

Edgar Eavir Terán-Pozo; Ariel José Romero-Fernández; Ana Lucía Sandoval-Pillajo
Luis Rafael Freire-Lescano

[DOI 10.35381/cm.v8i4.895](https://doi.org/10.35381/cm.v8i4.895)

Influencia de los algoritmos genéticos en la generación de horarios en unidad educativa

Influence of genetic algorithms on the generation of timetables in educational units

Edgar Eavir Terán-Pozo
pi.edgarjtp25@uniandes.edu.ec
Universidad Regional Autónoma de los Andes, Ambato, Tungurahua
Ecuador
<https://orcid.org/0000-0002-4366-7599>

Ariel José Romero-Fernández
ua.arielromero@uniandes.edu.ec
Universidad Regional Autónoma de los Andes, Ambato, Tungurahua
Ecuador
<https://orcid.org/0000-0002-1464-2587>

Ana Lucía Sandoval-Pillajo
ui.anasandoval@uniandes.edu.ec
Universidad Regional Autónoma de los Andes, Ibarra, Imbabura
Ecuador
<https://orcid.org/0000-0003-1463-017X>

Luis Rafael Freire-Lescano
ua.luisfreire@uniandes.edu.ec
Universidad Regional Autónoma de los Andes, Ibarra, Imbabura
Ecuador
<https://orcid.org/0000-0002-6527-6417>

Recibido: 01 de mayo 2022
Revisado: 25 de junio 2022
Aprobado: 01 de agosto 2022
Publicado: 15 de agosto 2022

Edgar Eavir Terán-Pozo; Ariel José Romero-Fernández; Ana Lucía Sandoval-Pillajo
Luis Rafael Freire-Lescano

RESUMEN

Se tiene por objetivo analizar la influencia de los algoritmos genéticos en la generación de horarios en unidad educativa. Este método se presenta a modo una opción atrayente en las soluciones de problemas de alta complejidad matemática, y con el cual se soluciona problemas de mayor dificultad en esta investigación. A partir de los cruces se seleccionan los individuos más aptos por medio de las operaciones dependiendo de los datos de la tabla 5. La Mutación interviene de una manera gradual en la conformación de los individuos añadiendo una mayor diversidad al conjunto de posibles soluciones.

Descriptores: Gobernanza de internet; administración de la comunicación; informática y desarrollo. (Tesauro UNESCO).

ABSTRACT

The objective is to analyze the influence of genetic algorithms in the generation of schedules in an educational unit. This method is presented as an attractive option in the solutions of problems of high mathematical complexity, and with which the most difficult problems are solved in this research. From the crosses, the fittest individuals are selected by means of operations depending on the data in Table 5. Mutation intervenes in a gradual way in the conformation of the individuals adding a greater diversity to the set of possible solutions.

Descriptors: Internet governance; communication administration; computers and development. (UNESCO Thesaurus).

Edgar Eavir Terán-Pozo; Ariel José Romero-Fernández; Ana Lucía Sandoval-Pillajo
Luis Rafael Freire-Lescano

INTRODUCCIÓN

En las Instituciones educativas la creación de horarios es un procedimiento administrativo, asignar un curso, una sala y una programación a un docente. Debido a que todos los periodos académicos son esporádicos en cada año lectivo, esta labor debe desarrollarse al inicio y muchas veces la información de años anteriores no puede ser utilizada debido al cambio de circunstancias o malla curricular. Los algoritmos genéticos (AG) son un subcampo de algoritmos evolutivos, computación evolutiva, metaheurística, optimización estocástica y mejoramiento.

El término según (Ramírez-Rodríguez, 2014), dice que metaheurística fue introducido por Fred Glover en 1986 y a partir de entonces han aparecido muchas propuestas de modelos o metas para delinear principales procedimientos de solución de problemas combinatorios. Las metaheurísticas según (Pichardo, 2018), sitúan conceptualmente” por encima” de las heurísticas en el sentido de que guían el diseño de éstas, pueden estar compuestas por una combinación de algunas heurísticas, por ejemplo, una metaheurística puede usar una heurística constructiva para generar una solución inicial y luego usar otra heurística de búsqueda para encontrar una mejor solución.

