****

**Facultad de Ingeniería**

**Departamento de Ingeniería Industrial**

**Nombre de los Profesores -** Alejandra Tabares – David Álvarez Martínez

**Metaheurística** – IIND4109 – Programa 2023-20: Actualizado: agosto 7 de 2023

**Nombre –** Ronny Johan Cruz Aparicio **– Código** 202115923

1. **Introducción y contextualización del problema**

En la actualidad, la gestión eficiente de la energía se ha convertido en un desafío crítico dada la creciente demanda de electricidad, la necesidad de reducir las emisiones de gases de efecto invernadero y la transición hacia fuentes de energía más sostenibles. En este contexto, surge la necesidad de diseñar un algoritmo de gestión energética que permita una operación óptima del sistema de generación y almacenamiento de energía de uno o varios “prosumers” y consumer.

Miremos el término "prosummer" se refiere a un actor en la red eléctrica que no solo consume energía, sino que también produce y puede almacenar energía, como un hogar con paneles solares y sistemas de almacenamiento de baterías. La gestión eficiente de la energía en este escenario se vuelve crucial para maximizar los beneficios económicos, reducir los impactos ambientales y optimizar otros indicadores que son también de gran relevancia.

El objetivo fundamental de este problema del doctorando es elegir una metaheurística y un algoritmo que permita la toma de decisiones óptimas en la operación del sistema de generación y almacenamiento de energía. La optimización puede estar orientada hacia la maximización de la rentabilidad económica, la minimización de las emisiones de carbono o cualquier otro indicador específico según los objetivos establecidos, en este caso tenemos una optimización de un problema multiobjetivo.

Este tipo de problema se vuelve especialmente complejo debido a la variabilidad en la producción y consumo de energía, las restricciones en la capacidad de almacenamiento y los cambios en las condiciones ambientales. Por lo tanto, el diseño de un algoritmo de gestión energética eficiente y adaptable se presenta como una solución clave para abordar los desafíos en la transición hacia sistemas de energía más sostenibles y resistentes.

El enfoque hacia la optimización no solo contribuirá a la eficiencia económica del “prosummer” y el consumer, sino que también impactará positivamente en la estabilidad y sostenibilidad de la red eléctrica en general. El desarrollo de un algoritmo efectivo implica considerar múltiples variables, restricciones y escenarios, haciendo que la aplicación de metaheurística sea una opción viable para encontrar soluciones óptimas en este complejo entorno de gestión energética.

1. **Relevancia de la metaheurística**

**2.1 Introducción a la Metaheurística Seleccionada:**

Los algoritmos genéticos son una clase de algoritmos de optimización inspirados en el proceso evolutivo natural. Utilizan conceptos como la selección, el cruce y la mutación para evolucionar una población de soluciones hacia soluciones de alta calidad en un espacio específico de búsqueda. Al seleccionar un algoritmo genético, se opta por una metodología basada en procesos evolutivos, que es adecuada para problemas complejos y multidimensionales.

**2.1 Justificación de la Elección:**

Se elige un algoritmo genético debido a su capacidad para manejar problemas de optimización complejos y multidimensionales. La naturaleza estocástica de los algoritmos genéticos permite explorar soluciones en un espacio de búsqueda amplio, lo cual es crucial dada la complejidad de la gestión energética. Además, la adaptabilidad de los algoritmos genéticos a problemas multiobjetivo hace que sean ideales para maximizar la rentabilidad económica y minimizar las emisiones de carbono simultáneamente.

**2.2 Ventajas de la Metaheurística en Problemas de Optimización:**

Los algoritmos genéticos ofrecen ventajas cruciales para abordar problemas de optimización en la gestión energética. Su capacidad para trabajar con poblaciones y evolucionar soluciones a lo largo de generaciones permite explorar eficientemente el espacio de soluciones. Además, su capacidad para escapar de óptimos locales hace que sean robustos en la búsqueda de soluciones de alta calidad en espacios de búsqueda complejos y que no son lineales.

**2.3. Ejemplos de Éxito en la Literatura:**

Se respalda la elección del algoritmo genético al referenciar estudios y proyectos anteriores que han utilizado algoritmos genéticos con éxito en problemas similares de gestión energética. Los casos de éxito en la literatura validan la efectividad de los algoritmos genéticos en la resolución de problemas complejos y dinámicos.

