Meza Balvin, Sandra Jeannet

Asesora
Dra. Delgadillo Avila, Rosa Sumactika

Dedicatoria

A mis padres, porque creyeron en mí, dándome ejemplos dignos de superación y entrega. Hoy puedo ver alcanzada mi meta, ya que siempre estuvieron impulsándome en los momentos más difíciles de mi carrera porque admiro su fortaleza y por lo que han hecho de mí.

Agradecimientos

A Dios. Por haberme permitido llegar hasta este punto y haberme dado salud para lograr mis objetivos, además de su infinita bondad y amor.

A mis profesores y asesor que me apoyaron para que este trabajo salga bien.

Gracias a todos.

Índice general

Índice de gráfico

Índice de tablas

Resumen

Con el crecimiento de alumnos matriculados en universidades, se incrementa también las actividades administrativas que realiza cada casa de estudio; una de las actividades administrativas que se ve afectada por el aumento de alumnos es la elaboración del horario de examen, el cual se realiza dos veces en cada periodo académico regular. Elaborar el horario de exámenes demanda tiempo y recursos, considerándola una actividad de alta complejidad y en muchos de los casos por las restricciones que se debe tener en cuenta. Se debe considerar ciertos parámetros como: franjas horarias, distribución de recintos (salones, laboratorios), distribución de curso de cada carrera (dos cursos de distintas carreras deben estar en un mismo salón de clase), asignación de docentes o cuidadores, además de las restricciones que tienen que ser superadas para que el horario que se elabore sea aceptado como válido.

Para la presente tesis se utiliza una solución basada en algoritmo genético, que es una metaheurística; la cual pertenece a la rama de la inteligencia artificial. El cual es recomendado en la utilización de problemas de búsqueda y optimización; como el caso del horario de exámenes, para ello se diseña un cromosoma tipo vector binario. Teniendo en cuenta la problemática planteada en la tesis.

La función fitness es una fórmula matemática que nos brinda como resultado el valor cero, el cual nos certifica que el sistema genera un correcto horario de exámenes, teniendo en cuenta todas las restricciones en la elaboración.

Finalmente, la solución fue puesta a prueba para calibrar sus parámetros de generación de: población, selección, cruce y mutación; el cual arroja soluciones óptimas. Teniendo como resultados valores del fitness en promedio de 0.13665, en un rango de tiempo de 30 a 90 minutos, donde la solución arroja un horario válido por carrera.

Abstract

With the growth of students enrolled in universities, the administrative activities carried out by each study house are also increased; One of the administrative activities that is affected by the increase of students is the preparation of the exam schedule, which is done twice in each regular academic period. Preparing the examination schedule demands time and resources, considering it a highly complex activity and in many cases due to the restrictions that must be taken into account. It should consider certain parameters such as: time slots, distribution of venues (classrooms, laboratories), distribution of course of each race (two courses of different races must be in the same classroom), assignment of teachers or caregivers, in addition to the restrictions that have to be overcome so that the schedule that is elaborated is accepted as valid.

For the present thesis a solution based on genetic algorithm is used, which is a metaheuristic; which belongs to the branch of artificial intelligence. Which is recommended in the use of search and optimization problems; As in the case of the exam schedule, a binary vector type chromosome is designed for this. Taking into account the problems raised in the thesis.

The fitness function is a mathematical formula that gives us as a result the zero value, which certifies that the system generates a correct schedule of exams, taking into account all the restrictions in the elaboration.

Finally, the solution was put to the test to calibrate its generation parameters: population, selection, crossing and mutation; which throws optimal solutions. Taking as a result fitness values on average of 0.13665, in a time range of 30 to 90 minutes, where the solution yields a valid schedule per race.

Introducción

La generación de horarios de exámenes es una de las actividades más complejas que la casa de estudio afronta una o varias veces al año. La problemática radica en las restricciones que cuenta el horario, a eso le sumamos el consumo de tiempo y recursos en lo que se incurre, siendo en muchos de los casos varios días. Una solución basada en una Heurística simple podría ser incapaz de brindar buenos resultados, debido a que su estructura siempre mostrara los mismos resultados, para estos tipos de problemas una mejor opción es utilizar métodos basados en población, los cuales se tratan con varias soluciones y se van mejorándolas hasta que se obtiene la mejor solución posible. Los algoritmos genéticos pertenecientes a la familia de los métodos basados en población, permiten resolver problemas de búsquedas y optimización, como la creación de los horarios de exámenes. Para su mejor estudio será dividido en cinco capítulos.

CAPÍTULO I: denominado planteamiento del problema, se muestra la situación problemática actual que afronta la universidad al elaborar los horarios de examen, para ello tomaremos como referencia la universidad San Ignacio de Loyola, describiendo las restricciones con las que cuenta el horario; hecho que respaldará la formulación del problema general.

CAPÍTULO II: denominado Marco teórico, constituido por los antecedentes nacionales e internacionales sobre el tipo de problemática y el sustento teórico, que respalda la solución. Se determina lo objetivos y se formula las hipótesis de la tesis.

CAPÍTULO III: denominado diseño metodológico, se aborda las tipologías de la investigación, la población y muestra, la definición conceptual y operacionalización de las variables; además las técnicas de recolección de datos y aspectos éticos.

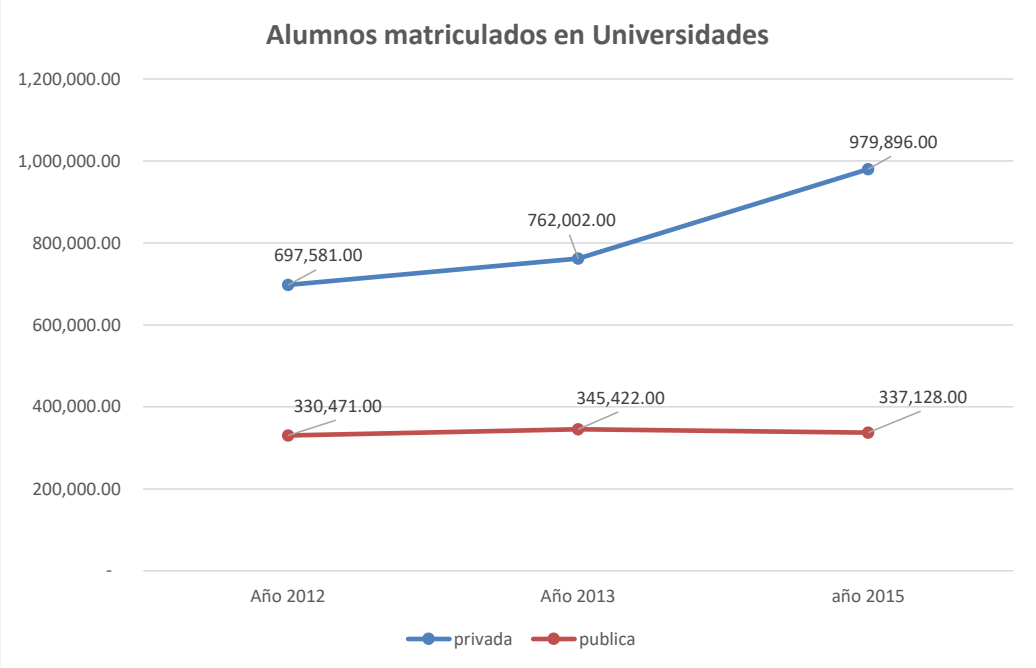
CAPÍTULO IV: denominado resultados, se detalla la solución al problema expuesto, que es la generación de horarios de exámenes, basado en el diseño de un algoritmo genético, que puede ser utilizado en un sistema computacional de generación de horarios.

CAPÍTULO V: denominado discusión, conclusiones y recomendaciones, Como corolario de esta investigación se ha considerado la discusión de los resultados de los experimentos, brindando también las principales conclusiones y recomendaciones de la presente investigación.

CAPÍTULO I: PLANTEAMIENTO DEL PROBLEMA

1.1. Descripción de la realidad problemática

Al inicio de cada año las universidades del Perú ofrecen carreras de alta demanda, que conllevan a un mayor número de alumnos matriculados en su institución. Esto genera un aumento de las actividades administrativas que realizan las facultades, entre las diversas actividades nos enfocaremos en la generación del horario de exámenes parciales y finales de los alumnos de pregrado, que en su mayoría se realiza en forma manual. Realizar esta tarea lleva varios días, ya sea por el volumen de datos que se maneja o las consideraciones que se debe contemplar, haciendo compleja la actividad. La tarea resulta ser costosa y no siempre culmina con un resultado satisfactorio de optimización de recursos. Anthony Wren (2005) citado por Pilot (2007, p.2) define este tipo de problemas como: “La asignación, sujeta a restricciones, de los recursos otorgados con el propósito de ser establecidos en un espacio de tiempo, de tal manera que satisfaga lo más cercanamente posible el conjunto de objetivos deseados”.



Fuente: INEI y SUNEDU

Gráfico 1: Crecimiento de alumnos matriculados en universidades del Perú

Un problema de horarios es un problema con cuatro parámetros: Un conjunto finito de franjas horarias, un conjunto finito de recursos, un conjunto finito de asignaciones, un conjunto finito de restricciones. El problema consiste en asignar tiempos y recursos a las aulas de manera que se satisfagan el mayor número posible de restricciones. (Moreno y Sánchez, 2017, pp 4).

Cabe señalar que cada universidad puede presentar un mayor o menor tipo de restricciones. Las cuales dividiremos en fuertes y débiles, teniendo en cuenta esta clasificación, agruparemos según los problemas más comunes que existan en las universidades.

Restricciones Fuertes: Estas restricciones se deben cumplir obligatoriamente, de lo contrario el horario de exámenes obtenido no será admitido como válido.

1. **Programación de un segundo horario de examen para un mismo curso**
   El aumento de nuevas carreras y las mejoras en los planes curriculares conlleva en un crecimiento de las asignaturas a dictarse para los alumnos. En muchos de los casos el aumento significa la apertura del curso en varios bloques de horarios (por la demanda de alumnos que tiene el curso). Por ello se dificulta la construcción del horario académico, debido a que el examen debe programarse para un mismo día y hora, tratando en lo posible no programar un segundo horario ya que el profesor tendría que elaborar otro examen con las mismas características del primero, entre ellas el nivel de dificultad, lo cual para los alumnos supone una desigualdad en el examen y en el tiempo de estudio.

2. **El uso de salones para dos cursos distintos en el mismo horario**
   Para evitar que los alumnos se copien durante el examen, se debe establecer que el aula a usar debe ser compartida como mínimo por dos cursos de distintas carreras. Se debe encontrar aulas disponibles con infraestructura adecuada, que se necesita en una determinada hora y día, sabiendo que tenemos 64 distintos tipos de aulas por cada curso (Aula-Física, Aula-Biol, Aula Arq, Aula-Cocina, etc.). El usuario encargado de la programación de horarios de exámenes tiene una ardua tarea, la cual debe verificar: la capacidad de cada aula, el número de alumnos matriculados, la cantidad de bloques creados por cada asignatura; por ello le dificulta la distribución del curso en los recintos correctos, haciendo que la distribución no sea optima en muchos de los casos.

3. **Cruces de exámenes**

   Plasmar el Horario de examen conlleva a tener que crear distintas franjas horarias (día y hora), donde las asignaturas puedan coincidir con los recintos separados y en los días que se ofrece el servicio educativo sin considerar feriados o domingos. Este proceso se debe plasmar de tal forma que ningún alumno debería tener algún cruce de examen por carrera. Se debe considerar el tiempo entre examen para que los alumnos se pueden trasladar de los locales o pabellones con la facilidad necesaria, sin generar algo perjudicial. Buscar la mejor franja horaria para evitar esta restricción requiere una inversión de tiempo significativo.

4. **Recintos adecuados según el curso**

   Al momento de crear los horarios de exámenes, se debe asignar los recintos adecuados según el curso “salones o laboratorios”, para que el alumno pueda rendir en forma cómoda su examen.

Restricciones Débiles: miden la calidad del horario**,** estas restricciones son de uso de mejoras en el horario de exámenes, atienden a condiciones de comodidad enfocados a los alumnos o Administrativos.

1. **Establecer el mismo número de alumnos para todos los días de examen**

   Contar con una misma cantidad de alumnos en los días que se rendirán los exámenes. El cual permitirá tener en orden las actividades a realizar por la parte administrativa, sin caer en el exceso de trabajo.

1.2. Formulación del problema

1.2.1. Problema general

A los problemas propuestos se pretender averiguar ¿De qué manera la complejidad de la elaboración del horario de examen, se puede simplificar aplicando algoritmos genéticos, para una universidad Privada?

1.2.2. Problemas específicos

1. ¿De qué manera las aulas que utilizan el horario de examen se puede optimizar, con el fin de que cada curso este en el aula correcta?

2. ¿De qué manera la elaboración del horario de examen puede ser agilizado, utilizando los algoritmos genéticos?

1.3. Justificación de la investigación

La generación del horario de exámenes es una actividad que todas las universidades realizan, con esta investigación se pretende reducir el tiempo de generación del horario de exámenes, eliminar errores en el proceso manual y el mejor uso de salones los cursos de cada examen, que realizándolo en un proceso empírico puede tomar días conseguirlo. Liberando horas hombres del personal administrativo para que pueda apoyar en otras funciones. El uso de algoritmos basado en técnicas tradicionales como búsquedas exhaustivas o programación lineal, no garantizan la satisfacción de las restricciones. (Pilot, 2007, pp. 1). Esto se debe a que cada periodo académico se le puede sumar restricciones nuevas. Para ello se diseñará y desarrollará un algoritmo basado en la técnica evolutiva que sea capaz de resolver estos tipos de problemas de optimización que satisfagan ciertas restricciones, determinando así un horario aceptable. Esto conlleva a ciertos beneficios de los recursos de la universidad.

1. **Reducción de las actividades administrativas**
   Contar con el Horario de exámenes optimizado ayuda a planificar las actividades del día, trayendo como resultado menor carga de trabajo y disminución de las horas hombre del personal administrativo. Este tiempo se pueden emplear para otros fines.

2. **Optimización de la Infraestructura**

   Saber con semanas de anticipación los recursos disponibles en cuento a la infraestructura de la universidad permite poder organizar en forma adecuada los recintos disponibles para beneficios de la institución, trayendo rentabilidad en el uso de infraestructura.

3. **Disponibilidad de los cuidadores de exámenes**

   En la mayoría de universidades y por motivos económicos, la semana de exámenes los profesores son remplazados por cuidadores de exámenes. Contar con la disponibilidad de los cuidadores es una tarea muy importante, saber la cantidad exacta de cuidadores que se necesiten, los horarios y recintos donde estarán distribuidos es información que debería estar disponible con días o semanas de anticipación.

1.4. Limitaciones de la investigación

En la realización de la investigación solo se tomará como muestra una universidad privada de Lima, pudiendo no tener todas las casuísticas de la generación de los exámenes.

La investigación se limita a solo la creación del horario de exámenes optimizado a los recursos de la universidad, teniendo el supuesto que las facultades comparten una misma sede para el dictado de sus cursos y el número de alumnos no supera la infraestructura. La investigación no discutirá puntos, como la optimización de horarios académicos en el rendimiento de su infraestructura.

La investigación solo construirá el algoritmo que se compilará para que pueda ser integrado a cualquier sistema académico de una universidad.

1.4.1 Delimitación del estudio

1. **Espacio**

   Para la presente investigación de estudio, la recopilación de datos y especificaciones de los requisitos se llevara mediante un análisis de levantamiento de información de la universidad San Ignacio de Loyola en el área de registros académicos. La cual se encuentra en su campus principal en la Molina (Lima-Perú), de esta manera se tendrá un conocimiento más

claro de la elaboración del horario de exámenes parciales y finales. La información usada es permitida y distribuida para fines de esta tesis y será viable realizar dicho análisis.

2. **Temporal**

   El levantamiento de información se realizará con los 2 últimos periodos (2017-01 y 2017-02) y en los meses donde generan los horarios, que son: marzo a julio y agosto a diciembre.

3. **Conceptual**

   Se estudiará el horario de exámenes, solo para el programa de carreras universitarias (pregrado) y en la modalidad de ingreso regular y beca 18.

   Se tomará en cuenta todas las facultades, debido a que la universidad está organizada por sedes, en donde se pueden dictar clase de distintas carreras.

   Teniendo esta información disponible se podrá desarrollar un algoritmo genético que permita poder ser reutilizado en otras universidades peruanas.