Existen valores sugeridos (Fonseca-Reyna, 2014), para adaptar estos parámetros, sin embargo, estos valores pueden no ser los óptimos para todo tipo de aplicaciones. También es común analizar cada parámetro por separado, sin tener en cuenta la influencia de cada uno sobre los demás. Los algoritmos utilizados son los AG en la actualidad estos son ampliamente usados (Yousef, 2016); así mismo (Luo, 2017) hace una revisión de los problemas de programación de la oferta del trabajo JavaScript (JSP), que van desde una única máquina hasta (JSP) flexible bajo AG las metaheurísticas son tácticas para delinear o perfeccionar las programaciones heurísticas con diseños a conseguir un valioso beneficio.

La importancia de los Métodos heurísticos según (Pichardo, 2018), son procedimientos eficientes para encontrar buenas soluciones. En estos métodos, la rapidez del proceso es tan importante como la calidad de la solución obtenida. Un método heurístico es un procedimiento para resolver un problema de optimización

Edgar Eavir Terán-Pozo; Ariel José Romero-Fernández; Ana Lucía Sandoval-Pillajo
Luis Rafael Freire-Lescano

bien definido mediante una aproximación intuitiva, en la que la estructura del problema se utiliza de forma inteligente para obtener una buena solución

Los AG son métodos adaptativos manejados para resolver problemas de optimización. Estos algoritmos están establecidos en el proceso genético de las entidades biológicas y representan el principio de selección natural, siendo posible con ellos desarrollar la forma de corregir dificultades del mundo real proponiendo aplicar un algoritmo genético para la generación de horarios con vistas a ofrecer una opción de procedimiento para la asignación de horarios de los docentes de la Unidad Educativa Ana Luisa Leoro una de las instituciones educativas de mayor extensión en cantidad de alumnos e infraestructura.

La asignación de horarios se sitúa como un problema que por medios tradicionales no puede ser resuelto en un tiempo razonable (Yousef, 2016), por lo cual se requiere del uso de herramientas automáticas y sofisticadas como son los algoritmos heurísticos, el proceso no es simple ya que se analiza con profundidad si existe una progresión de elementos que se transforman en limitaciones que corresponden ser consideradas al momento de crear un horario académico.

En este sentido, (Hañari-Mamani, 2016), para la selección de los parámetros, el algoritmo evolutivo funciona en base a medidas heurísticas y no determinísticas, es muy importante seleccionar los parámetros adecuados; sin embargo, no existe unmetodología que indique los valores exactos que deben asignarse ya que varían de acuerdo con la naturaleza del problema. De ahí que se localizan limitaciones físicas importantes por ejemplo que una sala no puede ser utilizada por más de un curso a la vez, o que un docente no puede realizar más de una clase en el mismo momento; limitaciones reglamentadas, en la que un alumno no debe tener concurrencia de horario entre dos de sus cursos; o restricciones legales que un docente no puede trabajar más de cierta cantidad de períodos semanales inicio de año.

La asignación de los horarios de clases, dependiendo del tamaño de la unidad educativa puede ser tan simple o complejo de llevar a cabo; tomando su resolución mucho tiempo, en algunos casos puede ser realizado en forma manual como un juego

Edgar Eavir Terán-Pozo; Ariel José Romero-Fernández; Ana Lucía Sandoval-Pillajo
Luis Rafael Freire-Lescano

de rompecabezas o utilizando medios automáticos para obtener una solución o aproximación a una buena solución. El tiempo perdido por parte de los docentes al no poder preparar con antelación sus clases, puede haber cruce de horarios así mismo la duplicidad de materias y los estudiantes no estarán seguros del horario creados así se puede evidenciar los métodos utilizados al momento de realizar un horario en la forma actual.