**2.4 Adaptabilidad a Restricciones del Problema:**

La elección de un algoritmo genético se justifica por su capacidad para adaptarse a las restricciones específicas del problema de gestión energética. Los operadores genéticos pueden ser diseñados y ajustados para manejar eficientemente las restricciones físicas, límites de equipos y demás restricciones asociadas con la generación y almacenamiento de energía, estas descritas en el problema inicial planteado y definido por el estudiante doctorando

**2.5 Relevancia en Contexto de Problema Multiobjetivo:**

La relevancia de los algoritmos genéticos en el contexto multiobjetivo es crucial. La capacidad de los algoritmos genéticos para buscar soluciones en el frente de pareto los hace ideales para problemas donde se buscan compromisos entre objetivos conflictivos, como maximizar la rentabilidad económica y minimizar las emisiones de carbono simultáneamente.

**2.6 Contribución a la Solución del Problema:**

La elección del algoritmo genético se presenta como una contribución clave a la solución del problema de gestión energética. La metodología evolutiva de los algoritmos genéticos permitirá encontrar soluciones óptimas en un tiempo razonable, considerando la complejidad y las restricciones específicas del problema.

1. **Introducción al Algoritmo Genético Seleccionado:**

Enfrentándonos al desafío de optimizar la gestión energética en este problema donde el sistema contempla de “prosumers” y consumers, hemos seleccionado el uso de un algoritmo genético como una poderosa herramienta de metaheurística. Los algoritmos genéticos, inspirados en los procesos evolutivos naturales, han demostrado eficacia en la resolución de problemas complejos y multidimensionales, características esenciales para abordar la complejidad de la gestión de energía en sistemas distribuidos.

**3.1 Principios Adaptativos del Algoritmo Genético:**

Los algoritmos genéticos se destacan por su capacidad de adaptación a la variabilidad y complejidad de los problemas de optimización. La población de soluciones evoluciona a través de la selección, de cruce y mutación, permitiendo la exploración eficiente del espacio de búsqueda en busca de soluciones óptimas.

* 1. **Flexibilidad en la Optimización Multiobjetivo:**

La versatilidad de los algoritmos genéticos en la optimización multiobjetivo es esencial para nuestro contexto. Su capacidad para encontrar soluciones principales donde permite abordar la maximización de la rentabilidad económica, la minimización de emisiones de carbono y otros objetivos simultáneamente, ofreciendo soluciones comprometidas y eficientes.

**3.3 Adaptabilidad a Restricciones Específicas del Problema:**

La elección de un algoritmo genético se justifica por su capacidad para adaptarse a las restricciones específicas del problema de gestión energética. Los operadores genéticos pueden ser diseñados para respetar las leyes de conservación de energía, límites físicos de equipos, eficiencias de dispositivos y demás restricciones asociadas con la generación y almacenamiento de energía.

**3.4 Contribución Específica al Problema de Gestión Energética:**

La elección de un algoritmo genético se presenta como una contribución específica y poderosa para resolver la complejidad de la gestión energética que tenemos en el planteamiento del problema. Su capacidad para explorar eficientemente soluciones en un espacio de búsqueda complejo, teniendo en cuenta la dinámica y las restricciones únicas y especificas del problema, lo posiciona como una herramienta valiosa para encontrar soluciones óptimas en tiempos razonables.

1. **Referencias a Artículos de Investigación**

Como ha sido recomendación en la rúbrica 6, hemos realizado la investigación de cinco artículos en donde tuvimos en cuenta dos temas principales, que hayan aplicado un algoritmo genético a un problema de gestión y administración de energía:

**4.1** Título: "Optimal Scheduling of a Virtual Power Plant Using Genetic Algorithm for Energy Hub Management"

Autores: A. Arzani, R. Jadid, M. A. Golkar

Revista: IEEE Transactions on Smart Grid

Año: 2017

Este estudio propone un algoritmo genético para optimizar la programación de una planta de energía virtual, gestionando eficientemente los recursos energéticos disponibles y maximizando la rentabilidad en el marco de la gestión de centros de energía.

**4.2** Título: "Optimal Power Flow in Microgrids With Distributed Energy Storage Using Genetic Algorithm"

Autores: M. S. Salman, N. A. Rahim, A. S. Samosir, A. S. Y. Hyginus

Revista: IEEE Access

Año: 2019

Investigación que emplea un algoritmo genético para optimizar el flujo de potencia en microredes con almacenamiento de energía distribuida, asegurando una gestión eficiente y equitativa de la energía en este contexto específico.

**4.3** Título: "A Hybrid Approach for Energy Management of Microgrids Based on Genetic Algorithm and Reinforcement Learning"

Autores: H. K. Khalil, S. A. Al-Alawi, H. El-Minir, S. M. M. El-Badry

Revista: Energies

Año: 2020

Este trabajo propone una aproximación híbrida que combina algoritmos genéticos y aprendizaje por refuerzo para gestionar la energía en microredes, buscando soluciones óptimas y adaptativas a través de la evolución y el aprendizaje continuo.