1.5. Viabilidad de la investigación

En cuanto a la viabilidad del proyecto a investigar, la generación del horario académico es una actividad que toda universidad realiza. Para resolver esta dificultad de horarios se cuenta con artículos publicados en revistas científicas y tesis de investigación sobre la problemática que existe en la generación de horarios académicos y horario de exámenes. La cual utilizan la técnica de algoritmos evolutivos también conocidos como algoritmos genéticos, esta información se tomará como referencia para diseñar y desarrollar un algoritmo capaz de generar el horario de exámenes de universidades privadas y orientado a la optimización de recursos. El uso de algoritmos genéticos es usado para solucionar problemas de optimización, donde se ha mostrado ser muy eficientes y confiables.

CAPÍTULO II: MARCO TEÓRICO

2.1. Antecedentes de la investigación

1. Antecedentes nacionales

En la revisión de los antecedentes nacionales, se encontraron tesis que utilizan los algoritmos genéticos (AG), para la optimización de horarios académicos de universidades peruanas. Las cuales se mencionan a continuación:

1.1 Castro, H. (2016), Tesis "Algoritmo genético aplicado en la programación académica de una Escuela Profesional en una Facultad Universitaria", Universidad del Callao.

Problemática: Con el aumento de los alumnos y cursos en la facultad, se hace complejo poder construir en forma óptima los horarios académicos que utilicen eficientemente la asignación de aulas y docentes, por lo cual se necesita saber si los algoritmos genéticos pueden ayudar a sistematizar este proceso.

Objetivos: Desarrolla un sistema inteligente basado en algoritmos genéticos que permita mejorar y automatizar la asignación de aulas y docentes en la programación académica de las escuelas profesionales de Ingeniería Industrial.

Metodología: Es un tipo de investigación descriptiva, correlacional.

Resultados: El algoritmo alcanza un 91% de efectividad en asignación de aulas y docentes. El algoritmo genético muestra los primeros resultados válidos a los 10 minutos en promedio a comparación de expertos, que los primeros resultados lo tienen en 3 a 5 días.

Conclusión: Aplicando la combinación de técnicas de algoritmos entre la búsqueda tabú y algoritmo genético recomendado por R. raghavjee and N. Pillay en su publicación del libro "An Application of Genetic Algorithms to the School Timetabling Problema SAICSIT 2008". Moldea el algoritmo a la realidad de restricciones que tiene la facultad.

Concluyendo que para que los algoritmos sean eficientes tienen que ser adaptados, de lo contrario no se podrá realizar con eficiencia la optimización deseada.

1.2 Blaz, S. (2016), Tesis "Un sistema de generación de horarios para la enseñanza de pregrado en universidades peruanas mediante algoritmos genéticos", universidad de San Marcos.

Problemática: Complejidad de la programación de horarios, por el gran número de restricciones que son definidas de forma particular por cada institución, sin poder asignar en forma óptima los recursos (salones, docentes, periodo).

Objetivos: Diseñar un sistema para generar en forma automática los horarios para la enseñanza en las universidades, considerando los recursos disponibles; que a diferencia de los horarios elaborados actualmente permitan cumplir con todas las restricciones obligatorias.

Metodología: Es un tipo de investigación descriptiva, correlacional.

Resultado: Para la elaboración del horario de clase, el sistema obtuvo buenos resultados a los 10 minutos de generación del horario, mostrando que se había considerado todas las restricciones fuertes y las mejores restricciones débiles propuestas para el desarrollo, en comparación a lo realizado manualmente que tiene una demora de varias semanas.

Conclusión: Utiliza el algoritmo genético de búsqueda guiada, denominado GSGA, para resolver el UCTP ("University Course Timetabling Problem") , que consiste en una estrategia de búsqueda guiada y una técnica de búsqueda local, que fue propuesto por Shengxiang Yang, Member y Sadaf Naseem Jat en la publicación del 2019 en Reinos Unido por la Universidad de Leicester. Que dieron buenos resultados para resolver estos tipos de problemas.

1.3 Cortez, A., Rosales, G., Naupari, R. & Vega. H. (2010), Revista de investigación de sistemas e informática "Sistema de apoyo a la

generación de horarios basado en algoritmos genéticos", universidad San Marcos.

Problema: al inicio de cada periodo académico se presenta la necesidad de asignar y coordinar los recursos económicos, materiales y humanos en beneficios de los estudiantes. La creación de los horarios académicos se realiza en forma manual, estimando y estipulando los recursos como mejor se parezca. Esta tarea normalmente requiere varios días de trabajo, teniendo en cuenta el volumen de datos y variables que intervienen en el proceso, la tarea resulta ser cara, complicada y no siempre culmina con un resultado satisfactorio.

Objetivos: Generar un procedimiento basado en algoritmo genético que solucione los problemas de la confección de horarios.

Metología: tecnológico, correlacional

Resultados: el artículo no muestra tiempos de ejecución a comparación de lo realizado en forma manual, pero se indica que utilizando la fase de órdenes diseñadas se puede obtener un horario adecuado y con margen de errores aceptables.

Conclusiones, Para la investigación utilizan la propuesta Original de Goldberg (1989), conocido como algoritmos genéticos simples.
Proponen 3 fases para la creación de horarios: asignación de docentes a las clases, asignación de horarios a las clases y asignación de aulas a las clases.

2. Antecedentes Internacionales

En los antecedentes internacionales se encontraron trabajos muy similares a la investigación realizada sobre la creación del Horario de exámenes, que se usarán como base para adaptarlo a la realidad de las universidades privadas peruanas.

2.1 Moreno, P. & Sánchez, J. (2017), Revista Tecnológica "Revisión de algoritmos de búsqueda aplicadas al problema de creación de Horarios de exámenes", País España.

Problema: la creación del horario de exámenes, en donde las asignaturas deben plasmarse en una franja horaria (día y hora), teniendo en cuenta que a ningún alumno debería tener cruces de exámenes para una misma carrera y ciclo. Que pueda atender las restricciones que mejoren la calidad del horario, ateniendo a uso de equipos, disponibilidad del profesor o vigilantes para el cuidado de los exámenes, aulas con capacidad suficiente, etc.

Objetivos: realizar una revisión con los distintos algoritmos de resolución del problema de generación de horarios de exámenes.

Metología: Observacional

Resultados: En este artículo científico se muestra las ventajas y desventajas de las distintas técnicas de algoritmo que existen para la resolución del problema sobre creación de horarios

Conclusión: De todos los algoritmos que se considera en esta revista se puede observar distintas técnicas de solución al problema de creación de horarios de examen. En las cuales destaca los algoritmos genéticos (AG) y el algoritmo de búsqueda tabú (Tabu Search) como los más utilizados, seguido de ello se encuentran las técnicas Hill-Climbing y Simulated Annealing, como alternativas.

Se recomienda utilizar técnicas híbridas que utilizan un sistema de 2 o 3 fases, donde se construye una solución que cumpla las restricciones fuertes y se afina usando una técnica de optimización. Se tomarán en cuenta estas recomendaciones que realizan Moreno y Sanchez para la optimización del Horario de exámenes enfocado a la realidad Peruana.

2.2 Caballero, R. (2012), trabajo de investigación "Aplicación generadora de Horarios de exámenes", país España

Problema: La tarea de planificar exámenes para una titulación de la universidad es un trabajo complicado, al que tienen que enfrentarse los profesores año tras año. Es bastante tedioso el tener que asignar los

exámenes de las asignaturas de una titulación de forma que el mayor número de alumnos quede satisfecho con el resultado.

Objetivos: Implementar una aplicación que permita generar los Horarios de exámenes del primer cuatrimestre, el segundo, y la convocatoria extraordinaria, de las dos titulaciones “Doble grado en informática – ADE” y “Grado en ingeniería informática”.

Resultados: Se realizaron varias pruebas de generación del Horario de exámenes, gracias al algoritmo utilizado en los tiempos promedios fueron de 36 segundos, que es un tiempo muy bueno para la investigación

Conclusión: Se ha desarrollado una solución para la generación automática de Horarios de exámenes de la universidad, utilizando la técnica de (Hill-Climbing) el cual es un algoritmo voraz adaptado, capaz de poder distinguir diversas restricciones y de cumplirlas, en el mayor grado posible. Esta solución cumple de forma satisfactoria con todos los requerimientos establecidos por el cliente.

2.2. Bases teóricas

A continuación se hace un resumen de las técnicas usadas para la resolución de los horarios de exámenes. La idea es comprender la importancia que se ha dado en el trascurso del tiempo para encontrar una solución a este problema.

1. **Algoritmos aplicados a la solución del problema de creación de horarios de exámenes**

La investigación de la problemática de creación de horarios de exámenes se inició en los años 80 y desde entonces surgieron varias propuestas de algoritmos diferentes para abordar el problema. Las resoluciones propuestas se clasifican en dos métodos: trayectorias y población.

A continuación, se hace un resumen de los principales métodos que se usan para la generación de Horario de exámenes.

Métodos basados en trayectorias,

Son aquellos métodos que utilizan una solución única y exploran de forma aleatoria el espacio de estados, cada vez encontrando mejores soluciones hasta que se llega a la solución esperada que se le conoce como parada determinada. Entre los principales algoritmos tenemos lo siguiente:

- **Hill-Climbing (HC)**

  Llamado también ascenso de colinas. Perteneciente a los algoritmos de búsqueda local. Es un algoritmo de interacciones, se selecciona una solución candidata a un problema, luego se intenta encontrar una mejor solución variando incrementalmente un único elemento de la solución. Si el cambio produce una mejor solución, otro cambio incremental se le realiza a la nueva solución, repitiendo este proceso hasta que no se puedan encontrar mejoras. Suele llamarse a esta búsqueda Algoritmo voraz local, porque toma un estado vecino "bueno" sin pensar en la próxima acción.

  Por ejemplo (Merlot, Boland, Hughes & Stuckey, 2003) lo utiliza para resolver el problema de horarios de exámenes un método en varias fases que incluye programación con restricciones, Simulated annealing y Hill-Climbing.

- **Simulated Annealing (SA)**

  Fue propuesto por Kirkpatrick en 1983, es un algoritmo de búsqueda meta-heurística para problemas de optimización global; el objetivo general de este tipo de algoritmos es encontrar una buena aproximación al valor óptimo de una función en un espacio de búsqueda grande. A este valor óptimo se lo denomina "Óptimo global"

  SA se ha empleado con éxito en muchas áreas, entre ellas la generación de horarios de exámenes. (Thompson & Dowsland, 1995) resolvieron el problema en dos fases, una fase constructiva, generando una solución inicial factible, y una de fase de mejora, en ella mejorando la calidad de la solución. Usaron un proceso de enfriamiento adaptativo. Sus resultados mostraban la mejora de la

calidad, usando un enfriamiento geométrico. (Thompson & Dowsland, 1996, 1998) investigaron con distintos enfriamientos y Generación de la vecindad. Los resultados mostraban que la población de cadenas de Kempe (kempe chains) generaba las mejores soluciones. El resultado se debe a que este proceso permite generar un mayor número de exámenes, que moverlos; lo que redunda en una mejora de la calidad de la solución.

- **Tabu Search** (TS)
  También llamado búsqueda tabú, es un método propuesto por (Glover, 1986) que funciona de forma muy parecida a Hill-Climbing. Es un método de optimización matemática, perteneciente a la clase de técnicas de búsqueda local. La búsqueda tabú aumenta el rendimiento del método de búsqueda local mediante el uso de estructuras de memoria: una vez que una potencial solución es determinada, se la marca como "tabú" de modo que el algoritmo no vuelva a visitar esa posible solución. Según (Glover & Laguna, 1997) TS es “una metaheurística que guía a un proceso de búsqueda heurístico local para explorar el espacio de soluciones más allá del óptimo local”.

  Wilke & Ostler, 2010, aplicaron la búsqueda tabú al problema de horarios académicos y lo compararon con varios métodos (Simulated annealing, algoritmos genéticos y Branch & bound) para crear un programa que sea capaz de resolver distintos problemas de horarios. SA generaba los mejores resultados, pero TS era capaz de generar un buen resultado en muy poco tiempo. (Mushi, 2006) implementó un algoritmo TS para generar horarios de clases y así minimizar el incumplimiento de restricciones.

- **Great Deluge Algoritmo** (GDA)
  Es un algoritmo genérico aplicado a problemas de optimización. Es similar en muchos aspectos a los algoritmos de hill-climbing y simulated annealing, La diferencia principal con SA es que utiliza un límite superior, denominado normalmente nivel de agua (WATER-LEVEL), como límite de aceptación.

Burke, Bykov, Newall & Petrovic, 2004, implementaron un GDA con tiempo predefinido para el problema de horarios de exámenes. El algoritmo incluye dos parámetros de ajuste: el tiempo de cómputo y una estimación del coste de la solución. La velocidad de reducción se calcula como la diferencia entre la solución inicial y la solución deseada dividida por el tiempo de cómputo. Según sus autores este algoritmo genera buenas soluciones.

- **Variable Neighbourhood Search** (VNS)

  Es un método metaheurístico para resolver un conjunto de optimización combinatoria y problemas de optimización global. Explora los vecindarios distantes de la solución titular actual, y se mueve de allí a uno nuevo si y solo si se realiza una mejora. El método de búsqueda local se aplica de forma repetida para obtener soluciones desde el vecindario hasta óptimas locales.

  Burke, Eckersley, McCollum, Petrovic & Qu, 2010, hibridan VNS con algoritmos genéticos. Investigan el uso de diferentes estructuras de vecindad, entre las que se incluyen: ascenso/descenso que acepta movimientos a peor con una cierta probabilidad, VNS con límites que implica mover un examen que genera una gran penalización.

  Vecindades específicas del problema que implica reducir el número de vecindades y diferentes estrategias de inicialización, como por ejemplo inicialización ávida o aleatorizada. Del análisis estadístico que realizan se demuestra que los problemas son dependientes de sus vecindades, donde ciertas vecindades pueden generar mejoras en un problema, pero no en otros.

Los métodos basados en poblaciones, tratan con varias soluciones y van mejorándolas hasta obtener la mejor solución posible. Entre los principales de este tipo de métodos están los siguientes:

- **Genetic Algorithm** (GA)

  Son un mecanismo muy popular de búsqueda basado en poblaciones que utiliza la teoría de la evolución biológica para generar mejores soluciones de generación en generación. Este

método utiliza los conocidos operadores biológicos de selección, cruce y mutación para manipular las soluciones, que se denominan cromosomas en el algoritmo. Estas operaciones se utilizan durante un número de generaciones para ir mejorando el coste de las soluciones.

(Corne, Fang, Mellish & Corne, 1993) utilizan un GA para el problema de horarios de exámenes. Utilizan un cromosoma, cuyo tamaño es el número de exámenes. Para evitar soluciones imposibles proponen en (Ross & Corne, 1995) utilizar sólo el operador de mutación para generar soluciones viables. Esta modificación genera mejores resultados que un operador de cruce uniforme. Añaden también un mecanismo de reparación que resuelve posibles soluciones imposibles debidas a la representación del cromosoma.

A pesar que en algunos casos el GA tiende a demorar en la resolución del problema esto lo compensa al generar varias distintas soluciones cercanas al óptimo simultáneamente.

- **Memetic Algorithm (MA)**

  Los algoritmos genéticos realizan la búsqueda en todo el espacio de estados sin centrarse en una parte concreta del mismo, lo que puede hacer que se pierda información muy valiosa de uno de los individuos Sin embargo, la ventaja de los algoritmos genéticos es que realizan la búsqueda en muchas direcciones y desde muchos puntos distintos utilizando un conjunto de soluciones candidatas, lo que puede resultar una ventaja si se incluye un proceso de búsqueda local para mejorar el cromosoma solución. A este proceso general sobre búsqueda de poblaciones se le conoce como algoritmo memético

  En (Hung, Binh & Anh, 2005) se presenta una modificación al algoritmo de (Burke,Newall & Weare, 1996) en el que generar horarios de exámenes para la Ho Chi Minh City University of Technology. Aplican los mismos operadores evolutivos que Burke, pero añaden HC con penalización y con restricciones. Su modificación genera buenas soluciones pero a costa de tardar un mayor tiempo.

- **Ant Colony Optimisation (ACO)**

  En ciencias de la computación y en investigación operativa, el algoritmo de la colonia de hormigas, algoritmo hormiga u optimización por colonia de hormigas (Ant Colony Optimization, ACO) es una técnica probabilística para solucionar problemas computacionales que pueden reducirse a buscar los mejores caminos o rutas en grafos.