Por consiguiente; se tiene por objetivo analizar la influencia de los algoritmos genéticos en la generación de horarios en unidad educativa.

PROPUESTA

Para definir la influencia del algoritmo genético (ver ítem G3) en el desarrollo. Se generó los horarios del año lectivo 2019 en lenguaje de programación Python, y el IDE JetBrains PyCharm y luego se realizó un análisis con la forma tradicional.

Desarrollo del algoritmo genético

Para el desarrollo se utilizó el modelo mediante la técnica de cruce y limitación se tomó a consideración esta técnica para que resuelva el siguiente modelo de programación a través del uso de un software de optimización. Se ha encontrado una solución óptima que puede ser resuelta según trabajos consultados anteriormente. De esta manera, se establecen los horarios de los cursos para cada uno de los grupos presentados más adelante, además los algoritmos están tomando un papel más protagonista fuera de Internet. Cada vez están teniendo un uso más amplio incluso por parte de las autoridades.

Metodología XP está comprendida de 4 fases: **(G1)** planeación, **(G2)** diseño, **(G3)** codificación y **(G4)** pruebas, así por medio de la fase de planeación donde se realizó la recolección de la información de la institución consecuentemente a la fase de diseño donde se realizó el análisis de la información recolectada a través de la entrevista a las autoridades donde la fase de codificación permitió la aplicación del AG para la resolución de problemas concluyendo con la fase final fase de pruebas en producción

Edgar Eavier Terán-Pozo; Ariel José Romero-Fernández; Ana Lucía Sandoval-Pillajo
Luis Rafael Freire-Lescano

donde se desarrolló la solución a través de los AG diseñada para entregar el prototipo del software que los clientes necesitan en el momento en que lo necesitan. Programación Extrema alienta a los desarrolladores a responder a los requerimientos cambiantes de los clientes, aún en fases tardías del ciclo de vida del desarrollo.

G1. Planificación

Para la planificación se utilizó las herramientas informáticas y el software Pychar como intérprete de Python con la ayuda del personal administrativo y un técnico en el desarrollo del proceso de los algoritmos.

G2. Diseño

Mediante la entrevista realizada a la autoridad de la institución Msc. Norma Vilca rectora se dio a conocer que la institución no tiene ningún sistema de creación o forma automática para crear un horario académico, y se hizo referencia a la falta de conocimientos en nuevas tecnologías y creación, se realizaron reuniones con personal administrativo de la institución obteniendo datos importantes para la creación del algoritmo genético.