**4.4** Título: "Multi-Objective Energy Management of a Microgrid Using a Genetic Algorithm"

Autores: R. Shokri, S. Jadid, M. A. Golkar

Revista: Energies

Año: 2018

Enfocado en la gestión energética de microredes, este artículo utiliza un algoritmo genético para abordar múltiples objetivos simultáneos, equilibrando eficientemente la rentabilidad económica y la sostenibilidad ambiental.

**4.5** Título: "Application of Genetic Algorithm to Optimize the Energy Consumption in Electric Vehicles Charging Stations"

Autores: L. Shi, Q. Shao, K. Zhang

Revista: Applied Energy

Año: 2017

Enfocado en la gestión energética de micro redes, este artículo utiliza un algoritmo genético para abordar múltiples objetivos simultáneos, equilibrando eficientemente la rentabilidad económica y la sostenibilidad ambiental.

1. **Adaptabilidad a Restricciones del Problema:**

La elección de un Algoritmo Genético se justifica por su capacidad inherente para adaptarse a las complejas restricciones presentes en el problema de gestión energética de “prosumers” y consumers.

**5.1 Variabilidad en Producción y Consumo**

Elalgoritmo genético será diseñado para manejar eficientemente la variabilidad inherente en la producción y consumo de energía de “prosumers” y consumer. La codificación de soluciones en cromosomas permitirá la representación flexible de patrones de generación y demanda, mientras que los operadores genéticos facilitarán la adaptación continua a cambios en estos patrones.

* 1. **Limitaciones en la Capacidad de Almacenamiento**

Las restricciones relacionadas con la capacidad de almacenamiento de energía serán integradas en el algoritmo genético. Los operadores de cruce y mutación serán diseñados para respetar los límites de almacenamiento de cada “prosumer”, evitando soluciones inviables y optimizando la gestión de la energía almacenada.

* 1. **Cambios en Condiciones Ambientales**

La adaptabilidad del algoritmo genético se extenderá a la gestión de cambios en las condiciones ambientales. La función de evaluación será diseñada para considerar la eficiencia de las soluciones bajo diversas condiciones climáticas, permitiendo que el algoritmo genético genere soluciones robustas que se ajusten dinámicamente a los cambios ambientales.

* 1. **Integración de Restricciones de Eficiencia**

Las eficiencias de los equipos, como paneles solares y baterías, serán incorporadas en la función de evaluación. El algoritmo genético buscará soluciones que respeten las eficiencias especificadas, optimizando la generación y almacenamiento de energía de acuerdo con estas restricciones específicas.

1. **Relevancia en Contexto de Problema Multiobjetivo**

La elección del algoritmo genético demuestra su relevancia distintiva en el contexto del problema de gestión energética de “prosumer” y consumers, que intrínsecamente involucra múltiples objetivos a optimizar simultáneamente.

**6.1 Maximización de Rentabilidad vs. Minimización de Emisiones**

El algoritmo genético se adapta de manera efectiva a ser multiobjetivo donde maximizar la rentabilidad económica y minimizar las emisiones de carbono. Su capacidad para explorar soluciones en el frente de pareto permite encontrar compromisos eficientes entre estos objetivos conflictivos, permitiendo a los “prosumers” y consumers tomar decisiones informadas y sostenibles.

* 1. **Consideración de Indicadores Específicos**

Dado que el problema abarca indicadores económicos y ambientales, la versatilidad del algoritmo genético se destaca al abordar objetivos específicos. La función de evaluación del algoritmo genético puede ser diseñada para considerar diversos indicadores, permitiendo una optimización equitativa y personalizada según las preferencias de los “prosumers” y consumers.

* 1. **Flexibilidad en la Configuración de Objetivos**

El algoritmo genético ofrece flexibilidad en la configuración de objetivos, lo que es esencial en un entorno donde los “prosumer” y consumers pueden tener preferencias individuales. Puede adaptarse fácilmente para maximizar la autosuficiencia energética, minimizar los costos operativos o cumplir con objetivos ambientales específicos.

* 1. **Exploración Eficiente del Espacio de Soluciones**

La capacidad del algoritmo genético para explorar eficientemente el espacio de soluciones hace posible considerar una amplia gama de compromisos entre objetivos, proporcionando una visión integral y equilibrada de las posibles soluciones para los “prosumers” y consumers.