  (Dowsland & Thompson, 2005) investigan el uso de ACO al problema de horarios de exámenes. El objetivo de la investigación era por una parte comprobar cómo se comportaba la herramienta ANTCOL en grafos de horarios con los conjuntos de grafos que habían creado (Costa & Hertz, 1997) y, por otra parte, identificar combinaciones de Heurísticas constructivas que funcionasen con el problema. Los resultados experimentales mostraban que las propuestas para la herramienta ANTCOL aplicada al problema de horarios de exámenes resultaban competitivas con otras estrategias publicadas minimizando el número de franjas horarias necesarias para generar horarios factibles.

- **Particle Swarm Optimization (PSO)**

  La optimización por nube de partículas u optimización por enjambre de partículas conocida por sus siglas en inglés: PSO, hace referencia a una metaheurística que evoca el comportamiento de las partículas en la naturaleza. En este algoritmo los elementos que se manejan son partículas que se mueven en un espacio multidimensional. Cada una de las partículas tiene una velocidad y una mejor posición que ha alcanzado en un momento dado. En este algoritmo se utiliza una población, un enjambre, en donde cada partícula representa una posible solución. El enjambre se inicializa de forma aleatoria. El proceso de búsqueda se da realizando cambios en la velocidad y posición de cada una de las partículas, según se generan nuevas generaciones. De hecho, las partículas se mueven en un espacio de búsqueda Ndimensional siguiendo a la mejor partícula del enjambre. En cada iteración se evalúa la posición de cada una de las partículas. Cada una de las partículas guarda la

mejor posición que haya encontrado en algún momento Pbest y la mejor posición de la mejor partícula del enjambre, llamada Gbest

2. Definición del algoritmo genético

A continuación se describe las distintas definiciones que cada investigador ha especificado según su punto de vistas.

John Holland (1975)

Indica que "Son procedimientos sistemáticos basados en la selección natural de los seres vivos y el paso de información genética generación a generación."

DIAZ (1996)

Indica que "Un algoritmo genético es una estructura de control que organiza o dirige un conjunto de transformaciones y operaciones diseñadas para simular los procesos de evolución."

KOZA (1992)

Define que "Es un algoritmo matemático altamente paralelo que transforma un conjunto de objetos matemáticos individuales con respecto al tiempo usado operaciones modeladas de acuerdo al principio Darwiniano de reproducción y supervivencia del más apto, y tras haberse Presentado de forma natural una serie de operaciones genéticas de entre las que destaca la recombinación sexual."

WHITLEY (1997)

Un algoritmo genético está compuesto por: "1) Módulo evolutivo, presenta un mecanismo de decodificación que se encarga de interpretar la información de un individuo y una función de evaluación que mide la calidad del mismo. 2) Módulo poblacional, posee una representación poblacional y las técnicas necesarias para poder manipularla como son la técnica de representación, el criterio de selección y de reemplazo. Aquí también se define el tamaño de la población y la condición de terminación. 3) Módulo reproductivo, contiene los operadores genéticos."

3. Historia de los algoritmos evolutivos

Los primeros ejemplos de algoritmos genéticos aparecieron a principios de los años 60, programados en computadoras por biólogos evolutivos que buscaban explícitamente realizar modelos de aspectos de la evolución natural. A ninguno de ellos se le ocurrió que esta estrategia podría aplicarse de manera más general para la solución de diversas índoles, pero ese reconocimiento no tardaría en llegar, de hecho en palabras de Melanie Mitchell “La computación evolutiva estaba definitivamente en el aire en los días formativos de la computadora electrónica”.

En 1962 H.J. Bremermann había desarrollado algoritmos inspirados en la evolución para optimización de funciones y aprendizaje automático, pero sus trabajos generaron poca reacción.

En 1965 surgió un desarrollo más exitoso, cuando Ing. Rechenberg y Schwefel dieron a conocer lo que llamaron “estrategias evolutivas”
. En esta técnica no había población ni cruzamiento; un padre mutaba para producir un descendiente, y se conservaba el mejor de los dos, convirtiéndose en el padre de la siguiente ronda de mutación. Versiones posteriores introdujeron la idea de población.

El siguiente desarrollo importante en el campo vino en 1966, cuando L.J. Fogel, A.J. Owens y M.J. Walsh, introdujeron en América una técnica que llamaron programación evolutiva. En este método, las soluciones candidatas para los problemas se representaban como máquinas de estado finito sencillas; al igual que en la estrategia evolutiva de Rechenberg, su algoritmo funcionaba mutando aleatoriamente una de estas máquinas simuladas y conservando la mejor de las dos. También al igual que las estrategias evolutivas, hoy en día existe una formulación más amplia de la técnica de programación evolutiva que todavía es un área de investigación en curso. Sin embargo, lo que todavía faltaba en estas dos metodologías era el reconocimiento de la importancia del cruzamiento.

Un investigador de la universidad de Michigan llamado John Holland era consciente de la importancia de la selección natural, y a fines de los 60s desarrolló una técnica que permitió incorporarla a un programa. Su objetivo

era lograr que las computadoras aprendieran por sí mismas. A la técnica que inventó Holland se le llamó originalmente "planes reproductivos", pero se hizo popular bajo el nombre "algoritmo genético" tras la publicación de su libro en 1975. Holland fue también el primero en proponer explícitamente el cruzamiento y otros operadores de recombinación.

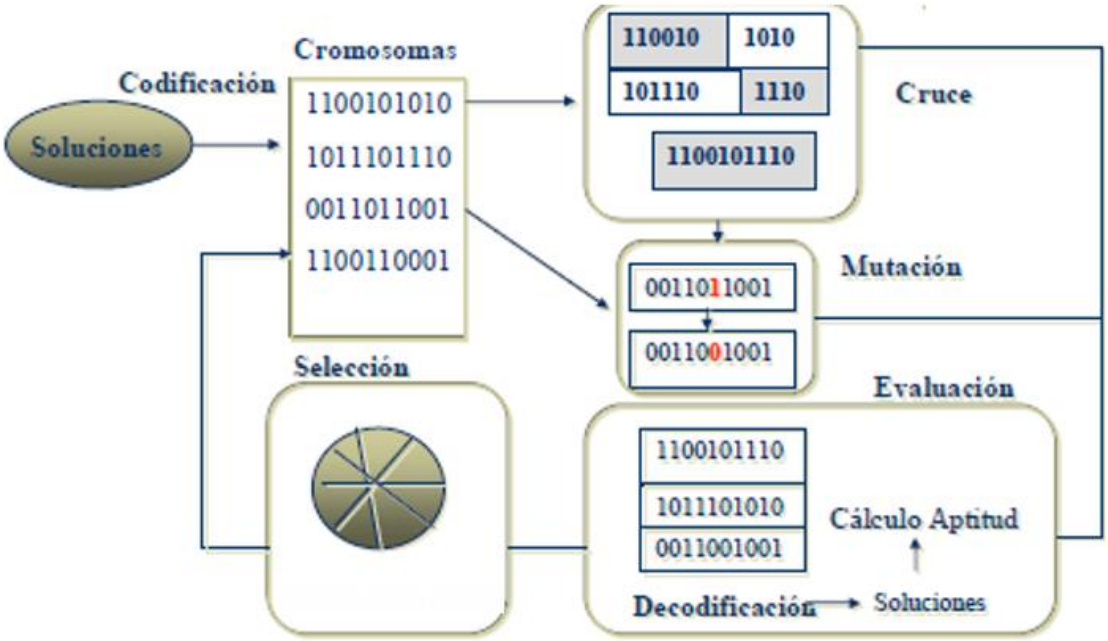
4. Esquema Básico

En la naturaleza todo el proceso de evolución biológica se hace de forma natural pero para aplicar el algoritmo genético al campo de la resolución de problemas habrá que seguir una serie de pasos. Una premisa es conseguir que el tamaño de la población sea lo suficientemente grande para garantizar la diversidad de soluciones. Se aconseja que la población sea generada de forma aleatoria para obtener dicha diversidad. En caso de que la población no sea generada de forma aleatoria habrá que tener en cuenta que se garantice una cierta diversidad en la población generada. Los pasos básicos de un algoritmo genético son:

- Evaluar la puntuación de cada uno de los cromosomas generados.
- Permitir la reproducción de los cromosomas siendo los más aptos los que tengan más probabilidad de reproducirse.
- Con cierta probabilidad de mutación, mutar un gen del nuevo individuo generado.
- Organizar la nueva población.

Estos pasos se repetirán hasta que se dé una condición de terminación. Se puede fijar un número máximo de iteraciones antes de finalizar el algoritmo genético o detenerlo cuando no se produzcan más cambios en la población (convergencia del algoritmo). Esta última opción suele ser la más habitual.

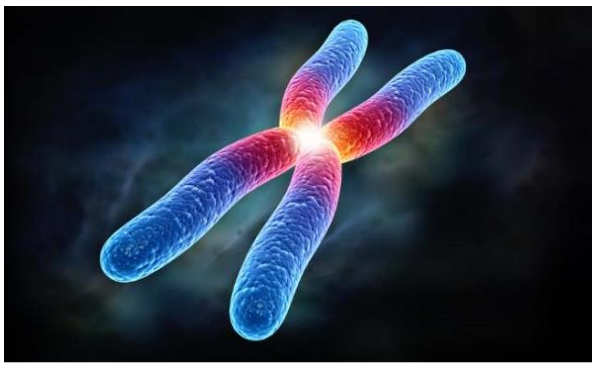
5. Estructura de un algoritmo genético



Fuente: https://es.scribd.com

Gráfico 2: Representación de la estructura de un algoritmo genético

Todos los organismos vivos están constituidos por células y cada célula contiene uno o más cromosomas (cadenas de ADN), que le sirven como una especie de plano al organismo. Un cromosoma puede ser conceptualmente dividido en genes cada uno de los cuales codifica una proteína. En términos generales, se puede decir que un gen se codifica como si fuera un rasgo, como puede serlo el color de ojos. Cada gen se encuentra en una posición particular del cromosoma y está formado por alelos.

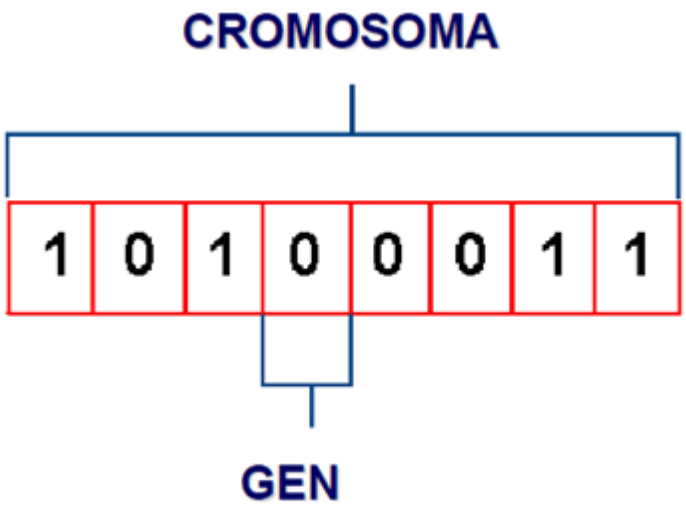


Fuente: https://respuestas.tips/

Gráfico 3: Representación de un cromosoma Humano

Debemos disponer de un mecanismo para codificar un individuo como un genotipo. Una vez elegida una representación, hemos de tener en mente como los genotipos (codificación) serán evaluados y qué operadores genéticos hay que utilizar.

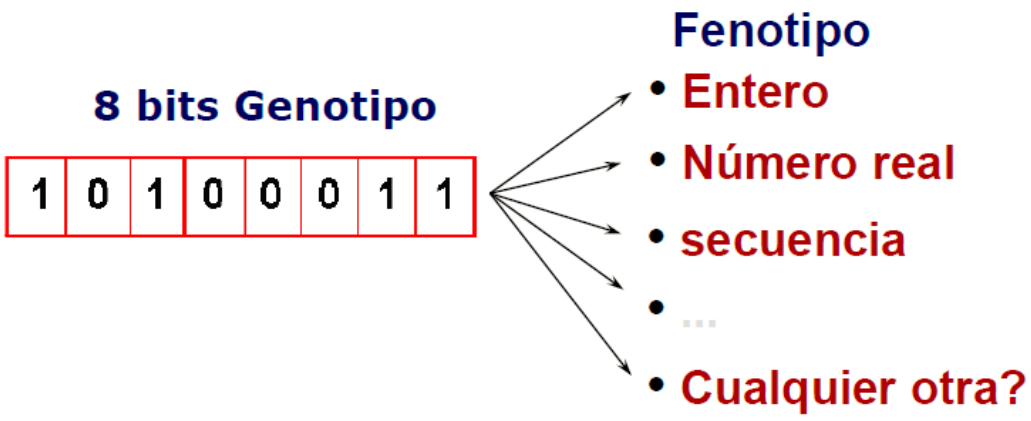
La representación de un individuo se puede hacer mediante una codificación discreta, y en particular binaria.



Fuente: http://sci2s.ugr.es/

Gráfico 4**:** Representación de un cromosoma Binario

En términos biológicos, el conjunto de parámetros representando un cromosoma particular se denomina Fenotipo. El fenotipo contiene la información requerida para construir un organismo, el cual se refiere como genotipo. Los mismos términos se utilizan en el campo de los algoritmos genéticos. La adaptación al problema de un individuo depende de la evaluación del genotipo. Esta última puede inferirse a partir del fenotipo, es decir puede ser computada a partir del cromosoma, usando la función de evaluación.



Fuente: http://sci2s.ugr.es/

Gráfico 5: Representación de un Genotipo de 8 bits

Codificación, los elementos característicos del problema se pueden representar de tal forma que resulte sencilla su implementación y compresión.

Población, nos indica el número de cromosomas que tenemos en nuestra población para una generación determinada. En caso de que esta medida sea insuficiente, el algoritmo genético tiene pocas posibilidades de realizar reproducciones con lo que se realizaría una búsqueda de soluciones escasa y poco óptima. Por otro lado si la población es excesiva, el algoritmo genético será excesivamente lento.

Función Fitness: asigna a cada cromosoma un número real, que refleja el nivel de adaptación al problema del individuo representado por el cromosoma.

Selección, es el proceso por la cual se eligen uno o varias parejas de individuos de la población inicial para que desempeñen el papel de progenitores.

Entre los tipos de selección podemos utilizar:

- **Selección Directa**, toma elementos de acuerdo a un criterio objetivo, como son “los x mejores”, “los x peores”, “el cuarto individuo

a partir del último escogido" son empleados con mucha frecuencia cuando se quieren seleccionar dos individuos distintos y se selecciona el primero por unos métodos aleatorios o estocásticos.

- **Selección aleatoria:** puede ser realizado por selección equiprobable o selección estocástica.
  Selección equiprobable: todos tienen la misma probabilidad de ser escogidos.
  Selección estocástica: la probabilidad de que un individuo sea escogido depende de una heurística

- **Por torneo:** escoge un subconjunto de individuos de acuerdo con una de las técnicas anteriores habitualmente aleatorias a estocástica y de entre ellos selecciona el más adecuado por otra técnica habitualmente, determinística de tipo "el mejor" o "el peor"

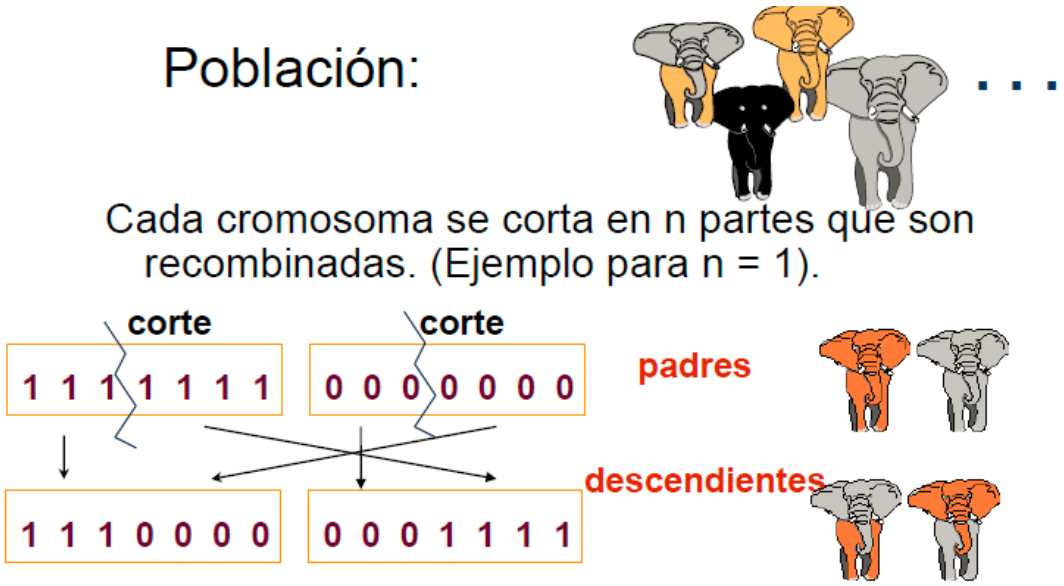
Cruce, Indica la frecuencia con la que se producen cruces entre los cromosomas padre es decir, que haya probabilidad de reproducción entre ellos. En caso de que no exista probabilidad de reproducción, los hijos serán copias exactas se los padres. En caso de haberla, los hijos tendrán partes de los cromosomas de los padres. Si la probabilidad de cruce es del 100% el hijo se crea totalmente por cruce, no por partes.