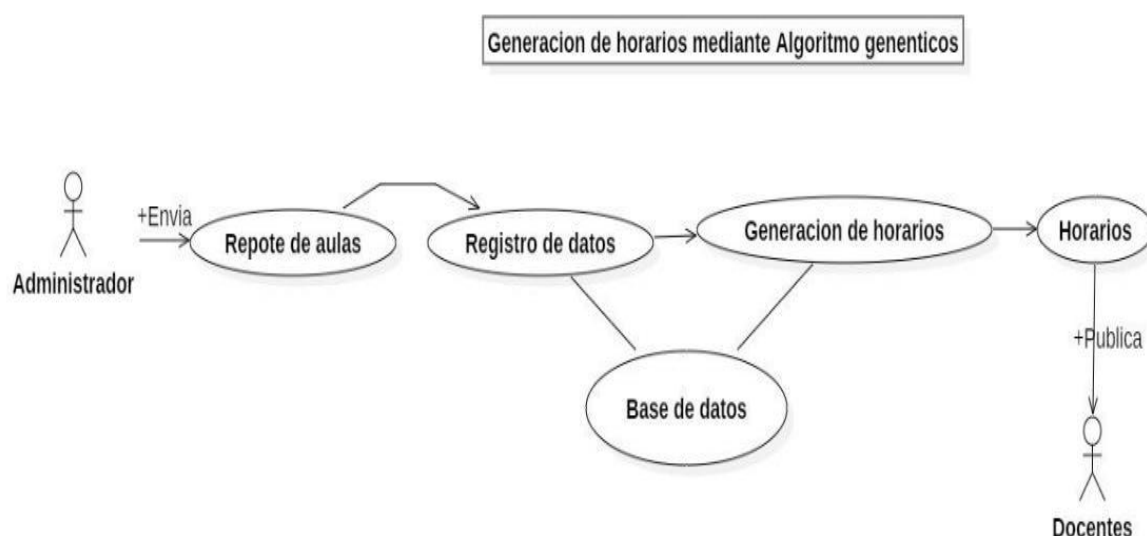


Figura 1. Administrador.

Edgar Eavir Terán-Pozo; Ariel José Romero-Fernández; Ana Lucía Sandoval-Pillajo
Luis Rafael Freire-Lescano

El administrador es el encargado de registrar los datos en una base de datos del sistema académico de la institución para mediante AG realizar la generación de horario y poder medir su funcionalidad. El docente registra los datos de las materias a dictar asignadas por la institución donde a través de AG se realiza la generación uniforme del horario según las observaciones relevantes del docente

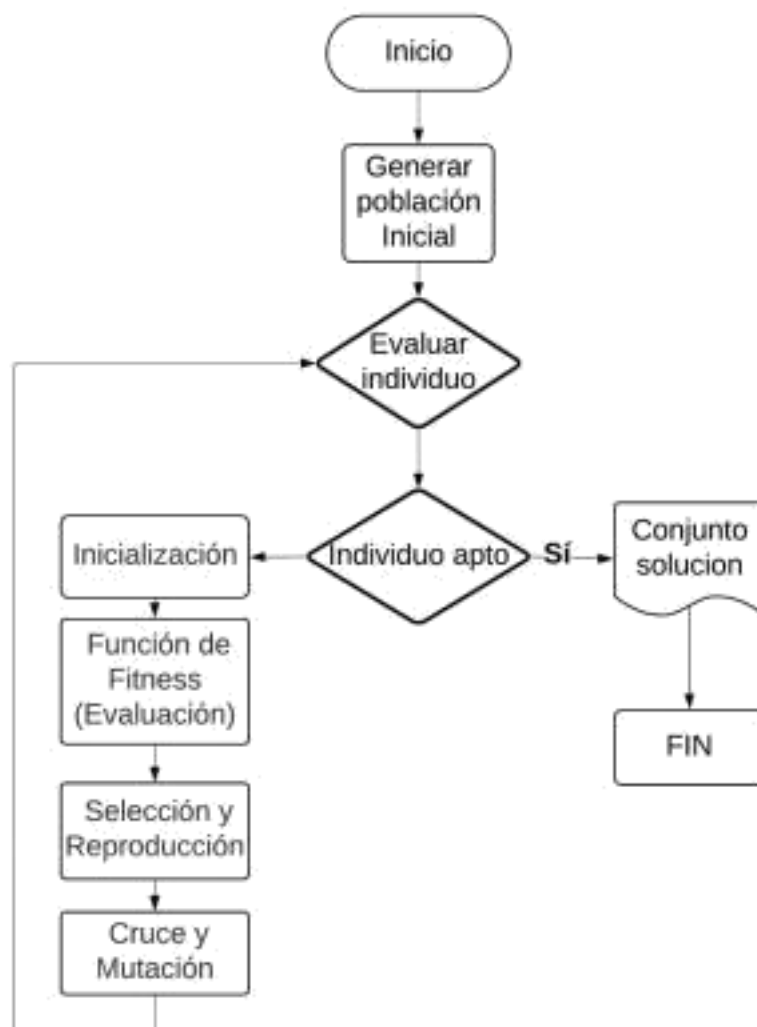


Figura 2. Diagrama de flujo del algoritmo genético usado en la generación de horarios.