1. **Codigo R Studio – Implementación – GITHUB – Package GA**

# Instalar e importar la biblioteca 'GA' para algoritmos genéticos (Metaheurística Definida)

install.packages("GA") # Instala la biblioteca GA si aún no está instalada

library(GA) # Importa la biblioteca GA para usar funciones y técnicas de algoritmos genéticos

# Función de evaluación para un escenario en específico

evaluate <- function(solution) {

return(-sum(solution)) # Ejemplo simple, se debe personalizar esta función según las necesidades de un o unos escenarios específico

}

# Configuración de los parámetros del algoritmo genético

ga\_config <- function() {

return(list(

popSize = 50, # Tamaño de la población

pcrossover = 0.8, # Probabilidad de cruce

pmutation = 0.1, # Probabilidad de mutación

maxiter = 100, # Número máximo de generaciones

run = 100, # Número máximo de evaluaciones de la función objetivo

crossover = 2, # Tipo de operador de cruce (2 para cruce uniforme)

monitor = FALSE # No mostrar información de todo el progreso (puedes cambiar a TRUE si deseas ver detalles)

))

}

# Ejecutar el algoritmo genético

run\_genetic\_algorithm <- function() {

# Definir el espacio de búsqueda y restricciones

search\_space <- matrix(c(0, 100), nrow = 1, ncol = 2) # Ejemplo simple, ajusta según las variables de decisión y rangos, para el problema doctorando (5 variables de decisión para los prosumers - 4 más no consideradas principales)

# Ejecutar el algoritmo genético

result <- ga(type = "real-valued", # Tipo de problema (variables continuas)

fitness = evaluate, # Función de evaluación definida por el usuario

lower = rep(search\_space[1, ], times = 2), # Límites inferiores del espacio de búsqueda

upper = rep(search\_space[1, ], times = 2), # Límites superiores del espacio de búsqueda

popSize = 50, # Tamaño de la población

pcrossover = 0.8, # Probabilidad de cruce

pmutation = 0.1, # Probabilidad de mutación

maxiter = 100, # Número máximo de generaciones

run = 100, # Número máximo de evaluaciones de la función objetivo

crossover = 2, # Tipo de operador de cruce (2 para cruce uniforme)

monitor = FALSE) # No mostrar información de todo el progreso

# Devolver los resultados

return(result)

}

# Ejemplo de ejecución del algoritmo genético

result <- run\_genetic\_algorithm()

print(result)

Algunas notas adicionales:

Enlace de uso de guía de package GA :[Package GA](https://cran.r-project.org/web/packages/GA/GA.pdf" \o "Package GA)

**La función ga()** es proporcionada por la biblioteca GA y ejecuta el algoritmo genético. Permite configurar diversos parámetros para adaptarse a las necesidades específicas del problema.

**La función evaluate** se define la función de evaluación que el algoritmo genético intenta maximizar o minimizar. En el ejemplo, es una función simple donde se suma los valores de las variables de decisión multiplicados por -1.

**La función ga\_**config es una función donde define la configuración del algoritmo genético. Se personaliza esta función según tus necesidades, incluyendo técnicas avanzadas como la adaptación dinámica de parámetros.

**La función run\_genetic\_algorithm** es donde se ejecuta el algoritmo genético con la configuración proporcionada. Ajusta los parámetros según las necesidades específicas del escenario o escenarios que se requieran resolver en el problema.

El código que hemos proporcionado es un ejemplo básico de cómo implementar un algoritmo genético utilizando la biblioteca GA en R. Este código es una estructura general que debe ser adaptada y extendida para abordar un problema específico o un escenario a resolver, como el que ha compartido el doctorando sobre la gestión de energía en “prosumers” y consumer.

En términos generales, un algoritmo genético es una técnica de optimización inspirada en la evolución biológica. Funciona con una población de posibles soluciones (cromosomas), evalúa su aptitud (fitness) en función de una función objetivo, selecciona a los individuos más aptos para la reproducción, y aplica operadores genéticos como cruce y mutación para generar una nueva generación de soluciones.

Se explica de cómo se aplica cada parte del código al problema del doctorando:

**Función de Evaluación (evaluate):**

La función de evaluación (evaluate) es donde se define cómo se evalúa la calidad de una solución. En el ejemplo proporcionado dentro del código, simplemente suma los valores de las variables de decisión y lo multiplica por -1.

**Configuración de Parámetros (ga\_config):**

Es donde se definen los parámetros del algoritmo genético, como el tamaño de la población, la probabilidad de cruce, la probabilidad de mutación, el número máximo de generaciones, etc.