Entre los tipos de cruce podemos utilizar:

- **Cruce Básico:** se selecciona un punto al azar de la cadena. La parte anterior del punto es copiado del genoma del padre y la posterior de la madre

- **Cruce Multipunto:** igual que el cruce básico, solo que estableciendo más de un punto de cruce.

- **Cruce Uniforme:** para cada gen de la cadena del descendiente existe una probabilidad de que el gen pertenezca al padre y otra de que pertenezca a la madre



Fuente: http://sci2s.ugr.es/

Gráfico 6: Operador de cruce para representación binaria

Mutación, Nos indica la frecuencia con la que los genes de un cromosoma son mutados. Si no hay mutación, los descendientes son los mismos que había tras la reproducción. En caso de que haya mutaciones, parte del cromosoma descendiente es modificado y si la probabilidad de mutación es del 100%, la totalidad del cromosoma se cambia. En este caso, no se cambian simplemente unos bits del cromosoma, sino que se cambian todos, lo que significa que se produce una inversión en el cromosoma y no una mutación por lo que la población degenera muy rápidamente.

Entre los tipos de mutaciones podemos utilizar:

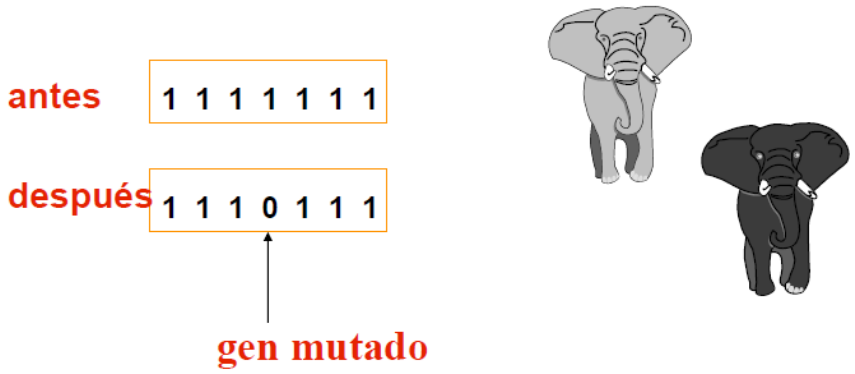
- **Mutación bit:** Existe una única probabilidad de que se produzca una mutación de algún bit. De producirse, el algoritmo toma aleatoriamente un bit, y lo invierte.

- **Mutación Multibit:** Cada bit tiene una probabilidad de mutarse o no, que es calculado en cada pasada del operador de mutación multibit.

- **Mutación de gen:** Igual que la mutación de bit, solamente que, en vez de cambiar un bit, cambia un gen completo.

- **Mutación Multigen:** Igual que la mutación de multibit, solamente que en vez de cambiar un conjunto de bits, cambia un conjunto de genes.

- **Mutación de intercambio:** existe una probabilidad de que se produzca una mutación. De producirse, toma dos bits/genes aleatoriamente y los intercambia.



Fuente: http://sci2s.ugr.es/

Gráfico 7**:** Mutación para representación discreta binaria

Los algoritmos genéticos están inspirados en la evolución natural, y son muy buenos en resolver problemas de optimización que para un humano sería muy difícil realizar dicha tarea y tendría que contar con el apoyo de un equipo de personas expertas.

```
BEGIN /* Algoritmo Genetico Simple */
    Generar una poblacion inicial.
    Computar la funcion de evaluacion de cada individuo.
    WHILE NOT Terminado DO
    BEGIN /* Producir nueva generacion */
        FOR Tamaño poblacion/2 DO
        BEGIN /*Ciclo Reproductivo */
            Seleccionar dos individuos de la anterior generacion,
            para el cruce (probabilidad de seleccion proporcional
            a la funcion de evaluacion del individuo).
            Cruzar con cierta probabilidad los dos
            individuos obteniendo dos descendientes.
            Mutar los dos descendientes con cierta probabilidad.
            Computar la funcion de evaluacion de los dos
            descendientes mutados.
            Insertar los dos descendientes mutados en la nueva generacion.
        END
        IF    la poblacion ha convergido THEN
            Terminado := TRUE
    END
END
```

Fuente: http://www.sc.ehu.es/ccwbayes/

Gráfico 8: Pseudocódigo del Algoritmo Genético Simple

6. Limitaciones

El poder de los Algoritmos genéticos proviene del hecho de que se trata de una técnica robusta, y puede tratar con éxito una gran variedad de problemas provenientes de diferentes áreas, incluyendo aquellos en los que otros métodos encuentran dificultades. Si bien no se garantiza que el algoritmo genético encuentre la solución óptima, del problema, existe evidencia empírica de que se encuentran soluciones de un nivel aceptable.

En el caso de que existan técnicas especializadas para resolver un determinado problema, lo más probable es que superen al algoritmo genético, tanto en rapidez como en eficacia.

El gran campo de aplicación de los algoritmos genéticos, se relaciona con aquellos problemas para los cuales no existen técnicas especializadas, incluso en el caso en que dichas técnicas existan, y funciones; bien pueden efectuarse mejoras de las mismas hibridándolas con los algoritmos genéticos.

7. Aplicación del Algoritmo Genético

La aplicación más común de los algoritmos genéticos ha sido la solución de problemas de optimización, en donde han mostrado ser muy eficientes y confiables. Sin embargo, no todos los problemas pueden ser apropiados para la técnica, por ello se recomienda en general tomar en cuenta las siguientes características del mismo, antes de intentar usarla:

- Su espacio de búsqueda (sus posibles soluciones) deben estar delimitadas dentro de un cierto rango.

- Debe poderse definir una función de aptitud que nos indique qué tan buena o mala es una cierta respuesta.

- Las soluciones deben codificarse de una forma que resulte relativamente fácil de implementar en la computadora.

El primer punto es muy importante, y lo más recomendable es intentar resolver problemas que tengan espacios de búsqueda discretos, aunque éstos sean muy grandes. Sin embargo, también podrá intentarse usar la técnica con espacios de búsqueda continuos, pero preferentemente cuando exista un rango de soluciones relativamente pequeño.

La función de aptitud no es más que la función objetiva de nuestro problema de optimización. El algoritmo genético únicamente maximiza, pero la minimización puede realizarse fácilmente utilizando el recíproco de la función maximizante (debe cuidarse, por supuesto, que el recíproco de la función no genere una división por cero). Una característica que debe tener esta función es que tiene ser capaz de "castigar" a las malas soluciones, y de "premiar" a las buenas, de forma que sean estas últimas las que se propaguen con mayor rapidez.

La codificación más común de las soluciones es a través de cadenas binarias, aunque se han utilizado también números reales y letras. El primero de estos esquemas ha gozado de mucha popularidad debido a que es el que propuso originalmente Holland, y además porque resulta muy sencillo de implementar.

8. Ventajas y desventajas de usar algoritmos genéticos

Ventajas

- No necesitan conocimientos específicos sobre el problema que intentan resolver.

- Poseen la habilidad para manipular muchos parámetros simultáneamente. Muchos problemas de la vida real no pueden definirse en términos de un único valor que hay que minimizar o maximizar, sino que deben expresarse en términos de múltiples objetivos.

- Cuando se usan para problemas de optimización maximizar una función objetivo- resulta menos afectado por los máximos locales (falsas soluciones) que las técnicas tradicionales.

- Resulta sumamente fácil ejecutarlos en las modernas arquitecturas masivamente paralelas.

- Usan operadores probabilísticos, en vez de los típicos operadores determinísticos de las otras técnicas.

Desventajas

- Al crear un AG es definir una representación del problema. El lenguaje utilizado para especificar soluciones candidatas debe ser robusto; es decir, debe ser capaz de tolerar cambios aleatorios que no produzcan constantemente errores fatales o resultados sin sentido.

- Pueden tardar mucho en converger, o no converger en absoluto, dependiendo en cierta medida de los parámetros que se utilicen tamaño de la población, número de generaciones, etc.-.

- Pueden converger prematuramente debido a una serie de problemas de diversa índole. esto a raíz de una incorrecta definición de la población inicial.

2.3. Objetivos de la investigación

Los objetivos de la presente tesis son los siguientes:

2.3.1. Objetivo general

Proporcionar una solución basada en algoritmo genético, que permita elaborar en forma automática el horario de exámenes, permitiendo cumplir con los requerimientos establecidos por la universidad San Ignacio de Loyola.

2.3.2. Objetivos específicos

1. Incrementar el uso eficiente de los salones de clase en la semana de exámenes parcial y final, utilizando los algoritmos genéticos.

2. Disminuir el tiempo de elaboración del horario de examen, utilizando Los algoritmos genéticos.

2.4. Formulación de hipótesis

Es necesario poder agilizar el proceso de elaboración del horario de exámenes utilizando una herramienta informática que cumpla con los objetivos propuestos, permitiendo poder minimizar errores y generar múltiples soluciones que estén cercanas al óptimo simultáneamente.

La determinación de la hipótesis de la investigación, comporta 3 elecciones:

1. **Poder resolver los problemas propuestos**: para ello se tiene como prioridad cubrir las restricciones fuertes que impiden poder contar con un horario aceptable, y afinarlo en forma adecuada con las restricciones débiles que permitirán contar con un horario optimizado.

2. **Optimizar el uso de salones**: crear un horario eficiente en el uso de salones que se distribuirá en los días de la semana que se realizara el examen.

3. **Tiempo óptimo de elaboración:** agilizar la elaboración del horario de examen para que en un tiempo prudencial de 16 horas poder contar con los horarios de exámenes.

2.4.1. Hipótesis general

La utilización del algoritmo genético permitirá poder cumplir con las restricciones establecidas en la elaboración del horario de exámenes de la universidad San Ignacio de Loyola.

2.4.2. Hipótesis especificas

1. El algoritmo genético optimizará el uso adecuado de salones referente a la capacidad y número de alumnos asignados.

2. El algoritmo genético obtendrá tiempo de respuestas aceptables para la elaboración del horario de exámenes de acuerdo a las restricciones.

CAPITULO III: DISEÑO METODOLÓGICO

3.1. Diseño de la investigación

El diseño del estudio es cuasi experimental debido a que el control de la variable no es absoluto y no existe un grupo de control.

3.2. Tipo

El tipo de investigación que se realiza corresponde a un estudio correlacional, donde se mide la variable Independiente (algoritmo genético) con la variable dependiente (elaboración del Horario de exámenes) y poder someter a prueba las hipótesis establecidas en esta tesis.

3.3. Enfoque

Es cuantitativo, donde se prueba las teorías descritas en esta tesis sobre el desarrollo de los algoritmos genéticos y la generación del horario de exámenes.

3.4. Población

La población se representa por los horarios de exámenes de todas las facultades de la universidad san Ignacio de Loyola, sede la Molina. En sus 26 carreras, que es un total de 52 horarios de exámenes entre parcial y final.

Muestra

Se trabaja con una muestra de 12 horarios de exámenes que están conformados para la siguiente carrera:

1. Carrera de Administración.
2. Carreras de Ingeniería civil.
3. Carrera de Psicología

Se utiliza la información de los periodos 2017-2 para la creación de los horarios de exámenes parciales y finales.

3.5. Operacionalización de variables

3.5.1. Variable independiente, algoritmo genético

- **Definición conceptual:** son algoritmo que utilizan los principios de la selección natural. Para ello combinan a los individuos más aptos y

compatibles entre las estructura de cadena de información aleatorizada intercambiada, para crear nuevos individuos con ciertas mejoras a comparación de sus antecesores. El objetivo del algoritmo es la supervivencia del más apto.

- **Definición Operacional:** El algoritmo genético crea una población al azar, selecciona dos cromosomas de la población para efectos de cruce, realiza el cambio de gen para poder crear una nueva descendencia y se evalúa el nuevo cromosoma.

  Las dimensiones que tendrá esta variable son:

  1. **Funcionamiento**: Dimensión donde encapsula los indicadores claves que muestren que el algoritmo genético este elaborando en forma correcta un horario de examen.

  2. **Termino:** Dimensión donde se puede comprobar que el horario generado cumple con todas las restricciones propuestas para la elaboración del examen.

1. Variable dependiente, elaboración del horario de examen optimizado

- **Definición conceptual:** Planificación y distribución de actividades humanas que permiten realizar una distribución de los recursos de la universidad (salones, profesores) en un espacio de tiempo que satisfagan los objetivos deseados.

- **Definición Operacional:** Facilita la elaboración del horario de exámenes a partir de las interrelaciones sistemáticas de sus distintas componentes que proporcionando múltiples soluciones de horarios.

  Las dimensiones que tendrá esta variable son:

1. **Planeación**: Dimensión que indica el tiempo y uso de aulas en la elaboración del horario de exámenes

2. **Restricciones** : Dimensión que indica el número de restricciones fuertes y débiles que debe cumplir el horarios de exámenes

3.6. Técnicas para la recolección de datos

La información es procesada internamente en la Universidad San Ignacio de Loyola, en el área de registros académicos a través de un sistema interno llamado sistema académico. De este sistema se recolectará la información necesaria como, las carreras, currículos, cursos y recintos para la elaboración del Horario de exámenes.

3.6. Técnica para el procesamiento y análisis de datos

La técnica estadística que se utiliza será la distribución T Studen, donde se comprará los indicadores en un antes y un después del desarrollo del algoritmo genético, los indicadores a ser comparados son:

1. Optimización del número de salones a utilizar.
2. Uso de salones para dos cursos de distintas carreras
3. Reducción de las horas de elaboración.
4. Establecer el mismo número de alumnos en los exámenes.

3.7. Aspectos éticos

En esta presente tesis, se pretende construir una investigación veraz y sincera, teniendo debidamente cuidado con la protección de información personal que se manipulará en el trascurso de las demostraciones y resultados que se realice para comprobar las hipótesis. Para ello se respetará la dignidad humana, la identidad, la diversidad, la confidencialidad y la privacidad de la información personal e institucional que no sea de consentimiento público, se contará con una manifestación de voluntad, informada, libre, inequívoca y específica, mediante la

cual las personas e instituciones estarán conscientes del uso de la información para los fines específicos de esta tesis.

El material que se publicará en esta tesis, se evitará en incurrir en faltas deontológicas que podría venir de las siguientes:

a) No se falsificará o inventará datos totales o parciales.
b) Plagiar lo publicado por otros autores de manera total o parcial.
c) Incluir como autor a quien no ha contribuido sustancialmente al diseño y realización del trabajo y publicar repetidamente los mismos hallazgos.

Los aportes que se utilicen de otros trabajos de investigación. Se respetará la propiedad intelectual institucional y demás normas de orden público referidas con los derechos de autores. Para ello se citarán cumpliendo las normas APA en la bibliográfica, según correspondan.

CAPÍTULO IV: RESULTADOS

En este capítulo se presentará la solución propuesta al problema de la generación de horarios de examen, basada en el diseño de un algoritmo genético, que puede ser utilizado en un sistema computacional de generación de horarios. Para ello se describirá lo siguiente: representación general del algoritmo, los datos necesarios para solucionar el problema, el diseño del cromosoma utilizado, las definiciones utilizadas en la función fitness, el pseudocódigo del algoritmo y los resultados de las pruebas.