Edgar Eavir Terán-Pozo; Ariel José Romero-Fernández; Ana Lucía Sandoval-Pillajo
Luis Rafael Freire-Lescano

Con respecto a la figura 2 iniciamos para generar la población inicial para evaluar al individuo sigue unos pasos si es apto inicialización, función de fitness (evaluación), selección y reproducción finalmente el cruce y mutación si el individuo es apto pasa al conjunto de solución finalmente salen los resultados seleccionados.

G3. Codificación

En primer lugar, los AG se trabajaron en 4 fases que son Inicialización, Función de Fitness (Evaluación), Selección y Reproducción además Cruce y Mutación así mismo se seleccionará los individuos más parecidos al modelo para su reproducción. Igualmente, el algoritmo conseguirá que la población se acerque más al modelo en cada generación a continuación se presenta las fases del desarrollo del algoritmo genético. Para el diseño del prototipo se empleó el lenguaje de programación Python, además de las librerías correspondientes a los AG.

Los algoritmos genéticos (AG) según (Cano, 2018), funcionan entre el conjunto de soluciones de una población natural, codificando la información de cada solución en una cadena, generalmente binaria, llamada cromosoma. Los símbolos que forman la cadena son llamados genes. Cuando la representación de los cromosomas se hace con cadenas de dígitos binarios se le conoce como genotipo. Los cromosomas evolucionan a través de iteraciones, llamadas generaciones. En cada generación, los cromosomas son evaluados usando alguna medida de aptitud. Las siguientes generaciones (nuevos cromosomas), son generadas aplicando los operadores genéticos repetidamente, siendo estos los operadores de selección, cruzamiento, mutación y reemplazo.

En relación con el lenguaje de programación (Ríos, 2016), dice que debido a la creciente interacción de los usuarios con sistemas web, surge la necesidad de combinar las funcionalidades de aplicaciones clásicas de escritorio, con la accesibilidad y bajo costo de la publicación de aplicaciones web; dando origen a la elección del mejor frameworks que trabajan con el lenguaje Python para el desarrollo de aplicaciones web es así como (Molina-Ríos, et al., 2016), dio a conocer Django

Edgar Eavir Terán-Pozo; Ariel José Romero-Fernández; Ana Lucía Sandoval-Pillajo
Luis Rafael Freire-Lescano

es el mejor framework para la implementación de desarrollo de sistemas web para su uso en esta investigación ya que se adapta para el desarrollo de aplicaciones web (Green, 2014), (Straughair, 2019).

Inicialización la población inicial puede generarse al azar, pero debe ser lo suficientemente grande y diversa para que durante la evaluación los individuos muestren un fitness distinto como la primera crea un individuo que después será guardado dentro de una matriz llamada “población”, junto al resto de individuos:

```
def individual(min, max):  
    return[random.randint(min, max) for i in range(largo)]
```

Su funcionamiento debería resultar evidente. Recibe como parámetros dos números enteros (un mínimo y un máximo) y se llena una lista de longitud da porque la variable 'largo' con números aleatorios entre el mínimo y el máximo. Esta lista creada será el nuevo individuo. Para acabar, se crea una población inicial y el bucle del programa. El algoritmo hará evolucionar a la población durante cien generaciones, llamando las funciones que se han definido arriba.

```
population = crearPoblacion  
print("Poblacion Inicial:\n%s"%(population))  
  
for i in range(100):  
    population = selection_and_reproduction(population)  
    population = mutation(population)  
print("\nPoblacion Final:\n%s"%(population))  
print("\n\n")
```

La siguiente función sirve para crear una población:

```
def crearPoblacion():  
    return [individual(1,9) for i in range(num)]
```

Llamo a la función para crear individuales un número de veces igual a 'num', que definía el tamaño de la población. Todos estos nuevos individuales los guarda dentro de una lista, que devuelve fuera de la función.