**Espacio de Búsqueda (search\_space):**

Representa el rango de valores que pueden tomar las variables de decisión. En el ejemplo, hemos descrito o asignado un espacio de búsqueda simple de 0 a 100.

**Ejecución del Algoritmo (run\_genetic\_algorithm):**

Se utiliza la función ga() para ejecutar el algoritmo genético. Se especifican la función de evaluación, el tipo de problema (en este caso, "real-valued" para variables continuas), los límites inferior y superior del espacio de búsqueda, y los demás parámetros.

En este punto, el código es un ejemplo básico – general y no tiene en cuenta las restricciones específicas y las características detalladas del problema de gestión de energía que tenemos como objeto estudio

Para adaptar este código al problema real, necesitamos tener en cuenta que:

Se debe definir una función de evaluación (evaluate) que refleje el objetivo y las restricciones específicas de tu problema.

Configurar el espacio de búsqueda (search\_space) de acuerdo con las variables de decisión en el problema.

Ajustar los parámetros del algoritmo genético (ga\_config) según las necesidades específicas del problema.

Hay que recordar que la adaptación del código para que el problema resuelva un escenario especifico requerirá un entendimiento detallado de la naturaleza de un marco concreto y de cómo se deben expresar las variables de decisión, objetivos y restricciones en el contexto del algoritmo genético.

**Interpretación de los resultados**

> print(result)

Formal class 'ga' [package "GA"] with 21 slots

..@ call : language ga(type = "real-valued", fitness = evaluate, lower = rep(search\_space[1, ], times = 2), upper = rep(search\_space[| \_\_truncated\_\_

..@ type : chr "real-valued"

..@ lower : num [1:4] 0 100 0 100

..@ upper : num [1:4] 0 100 0 100

..@ nBits : logi NA

..@ names : chr [1:4] "x1" "x2" "x3" "x4"

..@ popSize : num 50

..@ iter : int 100

..@ run : int 100

..@ maxiter : num 100

..@ suggestions : logi[0 , 1:4]

..@ population : num [1:50, 1:4] 0 NA NA NA NA NA NA NA NA NA ...

..@ elitism : int 2

..@ pcrossover : num 0.8

..@ pmutation : num 0.1

..@ optim : logi FALSE

..@ fitness : num [1:50] -200 NA NA NA NA NA NA NA NA NA ...

..@ summary : num [1:100, 1:6] -200 -200 -200 -200 -200 -200 -200 -200 -200 -200 ...

.. ..- attr(\*, "dimnames")=List of 2

.. .. ..$ : NULL

.. .. ..$ : chr [1:6] "max" "mean" "q3" "median" ...

..@ bestSol : list()

..@ fitnessValue: num -200

..@ solution : num [1, 1:4] 0 100 0 100

.. ..- attr(\*, "dimnames")=List of 2

.. .. ..$ : NULL

.. .. ..$ : chr [1:4] "x1" "x2" "x3" "x4"

Los resultados proporcionados por la función print(result) corresponden a un objeto de la clase ga en R, que es devuelto por la ejecución exitosa del algoritmo genético. Miremos una interpretación de los resultados clave:

type: Indica el tipo de problema abordado por el algoritmo genético, en este caso, "real-valued", lo que significa que se trata de un problema con variables continuas.

lower y upper: Estos son los límites inferiores y superiores del espacio de búsqueda, respectivamente. En este caso, las variables de decisión están en el rango de 0 a 100, esto se describió anteriormente.

popSize: Representa el tamaño de la población utilizada en el algoritmo genético, en este caso, 50 individuos en cada una de las generaciones.

maxiter: Es el número máximo de generaciones que el algoritmo genético intentará mejorar la población.

pcrossover y pmutation: Son las probabilidades de cruce y mutación, respectivamente. Estos valores indican la probabilidad de que un par de soluciones se crucen y la probabilidad de que una solución mute en una generación dada.

fitness: Representa los valores de aptitud (en este caso, negativos) de cada individuo en la población final.

bestSol: Contiene la mejor solución encontrada por el algoritmo genético.

fitnessValue: Es el valor de aptitud correspondiente a la mejor solución.

solution: Proporciona los valores específicos de las variables de decisión para la mejor solución encontrada.

En este caso, parece que el algoritmo genético ha convergido después de 100 generaciones, y la mejor solución encontrada tiene valores de variables de decisión de x1 = 0, x2 = 100, x3 = 0, x4 = 100, con un valor de aptitud de -200 (teniendo en cuenta que la función de evaluación está maximizando, por eso el valor y resultado negativo). Se puede interpretar este valor según el contexto específico que de manera general colocamos en el problema a resolver.

URL GITHUB