4.1 Representación general del algoritmo genético aplicado a la solución de horario examen

Para la solución al problema de elaboración del horario de exámenes, para la tesis se optó con esta representación general del funcionamiento del algoritmo, sabiendo que en un salón de clase debe ser compartido por dos cursos distintos de diferentes carreras. Para ello se utiliza una matriz general que guarda la disponibilidad de los salones en cada generación que realiza el algoritmo por carrera, así la siguiente carrera no podrá utilizar los salones que están llenos. El gráfico 9 muestra el pseudocódigo del algoritmo genético propuesto.

```
//Constante que tiene el número de generaciones
Numgeneraciones
//variables
Tcarrera[], Trecintosxslots[] , SalondisponM[]

Funcion Algoritmo genético aplicado a la solución de horario examen

Tcarrera[] =  obtiene las  carreras a crear horarios  base de datos ()
Trecintosxslots[] =  Obtiene la  infraestructura de la base de datos()
llena SalondisponM[] = Trecintosxslots[]

  for (int c = 0  to  c < TotalCarreras_)

        Función población inicial( Tcarrera{c});

        for (int i = 1 to  i <= Numgeneraciones)
          Funcion Aptitud()
                //funciones de penalización
             sub funcion Validacion reserva de salones para el mejor curso y slot()
             sub funcion Curso de una carrera en slots distintos()
             sub funcion  Establecer el mismo número de alumnos por día()
          Funcion selección()
          Funcion Realiza_Cruce()
          Funcion Mutacion ()
       if ( Función condición de termino ()= true )
          Termina proceso de generacion()
       End if
 End for
       funcion actualiza matriz de salones disponible( SalondisponM[])
 end for
End funcion Algoritmo genético aplicado a la solución de horario examen
```

Fuente: Propia
Gráfico 9: Pseudocódigo función Algoritmo genético aplicado a la solución de horarios examen

4.2 Datos involucrados

Para realizar el diseño del algoritmo genético se tuvo que analizar cada restricción descrita en la tesis para la creación del horario de examen y establecer la estructura de la data que se necesita recolectar para poder crear el horario de exámenes, entre los cuales se especifica:

- **Tabla Carrera:** Contiene todas las carreras, código y nombre de la carrera.

Tabla 1
Tabla carrera

codcarrera	carrera
1	ADMINISTRACIÓN
2	ADMINISTRACIÓN EN TURISMO
3	ADMINISTRACIÓN HOTELERA
4	ECONOMÍA
5	INGENIERÍA AGROINDUSTRIAL Y AGRONEGOCIOS
6	INGENIERÍA INFORMÁTICA Y DE SISTEMAS
7	MARKETING
12	COMUNICACIONES
13	EDUCACIÓN
4219	INGENIERÍA INDUSTRIAL Y COMERCIAL
4220	DERECHO
4289	ARQUITECTURA, URBANISMO Y TERRITORIO

- **Tabla Curso:** Contiene todos los cursos de cada carrera que se dicta.

Tabla 2
Tabla curso

codcurso	curso
8	ETICA PROFESIONAL
68	PLANEAMIENTO ESTRATÉGICO
2734	DERECHO EMPRESARIAL
2744	DIRECCION DE MARKETING (SE DICTA EN INGLES)
2823	VENTA MINORISTA
7339	PLANEAMIENTO Y CONTROL DE LA PRODUCCIÓN II
16862	INFORMÁTICA APLICADA A LA GESTIÓN
29155	GERENCIA DE PROY. ARQUITECTÓNICOS URBANÍSTICOS
30296	PANADERÍA AVANZADA
91029	ENTREVISTA Y OBSERVACIÓN
91036	CONSTRUCCIÓN DE PRUEBAS PSICOLÓGICAS
91100	ENGLISH COMPOSITION II
91615	ADMINISTRACIÓN PARA LOS NEGOCIOS

- **Tabla Carrera x curso**: Contiene los cursos de cada carrera activa, el ciclo, el tipo de recinto que necesita el curso y el total de alumno matriculados que tiene el curso. Esta tabla fue construida para el algoritmo genético.

Tabla 3
Carrera x curso

codcarrera	codcurso	ciclo	codrecinto	cantalumno
1	91615	1	1	1071
1	95862	1	1	170
1	92130	3	1	1051
1	99034	3	1	155
1	92557	4	1	413
1	92268	5	1	710
1	92351	5	1	301
1	99039	5	1	207
1	92355	6	1	454
1	92917	6	1	73
1	92914	6	1	542

- **Tabla Tipo de recinto:** Contiene la descripción de los diferentes tipos de recientes que cuenta la universidad

Tabla 4
Tipo de recinto

codrecinto	recinto
1	Aula
2	Lab. Info.
3	Auditorio
4	Aula-Física
5	Aul-Au-Vs
6	Aula-Biol.
7	Lab. Diseño
8	Lab. Inglés
9	Coliseo
10	Cafeteria

- **Tabla Infraestructura:** Contienen todos los recientes de la universidad, el nombre del recinto, capacidad, la sede, el campo fila par, que es un campo construido para el algoritmo genético

Tabla 5
Tabla de infraestructura

codinfra	local	cod_tiprecinto	Infraestructura	capacidad	fila_Par
1	CUSIL 1	1	C1A101	42	21
2	CUSIL 1	1	C1A102	42	21
3	CUSIL 1	1	C1A103	42	21
4	CUSIL 1	1	C1A104	42	21
5	CUSIL 1	1	C1A105	42	21
6	CUSIL 1	1	C1A107	42	21
7	CUSIL 1	1	C1A109	42	21
8	CUSIL 1	1	C1A110	42	21
9	CUSIL 1	1	C1A111	42	21
10	CUSIL 1	1	C1A112	42	21
11	CUSIL 1	1	C1A114	42	21
12	CUSIL 1	1	C1A116	42	21
13	CUSIL 1	1	C1A201	42	21

- **Tabla días:** Contiene los días de la semana.

Tabla 6
Tabla de días

coddia	dias
1	Lunes
2	Martes
3	Miercoles
4	Jueves
5	Viernes
6	sabado
7	domingo

- **Tabla hora examen:** Contiene las horas de inicio en las cuales se puede dictar el examen.

Tabla 7
Tabla de hora examen

hora	codhora
8	1
11	2
14	3
17	4
20	5

- **Tabla Slots:** Contiene la combinación de los días y horas activas que puede iniciar un examen en la semana. Tabla construida para el algoritmo genético

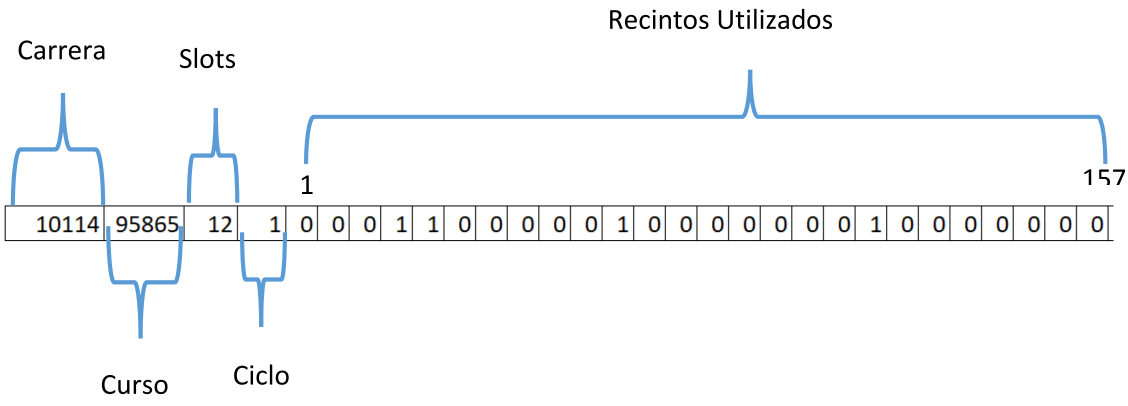
Tabla 8
Tabla de Slots

IdSlots	coddia	codhora
1	1	1
2	1	2
3	1	3
4	1	4
5	1	5
6	2	1
7	2	2
8	2	3
9	2	4
10	2	5
11	3	1
12	3	2

4.3 Representación del cromosoma

Para el diseño del cromosoma se toma en referencia el tipo de configuración que se requiere para resolver el problema de generación de horarios de exámenes, el cual se propone lo siguiente:

- Cada gen está codificado por carrera, curso, slots, ciclo y recintos utilizados, que está plasmado en un array, donde en la sección de los recintos, 1 indica si está utilizando el recinto y cero que no lo utiliza.
- Curso: Se distribuyen en un día y hora específica por distintos aulas o laboratorios que utilice el curso, por el total de alumnos de la carrera.
- Slost : Es la representación de un espacio de tiempo, ejemplo: si tenemos 6 días de la semana en la cual Lunes tiene el 1, Martes 2, Miércoles 3 así hasta sábado 6 y las horas de inicio de cada examen son : 8, 11,14,17,20 Diríamos que el slots 1, sería Lunes a las 8, slost 2 Lunes a las 11, slots 3 Lunes a las 14, así hasta el slots 30 que sería Sábado a las 20 horas.
- Recintos utilizados, aquí están los salones, laboratorios, salas de estudio o cualquier recinto que utilice el curso, se considera la mitad de la capacidad total de cada recinto, con el fin de que en un mismo salón de clase puedan estar distribuidos dos cursos de distintas carreras. Para el caso de los laboratorios se considera todo el espacio disponible.
- El número total de gen que contiene el cromosoma diseñado es de 161, donde los 4 primeros genes, representa el horario del examen para el curso y los 157 genes restantes en que salones estarán ubicados para ese horario.
- Cabe indicar que el cromosoma pude crecer en gen sin existiera más salones disponibles.



Fuente: Propia
Gráfico 10: Representación del cromosoma

4.4 Población inicial

Para la inicialización se tiene que crear soluciones al azar, sobretodo en el espacio de búsqueda que son los recintos a utilizar. El total de cromosomas que puede tener una carrera como población inicial está establecido por lo siguiente:

NC*CPI*Nslots

Donde el NC es la cantidad de curso que tiene la carrera, el CPI es la cantidad de población inicial que se le puede asignar a cada curso y el Nslots es el total de slot que puede tener el curso para iniciar el examen. Ahora si usamos la formular para una carrera que tiene 75 cursos, 14 combinaciones por cada curso y 30 slots, quedaría así: 75*14*30 =31500.

Se genera 31500 cromosomas para la carrera como la población inicial, En donde en forma aleatoria se le coloca 0 o 1 en cada bit del cromosoma.

10114	92130	2	3	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
10114	93851	2	8	0	1	0	0	0	0	1	1	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
10114	92257	3	6	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
10114	92973	3	8	0	0	0	1	1	0	1	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
10114	99114	3	9	0	0	0	1	1	0	0	1	0	0	1	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
10114	98212	4	6	0	0	0	1	1	0	0	1	0	0	1	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fuente: Propia
Gráfico 11: Representación de una población inicial

```
//Recibe la matriz cromosoma con el total de filas y columnas
Cromosoma [,]
//Recibe el tabla  de carrera x curso y slots
 Tcursoxcarrera ;  Tslots
//constantes
NumPoblacion ; Numslots; Totacurso; Totalsalon ;
//variables
Fila_Crom

Función población inicial
inicializa Fila_Crom=0

 for (int i = 0  to   TotalCurso)

      for (int j = 0 to Numslots)

           for (int p = 0 to  NumPoblacion )

                Cromosoma[Fila_Crom] =  carrera de  Tcursoxcarrera{i}  + curso de  Tcursoxcarrera{i} + slots de Tslots{i}  + ciclo de  Tcursoxcarrera{i}

                 for (int k = 4 to Totalsalon)
                       Cromosoma[Fila_Crom, k  ] =Random(0,1)
                 end for
                Fila_Crom = suma de la fila +1
           end for
     end for
 end for
end población inicial
```

Fuente: Propia
Gráfico 12: Pseudocódigo de la función de población inicial

4.5 Función fitness.

La función fitness determina si el algoritmo genético está distribuyendo en forma correcta los recintos según el curso y está utilizando en forma correcta la optimización de espacio.

Se consideran las siguientes variables, para la definición de la función fitness:

- U = Total de recintos
- I = Total de recintos incorrectos
- S = Total de Sillas por recinto
- $fSC(x)$ = Función que devuelve el total de sillas que debe utilizar el curso
- r_n = número de recintos utilizados

En donde: r_n
- 1 = si recinto está asignado al curso
- 0 = en caso contrario

- k_n = número de recintos incorrectos utilizados (incorrecto es cuando el recinto no corresponde a lo requerido por el curso)

- En donde: k_n
  - 1 = asignación de recinto incorrecto.
  - 0 = en caso contrario.

- y_n = número de sillas utilizados por recintos (cantidad de espacio que necesita utilizar el curso, se determina por el número de alumnos matriculados que son las sillas que utiliza)

En donde: y_n { 1 = asignación del número de sillas que tiene el recinto
0 = en caso contrario.

Definidas las variables, se indicara a continuación la función fitness:

$$f(x) = \left| \frac{fSC(x) - \sum_{n=1}^{S}(y_n)}{fsc(x)} \right| + \frac{\sum_{n=1}^{I}(k_n)}{\sum_{n=1}^{U}(r_n)}$$

En donde:

$f(x)$ { 0 = la asignación ha sido correcta, los recintos utilizados son los correctos que necesita el curso y el número de sillas utilizadas es lo requerido.
Diferente a cero, asignación no correcta.

```
//Recibe la matriz cromosoma con el total de filas y columnas
Cromosoma [,]
//constantes
Totalsalon;totalFilacromosoma_;Totalgen_; Totalsillasautilizar_
//variables
sumatotalrecintosutilizados_;sumatotalrecintoincorrectos_;sumatotalsillasutilizadas_, aptitud_

Funcion Aptitud
  for (int f = 2  to  totalFilacromosoma_)
          for (int c = 7  to  Totalgen_ )
              sumatotalrecintosutilizados_  = + recintos utilizados cromosoma[f,c]
              sumatotalrecintoincorrectos_ = + recintos incorrectos cromosoma[f,c]
             sumatotalsillasutilizadas_  =  + sillas utilizadas cromosoma[f,c]
        end for
     Totalsillasautilizar_ = Funcion Sillas a utilizar(cromosoma[f])
    aptitud_= ( valorabsoluto((Totalsillasautilizar_- sumatotalsillasutilizadas)/Totalsillasautilizar_) +(sumatotalrecintoincorrectos_ /sumatotalrecintosutilizados_
   cromosoma[f] =aptitud
  end for
end Aptitud
```

Fuente: Propia
Gráfico 13: Pseudocódigo de la función Aptitud

4.6 Penalización a las soluciones

Las penalizaciones se realizan con el fin de poder superar las restricciones que cuenta el horario de examen, aquellas restricciones que no son superadas por el horario de un curso se penalizan, sumando 1 al valor de aptitud. Para ello el algoritmo cuenta con las siguientes penalizaciones:

4.6.1 Penalización de reserva de recintos por curso y slot.

Esta penalización se realiza para poder reservar los salones al mejor cromosoma de una carrera, curso y slot, con ello se busca que el algoritmo tenga varias soluciones de horarios con distintos recintos utilizados. Para ello se compara cromosoma por cromosoma de un mismo curso y slot, se identifica al mejor cromosoma y este se queda con sus recientes distribuidos, a los demás se penaliza sumando 1 en su aptitud en caso estén utilizando los mismo recinto que el cromosoma con mejor aptitud.

```
//Recibe la matriz Cromosoma
Cromosoma [,]
//Matriz que almacena el mejor horario del curso x carrera
Musosalon [,]
//constantes
totalFilacromosoma_;mejorcromosoma_
Funcion Validacion reserva de salones para el mejor curso y slot
mejorcromosoma_=0
        for (int f = 1  to  totalFilacromosoma_)
            Cromosoma1  = obtengo el  cromosoma de un curso(F)
               if (cromosoma1 < mejorcromosoma )
                     mejorcromosoma = cromosoma1
                  if (cromosoma1 = esta en la matriz Musosalon[f] )
                       mejorcromosoma =  aptitud de mejorcromosoma  + 1
                  end if
              else
                     cromosoma1 =  aptitud de cromosoma1 + 1
            end if
          end for
end Validacion reserva de salones para el mejor curso y slot
```

Fuente: Propia
Gráfico 14: Pseudocódigo de la función reserva de aula

4.6.2 Penalización por curso de una carrera en slots distintos

Esta penalización se realiza con el fin de poder evitar cruce de horarios para una misma carrera, así cada curso por ciclo de una carrera estará en horarios distintos. La validación de los cruces de exámenes se realiza cuando el algoritmo debe penalizar a los cromosomas de una carrera, ciclo y cursos que compartan un mismo slot, para ello el cromosoma con la mejor aptitud se queda con el slot que define el día y hora del examen y los demás cromosomas son penalizados sumando 1 a su aptitud.

```
//Recibe la matriz Cromosoma
Cromosoma [,]
//Recibe la matriz que almacena el mejor horario del curso x carrera
Musosalon [,]
//Recibe la fila del mejor cromosoma de una carrera, curso y slos
FilaMejor_
//constantes que contiene el numero de cromosomas
totalFilacromosoma_
//variables
MFila_

Funcion Curso de una carrera en slots distintos
        MFila_ = 0;
    while (MFila_ < totalFilacromosoma_)

        if (slots del curso MUsosalon[MFila] = slots curso del Cromosoma[FilaMejor_] and ciclo del curso MUsosalon[MFila_] = ciclo del curso Cromosoma[FilaMejor_] )
          //se penaliza al curso porque cada ciclo, curso de una carrera solo deben utilizar un slots
          Cromosoma[FilaMejor_] = aptitud de Cromosoma[FilaMejor_] + 1
        end if
         Mfila_=Mfila +1
  end while
End Curso de una carrera en slots distintos
```