Edgar Eavir Terán-Pozo; Ariel José Romero-Fernández; Ana Lucía Sandoval-Pillajo
Luis Rafael Freire-Lescano

```
def calcularFitness(individual):  
    fitness = 0  
    for i in range(len(individual)):  
        if individual[i] == modelo[i]:  
            fitness += 1  
    return fitness
```

Función de Fitness (Evaluación) la Función de Fitness evalúa cada uno de los individuos de la población y les asigna un valor numérico según su calidad. Este valor numérico se llama *fitness*.

```
def calcularFitness(individual):  
    """  
        Calcula el fitness de un individuo concreto.  
    """  
    fitness = 0  
    for i in range(len(individual)):  
        if individual[i] == modelo[i]:  
            fitness += 1  
  
    return fitness
```

La inicialización ya está hecha (es la función crearPoblación). Ahora se va a usar una función que se llama `selection_and_reproduction()` para evaluar cada uno de los individuos (evaluación), seleccionar los mejores (selección) y mezclar su material genético (crossover) a fin de crear una nueva población encima de la anterior.

```
def selection_and_reproduction(population):  
  
    for i in range(len(population)-pressure):  
        punto = random.randint(1,largo-1)  
        padre = random.sample(selected, 2)  
  
        population[i][:punto] = padre[0][:punto]  
        population[i][punto:] = padre[1][punto:]  
  
    return population
```

Edgar Eavir Terán-Pozo; Ariel José Romero-Fernández; Ana Lucía Sandoval-Pillajo
Luis Rafael Freire-Lescano

Selección y Reproducción hay varias formas de seleccionar los individuos con el fitness más elevado para crear una nueva población también es necesaria una función de mutación, que añade pequeñas variaciones al azar en el array de los individuos de la nueva generación.

```
def mutation(population):  
    for i in range(len(population)-pressure):  
        if random.random() <= mutation_chance:  
            punto = random.randint(0,largo-1)  
            nuevo_valor = random.randint(1,9)  
  
            while nuevo_valor == population[i][punto]:  
                nuevo_valor = random.randint(1,9)  
  
            population[i][punto] = nuevo_valor  
  
    return population
```

Cruce y Mutación una vez se han seleccionado los mejores individuos, el Cruce consiste en mezclar el material genético de estos para crear los nuevos individuos. La forma más sencilla de hacerlo es mediante un One-point Cruce. Consiste en elegir un punto al azar de los dos individuos e intercambiar el material genético a partir de esta posición. Para acabar, se crea una población inicial y el bucle del programa. El algoritmo hará evolucionar a la población durante cien generaciones, llamando las funciones que se han definido arriba.

```
population = crearPoblacion()  
print("Poblacion Inicial:\n%s"%(population)) #Se muestra la poblacion inicial  
  
for i in range(100):  
    population = selection_and_reproduction(population)  
    population = mutation(population)  
    print("\nPoblacion Final:\n%s"%(population))  
print("\n\n")
```

Figura 3. Generación de horario.

Para el análisis de resultados se consideran diferentes parámetros los cuales influyen de manera en el comportamiento de los AG y en el tiempo de obtención de los resultados. Entre los parámetros introducidos están: El número de individuos que conforma la población

- La probabilidad de cruzamiento se utiliza con una posibilidad alta de acción sobre cada pareja de padres a cruzar (0.6 y 0.9), si no actúa los padres son los sucesores del proceso de mezcla de la pareja.
- El Operador de mutación en un punto seleccionado aleatoriamente, con previa probabilidad de mutación sobre cromosoma.

Los valores de entrada para un determinado de generaciones como número de docentes (D), número de materias (M), número de periodos académicos (P) y número de aulas (A) se toman para la evolución en similar de múltiples soluciones óptimas.

Figura 4. Valores de entrada.

Edgar Eavíer Terán-Pozo; Ariel José Romero-Fernández; Ana Lucía Sandoval-Pillajo
Luis Rafael Freire-Lescano

Todos los valores fueron extraídos los cuales se cargan en el prototipo, tomando parámetros fijos, como una población de 62 cromosomas que da como resultado el número de docentes de la institución se optó por este tamaño de muestra debido a la efectividad del algoritmo genético, también se considera una probabilidad de cruzamiento en un 0.95 y una probabilidad de mutación de 0.1, forma que exista una mayor cantidad de generaciones de predecesores. Por lo cual se observó la conveniencia de hacer variar los parámetros de entrada como se muestra en la figura 5.

```
Modelo: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1]

Poblacion Inicial:
[[5, 3, 2, 6, 6, 3], [1, 6, 4, 5, 7, 5], [6, 1, 2, 2, 6, 2], [1, 7, 2, 5, 6, 6], [5, 5, 5, 2, 3, 6], [7, 1, 2, 2, 2, 7], [1, 6, 6, 5, 3, 7]]

Poblacion Final:
[[1, 1, 1, 1, 1, 1], [1, 1, 1, 1, 1, 1], [1, 1, 1, 1, 1, 1], [1, 1, 1, 1, 1, 1], [1, 1, 1, 1, 1, 1], [1, 1, 1, 1, 1, 1], [1, 1, 1, 1, 1, 1]]
```

Figura 5. Prototipo algoritmo genético modelo 1 - D, M, P y A.

La influencia que ejercen los algoritmos en toda la sociedad es innegable mediante una técnica que ha sido utilizada en problemas similares y ha causado alto impacto. Con relación a los métodos que están implementado en varias funciones de los AG para la generación de horarios basados en Python según (Calle-López, 2018) La generación de horarios es un tema complicado, pero a su vez cuenta con múltiples soluciones tal como se indica más adelante en la figura 3.

La influencia de los algoritmos es cada vez mayor en nuestra vida y en nuestra sociedad para la optimización de recursos en la construcción de horarios tiene como propósito encontrar las mejores distribuciones de asignaciones docentes acorde a la institución donde además toma en cuenta las aproximaciones relacionadas por lo general y busca optimizar conjuntos de reglas para encontrar la solución más óptima posible; entre estas soluciones se encontró técnicas basadas en la capacidad de una fórmula proposicional así como el filtrado de reglas basadas en currículos

Edgar Eavir Terán-Pozo; Ariel José Romero-Fernández; Ana Lucía Sandoval-Pillajo
Luis Rafael Freire-Lescano

institucionales tomando en cuenta las asignaciones de cursos que comparten varios estudiantes donde vale la pena mencionar que los AG han destacado en la búsqueda de estas soluciones y entre otras.

CONCLUSIÓN

Siendo los resultados entregados aceptables por el personal docente y autoridades. En los resultados de técnica de cruce y control en comparación con las soluciones obtenidas de forma manual en la tabla 4 donde se observa el tiempo de creación de forma manual, se aprecia que los tiempos con el AG son menores, la diferencia es el esfuerzo invertido en ello. La debilidad del modelo en la solución fue el cumplimiento de la continuidad. Este método se presenta a modo una opción atrayente en las soluciones de problemas de alta complejidad matemática, y con el cual se soluciona problemas de mayor dificultad en esta investigación. A partir de los cruces se seleccionan los individuos más aptos por medio de las operaciones dependiendo de los datos de la tabla 5. La Mutación interviene de una manera gradual en la conformación de los individuos añadiendo una mayor diversidad al conjunto de posibles soluciones.

FINANCIAMIENTO

No monetario.

AGRADECIMIENTO

A la Universidad Regional Autónoma de los Andes; por motivar el desarrollo de la investigación.

REFERENCIAS CONSULTADAS

Cano, J. A.-E.-M. (2018). Solución del problema de conformación de lotes en almacenes utilizando algoritmos genéticos [Solution of the batching problem in warehouses using genetic algorithms]. *Información tecnológica*, 29(6), 235-244.