Fuente: Propia
Gráfico 15: Pseudocódigo curso de una carrera en slots distintos

4.6.3 Penalización para establecer el balance del mismo número de alumnos por día

Esta penalización permite poder balancear el número de alumnos por día, Para ello contamos con una constante que tiene el número de alumno por día de una carrera, que se obtiene el número de alumnos matriculados en una carrera “NC” y dividiéndolo por el número de días “ND” en las cuales se dictara el examen:

$$f(x) = \frac{\mathrm{NC}}{ND}$$

Se penaliza al cromosoma sumando más 1 en su aptitud, cuando ya no hay disponibilidad en el día.

```
//Recibe la matriz Cromosoma
Cromosoma [,]
//Recibe la matriz que tiene el numero de alumno por dia
  AluXdiaM[,]
//Recibe la fila del mejor cromosoma de una carrera, curso y slos
FilaMejor_
//variables
Ndia_
Funcion  Establecer el mismo número de alumnos por día
            Ndia_ = Obtengo el número de dia Cromosoma[FilaMejor_]
            if ( espacio utilizado  del Día AluXdiaM[Ndia_] <  espacio requerdio para el Día  AluXdiaM[Ndia_] )
              AluXdiaM[ Ndia_] = Suma el espacio utilizado por  Cromosoma[FilaMejor_]
            else
              Cromosoma[FilaMejor_] = aptitud del Cromosoma[FilaMejor_] + 1;
          End if
End Establecer el mismo número de alumnos por día
```

Fuente: Propia
Gráfico 16: Pseudocódigo establecer el mismo número de alumnos por día

4.7 Selección

La función de selección es mediante la utilizando del torneo entre dos, se realiza una comparación de aptitud de dos cromosomas de un mismo curso, para ello se selecciona por parejas y en forma consecutiva se compara el valor de aptitud de los dos, el mejor cromosoma se queda y el peor se elimina.

```
//Recibe la matriz Cromosoma
Cromosoma [,]
//constantes  que contiene el numero de cromosomas
totalFilacromosoma_
Funcion selección


    for (int f = 1  to f < totalFilacromosoma_)

        if (aptitud de cromosoma[F] > aptitud de cromosoma [f+1])
            elimino Cromosoma[f]="E"
          else
            elimino Cromosoma[f+1]="E"
        end if
   End for
End funcion selección
```

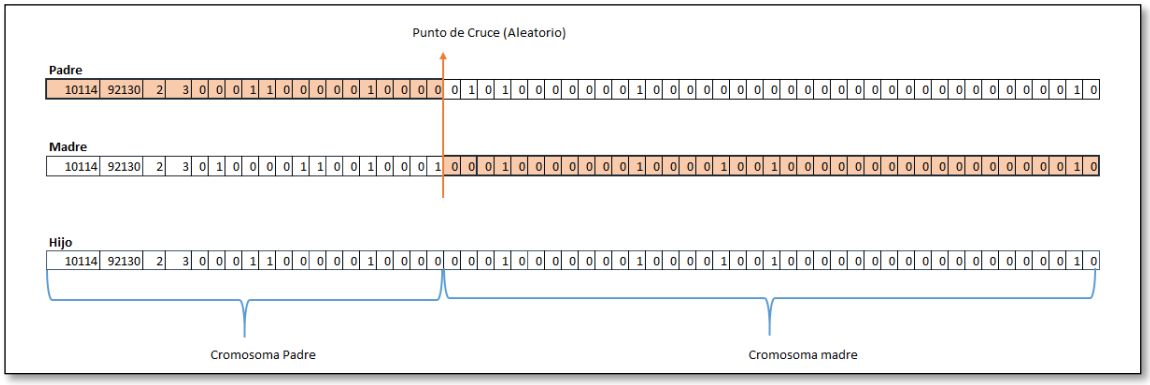
Fuente: Propia
Gráfico 17: Pseudocódigo función selección

4.8 Operadores

4.8.1 Cruce

En forma aleatoria se asigna las parejas de la población de un mismo curso, para mantener el mismo número de la población con el mismo tamaño fijo, cada cruce realizado solo tendrá un solo hijo. El tipo de cruce utilizando para el algoritmo es el cruce básico, donde se selecciona un punto al azar de la

cadena, la parte anterior del punto es copiado al genoma del padre y la posterior de la madre.



Fuente: Propia
Gráfico 18: Realiza cruce de Cromosomas

```
//Recibe la matriz Cromosoma
Cromosoma [,]
//constantes  que contiene el numero de cromosomas
totalFilacromosoma_ , totalBitcromosoma_
//variables
ParejaP_ ; ParejaM_ ; newcromosoma_

Funcion Realiza_Cruce

 for (int f = 2  to   f < totalFilacromosoma_ )

      ParejaP_ = Obtengo cromosoma Padre random(totalcromosomas )
      ParejaM_ = Obtengo cromosoma Madre Random(totalcromosomas)
       puntodecruce_ = random(totalBitcromosoma_)

        newHijo_ = cruce de los cromosomas ParejaP y ParejaM(puntodecruce)

       Cromosoma[]= newhijo
 end for
end Realiza_cruce
```

Fuente: Propia
Gráfico 19: Pseudocódigo función Realiza cruce

4.8.2 Mutación

La mutación que se realiza es Multibit, donde cada bit tiene la probabilidad de mutarse o no, la posición del bit que se muta es en forma aleatoria. Para el caso del problema, solo se mutará a los nuevos cromosomas generados del cruce siguiendo la teoría de la evaluación. La cantidad de cromosomas a

mutar y el número de bit que se debe mutar se obtienen de dos fórmulas que son la siguiente:

Se consideran las siguientes variables:

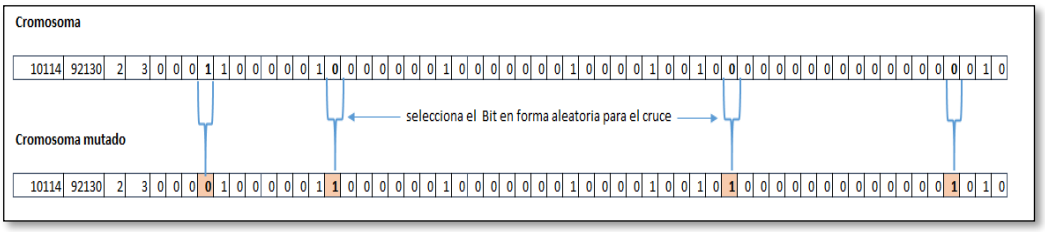
P = Cantidad de nuevos cromosomas generados.
C = Cantidad de bit que tiene un cromosoma
PMP = Valor en porcentaje de mutación por población.
PMB = Valor en porcentaje de mutación por bit.

1) CCM= P*PMP

2) CBM= C*PMB

Donde:

1) CCM=Devuelve la cantidad de cromosomas que serán mutados.
2) CBM=Devuelve la cantidad de Bit que serán mutado en un cromosoma.



Fuente: Propia
Gráfico 20: Mutación Multibit de Cromosoma

```
//Recibe la matriz Cromosoma
Cromosoma [,]
//constantes  que contiene el numero de cromosomas
totalFilacromosoma_, totalBitcromosoma_ ,  MutacionPoblacion_ , Mutaciongen, Totalgeneracionhijos
//variables
CCM_;   CBM_;Cromosomamutar_;PosicionBitamutar_;fcolum_;Tfila

Funcion Mutacion

   CCM_ = Totalgeneracionhijos_ *  PMP
   CBM_ =  totalBitcromosoma_ * PMB

 while (Tfila_ <=  CCM_)
         Cromosomamutar_ =Random (Totalgeneracionhijos_);

     while (fcolum_ <= CBM_)
           PosicionBitamutar_ = Random(totalBitcromosoma_ );
           Muta  Cromosomamutar_(PosicionBitamutar_)
          fcolum_=fcolum_+1
      End while
    Tfila_ = Tfila_ +1
 End while
End Mutacion
```

Fuente: Propia
Gráfico 21: Pseudocódigo función Mutación

4.8.3 Condiciones de término

La condición de término del proceso se determina cuando el algoritmo encuentra el horario a todos los cursos antes de llegar al número de generaciones máxima, esto se realiza a través de una comprobación a los mejores cromosomas del curso de la carrera que no tengan ningún salón incorrecto asignado y se deposita en un matriz de resultado final. Esta comprobación se realiza por cada generación hasta que se cumpla la condición.

```
//Recibe la matriz Cromosoma
Cromosoma [,]
//constantes  que contiene el numero de cromosomas
totalFilacromosoma_
//matriz donde se guarda el resultado final
CromosomaM[,]
//Variable
Count_curso_;

Función condición de termino
 for (int f = 0 to f < totalFilacromosoma_)
   if  ( Valor de aptitud  Cromosoma[f] < 1  and  error de salon incorrectos Cromosoma[f] =0   )
         CromosomaM[]= Copio el Cromosoma[F];
         Count_curso_ =Count_curso_+1
    end if
  end for

   if (TotalCurso_ = Count_curso_)
         Termina proceso de generacion ;
  end if

End condición de termino
```

Fuente: Propia
Gráfico 22: Pseudocódigo función condición de término

4.9 Evaluación de desempeño del algoritmo

El desempeño del algoritmo genético se centra en poder calibrar en forma correcta la solución planteada y pueda brindar buenos resultados.

4.9.3 Calibración de los parámetros de entrada del algoritmo genético

La prueba de calibración consiste en poder encontrar los parámetros adecuados como el número de la población a generar, el porcentaje de cromosomas a mutar, el porcentaje de bit que serán mutados. Permitiendo que el algoritmo pueda encontrar el recinto adecuado para toda la carrera con una calibración en común, antes que termine el número máximo de interacciones propuestas.

Para la calibración se utiliza la metodología de prueba y error.

Para este caso se escoge la carrera de Psicología, Administración e Ing. Civil donde todos los cursos Psicología y Administración necesitan solo un tipo de recinto que es Aula, en cambio Ing. Civil necesita Aula y laboratorio informático, teniendo como Objetivo que la solución pueda converger antes de la hora y media.

A continuación, las características de cada carrera:

Psicología

1. Total de curso: 44
2. Tipo de infraestructura a utilizar: Aula
3. Número total de sillas solicitadas por la carrera: 1740

Administración

1. Total de curso: 37
2. Tipo de infraestructura a utilizar: Aula
3. Número total de sillas solicitadas por la carrera: 8419

Ing. Civil

1. Total de curso: 44
2. Tipo de infraestructura a utilizar: Aula y Laboratorio de informática
3. Número total de sillas solicitadas por la carrera: 2978

Para las pruebas contamos con los siguientes recintos, sabiendo que el algoritmo genético solo tomara la mitad de cada recinto para la creación del horario de exámenes.

Tabla 9
Tabla recintos utilizada

Recinto	Capacidad	Espacio Utilizado por el algoritmo	Cantidad
Aula	20	10	1
Aula	24	12	1
Aula	42	21	63
Aula	48	24	61
Lab. Informatica	25	25	7

En la tabla 10, se tiene el número de pruebas (N°), la carrera, el CPI “cantidad de población inicial de un curso” que se explica con detalle en la parte de población inicial, Total de población generada para las pruebas, número de generación que tendrá el algoritmo, el PMP “valor en porcentaje de mutación por población”, CCM “cantidad de cromosomas que serán mutados”, PMB ”Cantidad de Bit que serán mutado en un cromosoma”, valor máximo del fitness “valor promedio del fitness al iniciar el algoritmo”, valor mínimo del fitness “valor promedio de fitness al terminar de converger el algoritmo”, número máximo de generaciones “valor que indica en que generación convergió el algoritmo”, Tiempo “calcula la demora de la prueba”, necesita más tiempo para converger “indicador que establece que el algoritmo necesita un número mayor de generación para converger”, Horario completo “Indica que el horario de examen está completo para los parámetros de las pruebas.

Tabla 10

Resultados de las pruebas de calibración según el número de generaciones por carrera

	Parámetros de calibración								Resultados					
N°	Carrrera	CPI	Total de poblacion	Número de Generaciones	PMP	CCM	PMB	CBM	Valor maximo Fitness f(x)	Valor Minimo Fitness f(x)	Número Máximo Generaciones	Tiempo	Necesita más tiempo para converger	Horario completo
1	Psicol.	14	16,200	3000	65%	5265	3%	5	62.1413	1.3954	3000	1 hora, 8 minutos, 59 segundo	Sí	No
2	Psicol.	16	21,120	3000	50%	5280	3%	5	62.2491	1.0853	3000	1 hora, 55 minutos, 3 segundo	Sí	No
3	Psicol.	16	21,120	3000	65%	6864	3%	5	62.2417	0.0848	2543	1 hora, 38 minutos, 14 segundo	No	Sí
4	Admin.	16	12,480	3500	65%	4056	3%	5	122.8891	0.1925	3235	47 minutos , 16 segundo	No	Sí
5	Admin.	14	10,920	3000	65%	3549	3%	5	122.7728	0.1822	1372	20 minutos , 2 segundo	No	Sí
6	Ing. Civil	14	18,480	3500	40%	3696	3%	5	59.7587	0.9042	3500	1 hora, 50 minutos, 46 segundo	Sí	No
7	Ing. Civil	16	21,120	3500	40%	5280	3%	5	59.8157	0.8900	3500	2 horas, 28 minutos, 20 segundos	Sí	No
8	Ing. Civil	16	21,120	3000	65%	6864	3%	5	59.7897	1.3750	3000	1 hora, 56 minutos, 36 segundo	Sí	No

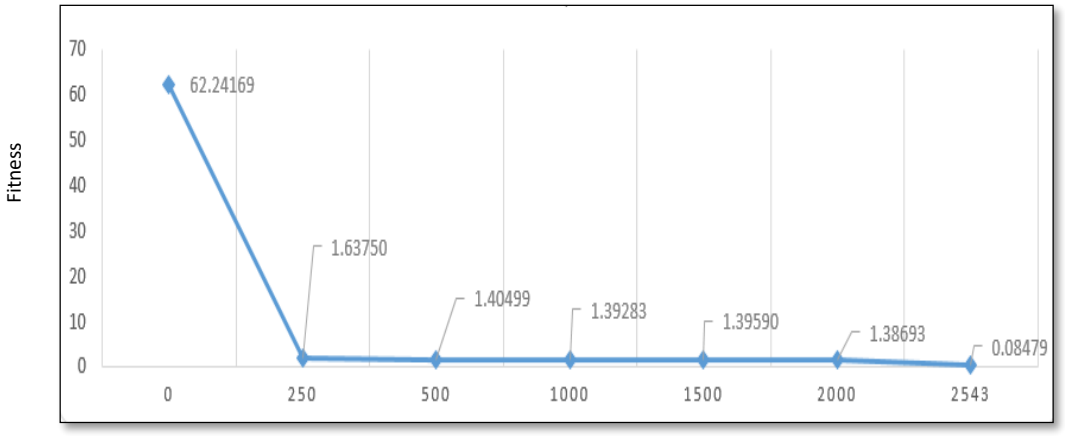
Los resultados de la tabla se muestra que el algoritmo genético converge en todas las pruebas realizadas, pero el horario generado no necesariamente es lo esperado porque converge en el número máximo de generaciones y no termina de encontrar horarios a todos los cursos.

Solo para las pruebas 3,4 y 5 el algoritmo entrego el horario antes de converger, quiere decir que encontró horarios para todos los cursos, tomando para ello los mismos parámetros de calibración PMP=65% y PMB=3%, y difiriendo en el CPI para la prueba 5 que fue 14 y origino un resultado muy favorable porque entrego el horario en la mitad del tiempo que en la prueba 4.