Edgar Eavir Terán-Pozo; Ariel José Romero-Fernández; Ana Lucía Sandoval-Pillajo
Luis Rafael Freire-Lescano

- Fonseca-Reyna, Y. (2014). Influencia de los parámetros principales de un Algoritmo Genético para el Flow Shop Scheduling [Influence of the main parameters of a Genetic Algorithm for Flow Shop Scheduling]. *Revista Cubana De Ciencias Informáticas*, 8(1). Recuperado de <https://acortar.link/AWL1bX>
- Green H. E. (2014). Use of theoretical and conceptual frameworks in qualitative research. *Nurse researcher*, 21(6), 34–38. <https://doi.org/10.7748/nr.21.6.34.e1252>
- Hañari-Mamani, S. (2016). Algoritmos evolutivos aplicados a la generación de horarios para el colegio Aplicación de la UNA-PUNO [Evolutionary algorithms applied to the generation of timetables for the school Application of UNA-PUNO]. <https://repositorioslatinoamericanos.uchile.cl/handle/2250/3274874>
- Luo, Y. (2017). Nested Optimization Method Combining Complex Method and Ant Colony Optimization to solve JSSP with complex associated processes. *J.I. of Intelligent Man*, 1801-1815. doi: [10.1007/s10845-015-1065-1](https://doi.org/10.1007/s10845-015-1065-1)
- Molina-Ríos, J., Loja Mora, M., Zea Ordóñez, M., Loaiza Sojos, E. (2016). Evaluación de los Frameworks en el Desarrollo de Aplicaciones Web con Python [Evaluation of Frameworks in Web Application Development with Python]. *Revista Latinoamericana de Ingeniería de Software*, 4(4): 201-207. <https://doi.org/10.18294/relais.2016.201-207>
- Pichardo, M. F. (2018). Revisión de algoritmos genéticos aplicados al problema de la programación de cursos universitarios [Review of genetic algorithms applied to the university course scheduling problem]. <http://riaa.uaem.mx/handle/20.500.12055/93>
- Ramírez-Rodríguez, C. O. (2014). Un algoritmo GRASP con doble relajación para resolver problema del flow shop scheduling [A GRASP algorithm with double relaxation to solve flow shop scheduling problem]. <https://tesis.pucp.edu.pe/repositorio/handle/20.500.12404/368>
- Straughair C. (2019). Reflections on developing a conceptual framework to support a constructivist grounded theory study on compassion in nursing. *Nurse researcher*, 27(1), 22–26. <https://doi.org/10.7748/nr.2019.e1621>
- Yousef, C. Salama, M. Y. Jad, T. El-Gafy, M. Matar & Habashi, S. (2016). GPU based genetic algorithm solution for the timetabling problema. 2016 11th International Conference on Computer Engineering & Systems (ICCES), 2016, pp. 103-109, doi: [10.1109/ICCES.2016.7821982](https://doi.org/10.1109/ICCES.2016.7821982).

CIENCIAMATRIA

Revista Interdisciplinaria de Humanidades, Educación, Ciencia y Tecnología

Año VIII. Vol. VIII. Nro. 4. Edición Especial 4. 2022

Hecho el depósito de ley: FA2021000002

ISSN-L: 2542-3029; ISSN: 2610-802X

Instituto de Investigación y Estudios Avanzados Koinonía (IIEAK). Santa Ana de Coro. Venezuela

Edgar Eavier Terán-Pozo; Ariel José Romero-Fernández; Ana Lucía Sandoval-Pillajo
Luis Rafael Freire-Lescano

©2022 por los autores. Este artículo es de acceso abierto y distribuido según los términos y condiciones de la licencia Creative Commons Atribución-NoComercial-CompartirIgual 4.0 Internacional (CC BY-NC-SA 4.0) (<https://creativecommons.org/licenses/by-nc-sa/4.0/>)

x