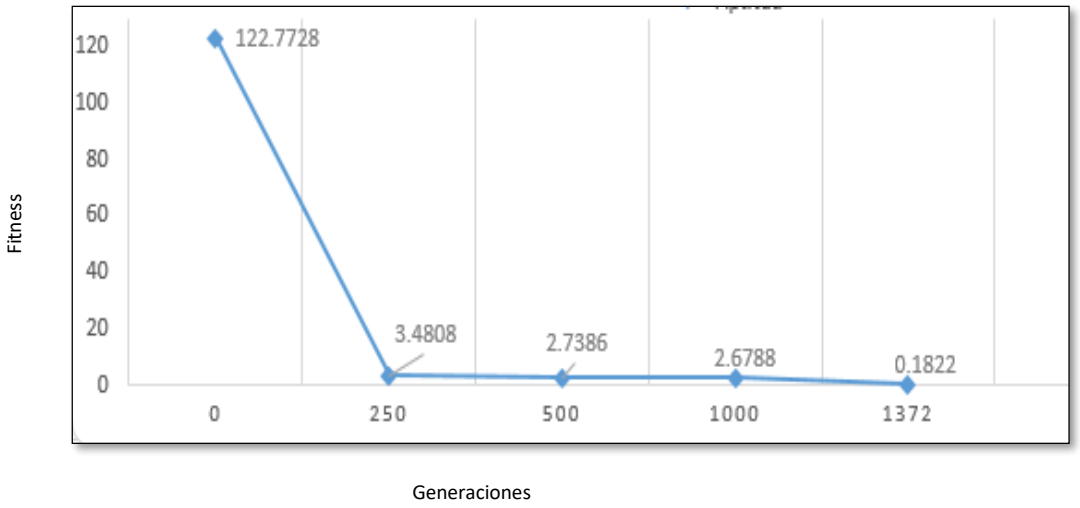
Para las pruebas de la carrera de Ing. Civil, el algoritmo no pudo encontrar el horario en todos los cursos en el horario establecido

A continuación se revisara el gráfico del valor de la función fitness, tomando en consideración los mejores resultados o el tiempo más corto en converger.

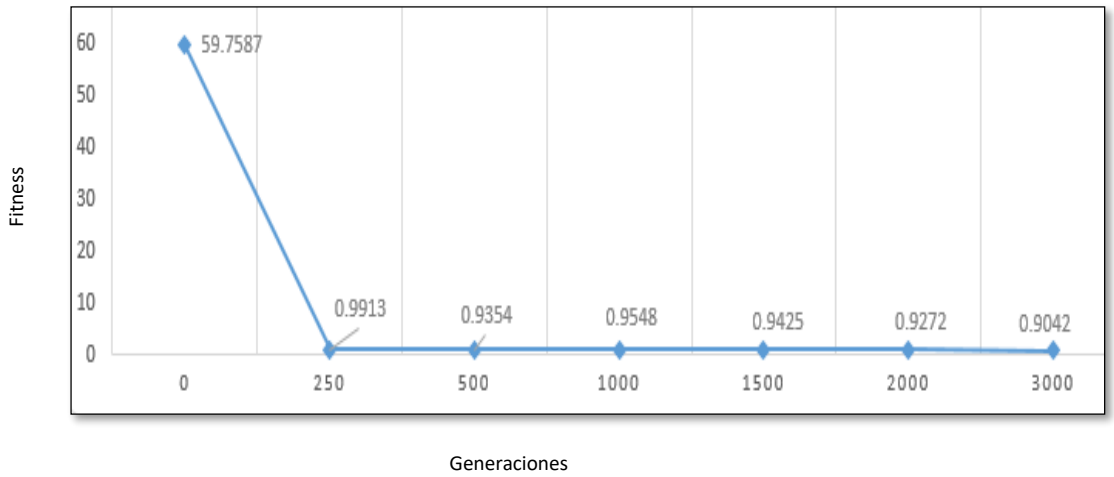
Prueba 3, Carrera de Psicología, tiempo: 1 hora, 38 minutos, 14



Prueba 5, Carrera de Administración, tiempo: 20 minutos, 2 segundos



Prueba 6, Carrera de Ing. Civil, tiempo: 1 hora, 50 minutos, 46 segundos



Fuente: Propia
Gráfico 23: Valor de la función fitnees por generaciones

Revisando las estadísticas de convergencia, podemos ver que en la generación 250 para toda la carrera, el algoritmo tiene un valor fitness cerca a cero y sigue así por varias generaciones más hasta que converge o llegue a su máxima generación.

Analizando el resultado final de las pruebas en las cuales el algoritmo genético no encontró horario por llegar al número máximo de generación, los cursos que se quedaron sin horario fueron aquellos en donde las sillas que solicitan son menores a la capacidad de la mayoría de salones. Así tenemos la siguiente lista:

1. La carrera Psicología cuenta con 8 cursos, donde las sillas que solicitan son en muchos de los casos menores a la capacidad de la mayoría de salones.

Tabla 11
Tabla tipo de recinto con sillas solicitadas, Psicología

ciclo	Tipo recinto	Sillas solicitadas
4	Aula	8
4	Aula	10
4	Aula	12
8	Aula	14
10	Aula	17
10	Aula	17
4	Aula	18
9	Aula	18

4. La carrera Administración cuenta con 15 cursos, donde las sillas que solicitan son en muchos de los casos menores a la capacidad de la mayoría de salones.

Tabla 12
Tabla tipo de recinto con sillas solicitadas, Administración

ciclo	Tipo recinto	Sillas solicitadas
3	Aula	8
7	Aula	8
8	Aula	8
8	Aula	8
10	Aula	8
10	Aula	8
6	Aula	9
3	Aula	10
7	Aula	11
8	Aula	12
7	Aula	14
2	Aula	17
10	Aula	18
4	Aula	19
9	Aula	20

4. La carrera Ing. Civil, cuenta con 10 cursos, donde las sillas que solicitan son menores a la capacidad de un salón.

Tabla 13
Tabla tipo de recinto con sillas solicitadas, Ing. civil

ciclo	Tipo recinto	Sillas solicitadas
9	Aula	8
9	Aula	8
9	Aula	8
6	Aula	8
1	Aula	11
9	Aula	11
9	Aula	13
9	Aula	17
9	Aula	19
9	Aula	20

Se puede concluir que el algoritmo tarda en encontrar un horario para estos cursos teniendo en cuenta las restricciones que existe para la creación del horario. Notamos que en la creación de la población inicial los cursos que se depositan en los cromosomas no llevan ningún orden, por lo cual los primeros cursos que estén en la población inicial tendrán más oportunidad de tener un horario a comparación de los últimos.

Para solucionar este problema vamos a realizar ciertas mejoras en el algoritmo:

1. Indicar al algoritmo genético que debería dar prioridad a los cursos que solicitan número de sillas menores a la capacidad de un salón. Esto se plasma en el algoritmo como un ordenamiento por número de sillas y cursos en forma ascendente.

```
//Recibe la matriz cromosoma con el total de filas y columnas
Cromosoma [,]
//Recibe el tabla  de carrera x curso y slots
 Tcursoxcarrera ;  Tslots
//constantes
NumPoblacion ; Numslots; Totacurso; Totalsalon ;
//variables
Fila_Crom

Función población inicial
inicializa Fila_Crom=0
          Tcursoxcarrera  = ordenar por los campos de Número de sillas solicitadas y curso ascendente  // Cambio de mejora
 for (int i = 0  to   TotalCurso)

      for (int j = 0 to Numslots)

           for (int p = 0 to  NumPoblacion )

                Cromosoma[Fila_Crom] =  carrera de  Tcursoxcarrera{i}  + curso de  Tcursoxcarrera{i} + slots de Tslots{i}  + ciclo de  Tcursoxcarrera{i}

                 for (int k = 4 to Totalsalon)
                        Cromosoma[Fila_Crom, k  ] =Random(0,1)
                 end for
                Fila_Crom = suma de la fila +1
           end for
     end for
 end for
end población inicial
```

Fuente: Propia
Gráfico 24: Pseudocódigo de la función de población inicial, con ordenamiento por número de sillas por curso

Una vez realizado la mejora, el algoritmo fue probado con diversos valores de calibración, los resultados los podremos apreciar en la tabla 17.

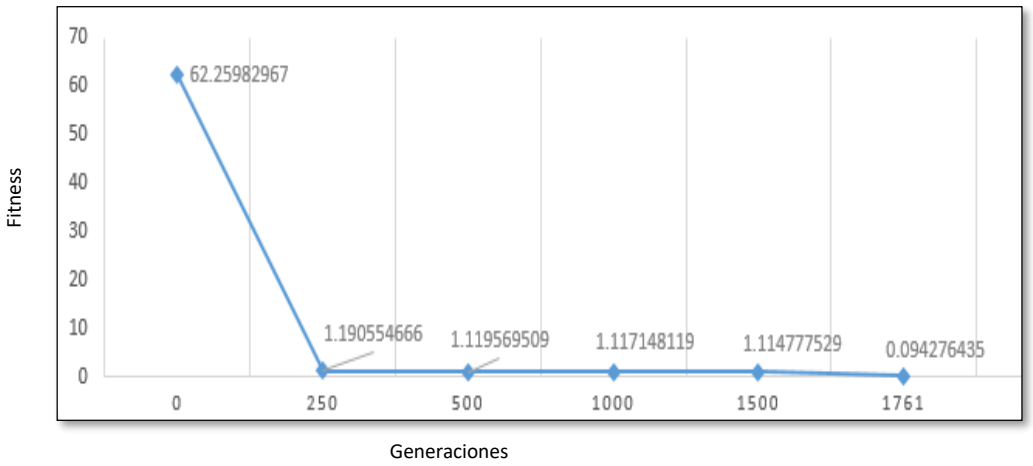
Tabla 14

Resultados de la segunda prueba de calibración, usando la mejora de ordenamiento

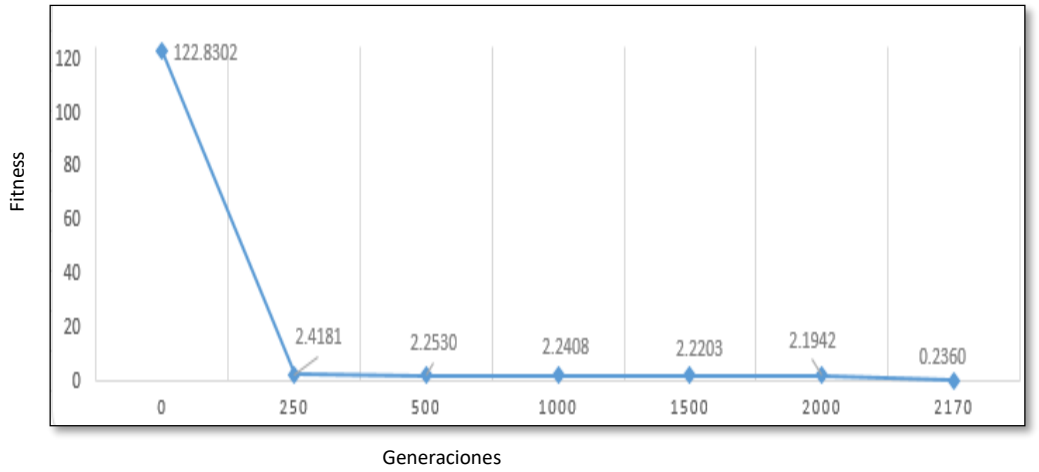
	Parámetros de calibración								Resultados					
N°	Carrrera	CPI	Total de poblacion	Número de Generaciones	PMP	CCM	PMB	CBM	Valor maximo Fitness f(x)	Valor Minimo Fitness f(x)	Generacion en converger	Tiempo	Necesita más tiempo para converger	Horario completo
1	Psicol.	16	21,120	3000	65%	6864	3%	5	62.3322	0.0768	2354	1 hora, 41 minutos,57 segundos	No	Sí
2	Psicol.	16	21,120	3000	50%	5280	3%	5	62.2598	0.0943	1761	1 hora, 10 minutos, 14 segundos	No	Sí
3	Admin.	16	12,480	3000	65%	4056	3%	5	122.7137	0.2132	1885	28 minutos, 28 Segundo	No	Sí
4	Admin.	16	12,480	3000	50%	3120	3%	5	122.8302	0.2360	2170	30 minutos, 16 segundos	No	Sí
5	ing.Civil	16	21,120	3000	65%	6864	3%	5	59.8125	0.1134	2009	1 hora, 32 minutos, 39 segundos	No	Sí
6	ing.Civil	16	21120	3000	50%	5280	3%	5	59.8221	0.1219	2386	1 hora, 33 minutos, 47 segundos	No	Sí

Se puede observar que el mejor resultado fue la configuración PMP=50% Y PMB=3%, que se puede utilizar como una configuración estándar para la generación de horarios de exámenes, el tiempo en que convergen los horarios se encuentran en el rango de tiempo que se determinó como objetivo para estas pruebas de hora y media.

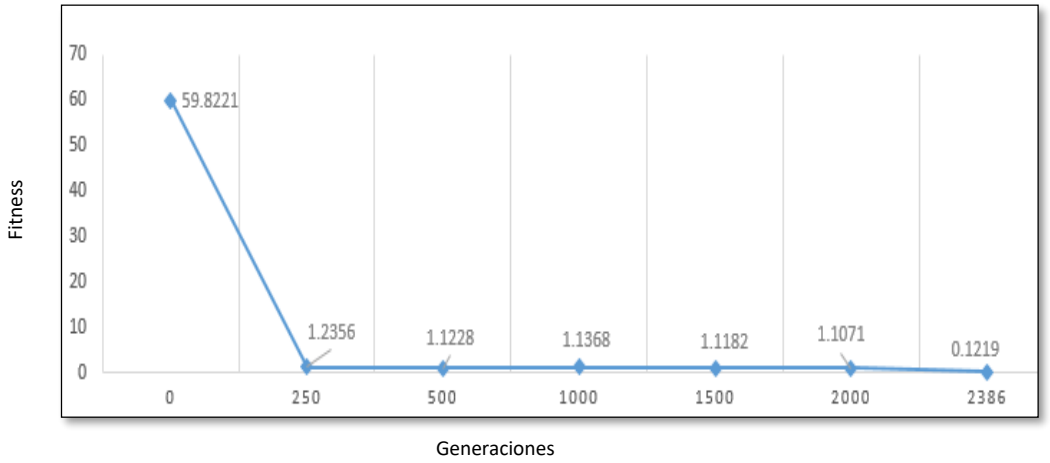
Prueba 2, Carrera de Psicología, tiempo: 1 hora, 10 minutos, 14



Prueba 5, Carrera de Administración, tiempo: 30 minutos, 16 segundos



Prueba 6, Carrera de Ing. Civil, tiempo: 1 hora, 33 minutos, 47 segundos



Fuente: Propia
Gráfico 25: Valor de la función fitnees por generaciones, después de la mejora

Analizando el gráfico 25 del valor fitness por generación, podemos notar que el algoritmo le toma menos generación poder encontrar los horarios, esto debido a la mejora de ordenamiento que brinda mayor oportunidad a los cursos con solicitudes de sillas muy bajas.

CAPÍTULO V: DISCUSIÓN, CONCLUSIONES Y RECOMENDACIONES

5.1 Discusión

En este capítulo se discuten los resultados obtenidos de las pruebas realizadas y se contrasta con las hipótesis realizadas en la tesis que nos permitirá determinar si las hipótesis son afirmativas.

Contrastación de la Hipótesis general con los resultados

La primera hipótesis dice el siguiente:

"La utilización del algoritmo genético permitirá poder cumplir con las restricciones establecidos en la elaboración del Horario de exámenes de la universidad san Ignacio de Loyola."

Para comprobar esta hipótesis se planteó lo siguiente:

1. Definir las restricciones del problema del horario de los exámenes.
2. Definir la forma que estas restricciones se consideren dentro del algoritmo.

A continuación, las restricciones del horario de exámenes se muestran en la tabla siguiente:

Tabla 15

Tabla que contrasta las restricciones del horario con el diseño del algoritmo genético

N	Restricciones	Tipo de restricción	comprobación
1	Programación de un segundo horario de examen para un mismo curso	Fuerte	el diseño del cromosoma garantiza que el horario de una carrera y curso, encuentre en un mismo día y hora el número de sillas solicitada.
2	El uso de salones para dos cursos distintos en el mismo horario	Fuerte	El diseño del cromosoma permite incluir en un mismo salón a dos cursos de distintas carreras, en cada ejecución solo se contempla la mitad de la disponibilidad del salón.
3	Cruces de exámenes	Fuerte	Las penalizaciones incluidas en la función Fitness, evitan el cruce de horarios de los cursos de una carrera por ciclo.
4	Recintos adecuados según el curso	Fuerte	La función fitness garantiza que cada curso cuente con los recintos adecuados para el examen.
5	Establecer el mismo número de alumnos para todos los días de examen	Débiles	Las penalizaciones incluidas en la función Fitness permiten una distribución adecuada de sillas a utilizar durante la semana.

Se puede determinar que la hipótesis general es aceptada debido que el algoritmo cumple con todas las restricciones propuesta y encuentra horarios para más de una carrera a la vez.

Hipótesis secundaria

Primera hipótesis secundaria

"El algoritmo genético optimizará el uso adecuado de salones referente a la capacidad y número de alumnos asignados."

Tabla 16

Tabla de distribución de uso de salones

N°	Carrera	Total de curso	solicitud de sillas	salones utilizados	Valor fitness f(x)
1	Administración	37	8,419	389	0.13847
2	Ingeniería Civil	44	2,978	147	0.13100
3	Psicología	44	1,740	90	0.14048
Total general		125	13137	626	0.13665

La hipótesis secundaria es aceptada puesto el diseño del algoritmo permite utilizar los recintos adecuados, según la tabla 16, donde se realizó 3 horarios de exámenes para las carreras de Administración, Ingeniería Civil y Psicología , se puede ver que la función fitness está cerca a cero eso garantiza que el algoritmo genético está distribuyendo en forma correcta los recintos según el curso y está utilizando en forma correcta la optimización de espacio teniendo en cuenta las restricciones que cuenta los horarios.

Segunda hipótesis secundaria

"El algoritmo genético obtendrá tiempo de respuestas aceptables para la elaboración del horario de exámenes de acuerdo a las restricciones"

Tabla 17

Tiempo de respuesta para la elaboración del horario de examen

N°	Carrera	Total de curso	solicitud de sillas	Generación en converger	Tiempo	Necesita más tiempo para converger	Horario completo
1	Administración	37	8,419	2170	0:30:16	No	Sí
2	Ingeniería Civil	44	2,978	1761	1:33:47	No	Sí
3	Psicología	44	1,740	1761	1:10:14	No	Sí

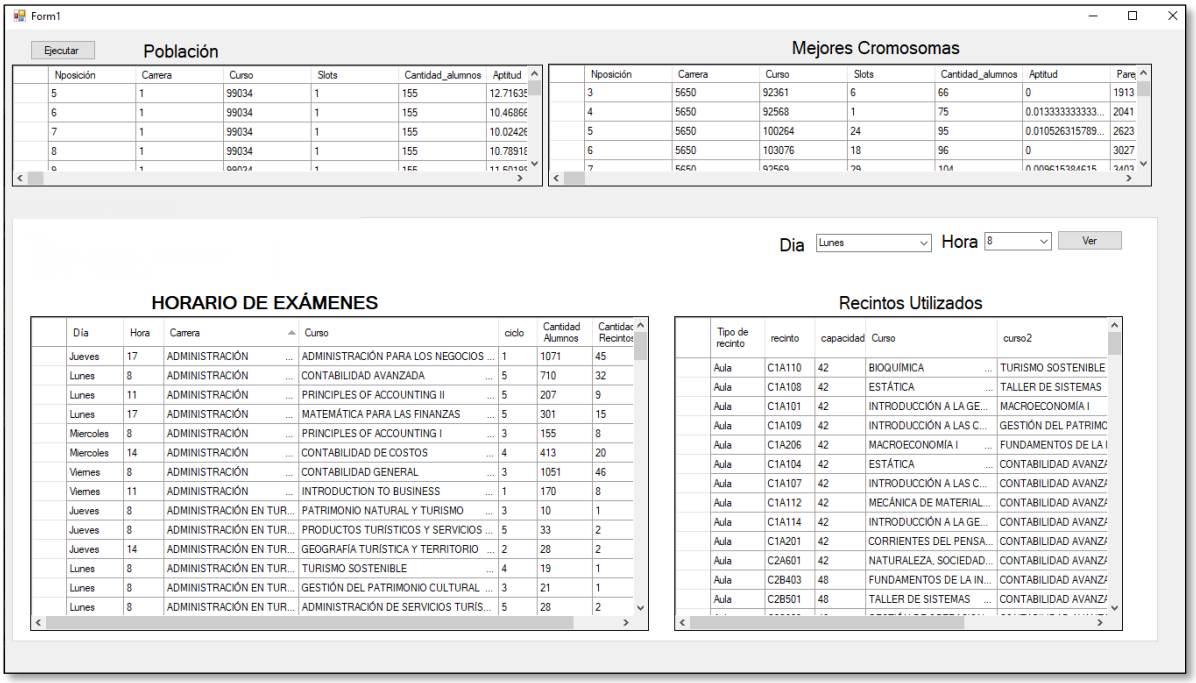
La hipótesis secundaria es aceptada, puesto el diseño del cromosoma propuesto permite optimizar el tiempo de respuesta.

5.2 Conclusiones

Objetivo general

"Proporcionar una solución basado en algoritmo genético, que permita elaborar en forma automática el horario de exámenes, permitiendo cumplir con los requerimientos establecidos por la universidad san Ignacio de Loyola"

El algoritmo genético diseñado y desarrollado cumple con dicho objetivo, presentando una solución al problema de encontrar el horario de exámenes como se muestra en la grafico siguiente, donde se elaboró el horario de exámenes para las carreras de Administración, Ingeniería civil y Psicología:



Fuente: Propia
Gráfico 26: Horario de examen, generado con el algoritmo genético propuesto.

A continuación se menciona los objetivos específicos:

- Incrementar el uso eficiente de los salones de clase en la semana de exámenes parcial y final, utilizando los algoritmos genéticos.

Tabla 18

Tiempo promedio y salones utilizados en la elaboración de horarios de exámenes en forma manual

N°	Carrera	Total de curso	solicitud de sillas	Tiempo promedio de elaboracion del horario	salones utilizados
1	Administración	37	8,419	1:30:00	406
2	Ingeniería Civil	44	2,978	1:30:00	154
3	Psicología	44	1,740	1:30:00	99

Tabla 19

Tiempo promedio y salones utilizados en la elaboración de horarios de exámenes utilizando algoritmos genéticos

N°	Carrera	Total de curso	solicitud de sillas	Tiempo de elaboracion del horario	salones utilizados	Valor fitness f(x)
1	Administración	37	8,419	0:30:16	389	0.13847
2	Ingeniería Civil	44	2,978	1:30:47	147	0.13100
3	Psicología	44	1,740	1:10:00	90	0.14048

En la tabla 19, se muestra la utilización en forma óptima de los salones de acuerdo al número de sillas solicitadas en comparación a la tabla 18, Esto se evidencia por el valor de la función fitness que esta aproxima al cero.

- Disminuir el tiempo de elaboración del Horario de examen utilizando Los algoritmos genéticos.

  El tiempo de elaboración del horario de exámenes se muestra en la tabla 19, Se concluye que el algoritmo diseñado cumple con el objetivo, donde por cada horario el promedio de elaboración máximo es de hora y media.

5.3 Recomendaciones

PRIMERA: En las pruebas realizadas se pudo comprobar que el algoritmo genético converge con un horario de examen esperado y con el valor fitness 0 a 0.9999, lo que varía entre las pruebas es el tiempo que converge. No siempre es exacto el tiempo promedio después de 30 pruebas realizadas a una misma carrera, se da entre 30 minutos a hora y media, hay momento que el algoritmo muestra resultados a las horas de iniciar la prueba rebalsando las 3000 generaciones, para que no suceda esto se recomienda volver ejecutar el algoritmo una vez más.

SEGUNDA: para este caso no habría recomendaciones porque el algoritmo genético consigue el uso eficiente de salones.

TERCERA: Sería interesante poder mejorar la solución, tener una solución híbrida con el método de búsqueda tabú, que permita poder generar una población controlada y no al azar, como es este caso, donde el algoritmo genético pueda encontrar mejores cromosomas muy cerca al esperado y el tiempo de elaboración se reduzca .

CUARTA: En caso las carreras compartan una misma sede al momento de generar los horarios de exámenes, se debe comenzar con las carreras que tienen un mayor número de alumnos matriculados, así el algoritmo genético tendrá a su disposición todos los salones que podrá utilizarlo en forma adecuada según la solicitud de sillas por cada curso. Elaborando horarios de exámenes optimizados.

QUINTA: La solución permite poder saber con anticipación, los salones que se utilizarán en la semana de exámenes, teniendo la opción de tener distintos escenarios de horarios en cada ejecución. Pudiendo planificar en forma adecuada la utilización de los salones.

Referencias bibliográficas

- Blaz, S. (2016). *Un sistema de generación de horarios para la enseñanza de pregrado en universidades peruanas mediante algoritmos genéticos (*Tesis de pregrado) Universidad Nacional Mayor de San Marcos, Lima.
- Castro, H. (2016). *Algoritmo Genético aplicado en la programación académica de una escuela profesional en una facultada universitaria (*Tesis de pregrado) Universidad Nacional del Callao, Callao.
- Caballero, R. (2012). *Aplicación Generadora de Horario de exámenes* (Proyecto de investigación). Universidad Carlos III de Madrid, Madrid.
- Cortez, A., Rosales, G., Naupari, R. & Vega. H. (2010), *Sistema de apoyo a la generación de horarios basado en algoritmos genéticos.* Revista de Investigación de sistemas e informática, 7(1), 37-55.
- Moreno, P. & Sánchez, J. (2017), *Revisión de algoritmos de búsqueda aplicada al problema de creación de Horarios de exámenes*. Revista Tecnologí@ y desarrollo, 15(1), 4-32.
- Pitol, F. (2014). *Uso de algoritmos evolutivos para resolver el problemas de asignación de horarios escolares en la facultad de Psicología de la universidad Veracruzana* (Tesis de Maestría de inteligencia artificial). Universidad Veracruzana, Veracruz.
- Imaicela, J. & Ordoñez, A. (2012). *Desarrollo de una aplicación web que permita la gestión de reservaciones y generación automática del Horario deportivo para*

*las canchas sintéticas zona Futbol (*Tesis de pregrado). Universidad Nacional de Loja, Loja.

- Castro, H. (2016). *Algoritmo genético aplicado en la programación académica de una escuela profesional en una facultad universitaria (*Tesis de pregrado) Universidad Nacional del callao, callao.
- University of Granada. (2014). *Estrategias de Evolución y Programación Evolutiva.* Recuperado de http://sci2s.ugr.es/graduateCourses/bioinformatica
- Lozano (2004).*Algoritmos genéticos.* Recuperado de http://www.sc.ehu.es/ccwbayes/docencia/mmcc/docs/
- Universidad católica los Angeles Chimbote (2016). *Código de ética para la investigación.* Recupera de https://www.uladech.edu.pe/images/stories/universidad/documentos/2016/codigo-de-etica-para-la-investigacion-v001.pdf
- Universidad Carlos III de Madrid. (2014). *Algoritmos Genéticos, recuperado* de http://www.it.uc3m.es/jvillena/irc/practicas/06-07/
- SCRIB. (2017). *Algoritmos Genéticos* https://es.scribd.com/presentation/296449401/ALGORITMOS-GENETICOS
- Mejia.F, (2010). *algoritmos genéticos* http://nando1-utb.blogspot.pe/p/algoritmos-geneticos.html

Abreviaturas

1. **Timetabling**

   Es una lista organizada, por lo general establecida en forma de cuadro, con información sobre una serie de eventos organizados: en particular, el momento en que se ha previsto que estos eventos tendrán lugar.

2. **Heurísticas**

   Las heurísticas son procedimientos simples basados en el sentido común, que se supone con ello se obtendrá una buena solución (no necesariamente la mejor) a un problema difícil de un modo sencillo y rápido.

3. **Metaheurística**

   Es un proceso iterativo que dirige y modifica las operaciones de otras heurísticas subordinadas para producir soluciones de alta calidad.

Puede manipular una solución única (completa o incompleta) o un conjunto de ellas en cada iteración. El heurístico subordinado puede ser un procedimiento de alto (bajo) nivel, una simple búsqueda local o un método de construcción. El nombre combina el prefijo griego "meta" ("más allá", aquí con el sentido de "nivel superior") y "heurístico" (de heuriskein, "encontrar").

4. **Un algoritmo voraz**

   También conocido como ávido, devorador o greedy, es aquel que, para resolver un determinado problema, sigue una heurística consistente en elegir la opción óptima en cada paso local con la esperanza de llegar a una solución general óptima. Este esquema algorítmico es el que menos dificultades plantea a la hora de diseñar y comprobar su funcionamiento. Normalmente se aplica a los problemas de optimización.

5. **Algoritmo Grasp**

   Es una técnica de los años 80 que tiene como objetivo resolver problemas difíciles en el campo de la optimización combinatoria. Esta técnica dirige la mayor parte de su esfuerzo a construir soluciones de alta calidad que son posteriormente procesadas para obtener otras aún mejores.

6. **La programación evolutiva**

   La programación evolutiva (PE) es una rama de la Computación Evolutiva. Esta es prácticamente una variación de los algoritmos genéticos, donde lo que cambia es la representación de los individuos. En el caso de la PE los individuos son ternas (tripletas) cuyos valores representan estados de un autómata finito

7. **The University Course Timetabling Problem**

   El UCTP es un problema de asignación multidimensional, en el que se asignan estudiantes y profesores (o miembros de la facultad) a cursos, secciones de cursos o clases y eventos (reuniones individuales entre alumnos y profesores) a las aulas y los espacios de tiempo.

8. **Búsqueda Local guiada (GLS)**

Es un método de búsqueda metaheurística. La búsqueda local guiada genera penalizaciones durante una búsqueda. Utiliza penalizaciones para ayudar a los algoritmos de búsqueda locales a escapar de mínimos y mesetas locales. Cuando el algoritmo de búsqueda local determinado se establece en un óptimo local, GLS modifica la función objetivo utilizando un esquema específico. Luego, la búsqueda local funcionará utilizando una función objetivo aumentado, que está diseñada para llevar la búsqueda fuera del óptimo local. La clave está en la forma en que se modifica la función objetivo.

9. Fenotipo

Son los rasgos específicos y observables de un individuo. Por lo tanto, podría entenderse como una solución de-codificada.

10. Genotipo

Es la información contenida en el genoma de un individuo. Por lo Tanto, podría entenderse como una solución codificada.

11. Cromosoma Digital

Representación de los individuos en vector que puede ser representado en 4 o 8 bits.

ANEXOS

Anexo 1. Matriz de consistencia

Problemas	Objetivos	Hipótesis
General ¿Cómo la complejidad de la elaboración del horario de examen, se puede simplificar aplicando algoritmos genéticos, para una universidad Privada?	**General** proporcionar una solución basado en algoritmo genético, que permita elaborar en forma automática el horario de exámenes, permitiendo cumplir con los requerimientos establecidos por la universidad san Ignacio de Loyola.	**General** La utilización del algoritmo genéticos permitirá poder cumplir con los restricciones establecidas en la elaboración del calendario de exámenes de la universidad san Ignacio de Loyola.
Secundario ¿Cómo las aulas que utilizan el horario de examen se puede optimizar, con el fin de que cada curso este en el aula correcta?	**Secundario** Incrementar el uso eficiente de los salones de clase en la semana de exámenes parcial y final, utilizando los algoritmos genéticos.	**Secundario** La optimización del uso de salones se produce cuando el algoritmo genético utilizar en forma adecuada las restricciones de los salones que cuenta la universidad.
Secundario ¿Cómo la elaboración del horario de examen puede ser agilizado, utilizando los algoritmos genéticos?	**Secundario** Disminuir el tiempo de elaboración del calendario de examen utilizando los algoritmos genéticos	**Secundario** Entre menos restricciones tenga el calendario de examen, el tiempo de respuesta del algoritmo genético para la elaboración del horario será optimo.

Anexo 2: Matriz de Operacionalización

Variables	Definición	Dimensión	Indicadores
INDEPENDIENTE Algoritmos genético	El algoritmo genético crea una población al azar, selecciona dos cromosomas de la población para efectos de cruce, realiza el cambio de gen para poder crear una nueva descendencia y se evalúa el nuevo cromosoma.	Funcionamiento	Población que se genera en forma aleatoria de "N" cromosomas
			Se evalúa la aptitud f(X) de cada cromosomas de la población "X"
			Se selecciona dos cromosomas padres de una población.
			Probabilidad de "X" reproducción entre cromosomas
			Aplica Mutación a "X" gen del cromosoma
		Termino	Si la condición de prueba esta satisfecha se para el algoritmo
			Sustituir el cromosoma con la "x" generación nueva
DEPENDIENTE Elaboración del horario de exámenes optimizado	Facilita la elaboración del horario de exámenes a partir de las interrelaciones sistemáticas de sus distintas componentes que proporcionando múltiples soluciones de horarios.	Planeación	Tiempo "T" de elaboración del horario
			Utilización de "N" aulas
		Restricciones	"N" número de restricciones fuertes