Universidad Autónoma
Metropolitana

unidad cuajimalpa

Resolución del Problema de Rutas Vehiculares con Ventanas de Tiempo mediante un Algoritmo Híbrido entre Colonia de Hormigas y Recocido Simulado

Proyecto Terminal

que presenta:
alejandro martínez guzmán

Licenciatura en Ingeniería en Computación

Departamento de Matemáticas Aplicadas e Ingeniería

División de Ciencias Naturales e Ingeniería

Asesor:
EDWIN MONTES OROZCO

Junio 2025

Declaración

Yo, ALEJANDRO MARTÍNEZ GUZMÁN, declaro que este trabajo titulado *«Resolución del Problema de Rutas Vehiculares con Ventanas de Tiempo mediante un Algoritmo Híbrido entre Colonia de Hormigas y Recocido Simulado»* es de mi autoría. Asimismo, confirmo que:

- Este trabajo fue realizado en su totalidad para la obtención de grado en esta Universidad.
- Ninguna parte de esta tesis ha sido previamente sometida a un examen de grado o titulación en esta u otra institución.
- Todas las citas han sido debidamente referenciadas y atribuidas a sus autores.

Firma: _____

Fecha: _____

Resumen

Aquí va el resumen en español. Este apartado debe sintetizar brevemente el objetivo del trabajo, la metodología empleada y los resultados más relevantes.

Abstract

Here goes the abstract in English. Briefly describe the goal of your project, methodology, and key results.

Dedicatoria

A mis padres y profesores, por su apoyo incondicional.

Agradecimientos

Aquí van los agradecimientos a las personas e instituciones que contribuyeron al desarrollo de este proyecto.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

Capítulo 2

Marco Teórico

2.1. Antecedentes

2.2. Conceptos Clave

2.2.1. El Problema en Optimización

La optimización se refiere al proceso de buscar, dentro de un conjunto de soluciones factibles, aquella que maximice o minimice una función objetivo determinada [1].

Formalmente, un problema de optimización se define como la búsqueda de un vector x en un conjunto factible $S \subseteq \mathbb{R}^n$ que minimice o maximice una función objetivo $f : S \to \mathbb{R}$, también denominada función costo o beneficio [3].

Esta disciplina se clasifica principalmente en dos tipos: la *optimización continua*, que trabaja con variables que pueden tomar valores dentro de un rango continuo, y la *optimización discreta*, que se enfoca en problemas donde las variables sólo pueden asumir valores discretos o enteros [3].

Optimización Continua

La optimización continua se refiere a la búsqueda de soluciones óptimas en un conjunto factible $S \subseteq \mathbb{R}^n$, donde las variables pueden tomar valores dentro de un intervalo continuo. En este contexto, el objetivo es encontrar un vector $x_{opt} \in S$ que minimice o maximice una función objetivo $f : S \to \mathbb{R}$.

Formalmente, para problemas de minimización, la solución óptima global x_{opt} satisface:

$$f(x_{opt}) \leq f(x), \quad \forall x \in S,$$

mientras que para maximización, se cumple que:

$$f(x_{opt}) \geq f(x), \quad \forall x \in S.$$

El valor $f(x_{opt})$ representa el costo o beneficio óptimo, y el conjunto de todas las soluciones óptimas se denota como S_{opt} [3].

Optimización Discreta o Combinatoria

Un problema de optimización combinatoria se formaliza mediante una pareja (S, f), donde S es un conjunto finito de soluciones posibles y $f : S \to \mathbb{R}$ es la función objetivo o función costo que asigna un valor real a cada solución [3].

El objetivo es encontrar una solución óptima global $i_{opt} \in S$ que cumpla:

$$f(i_{opt}) \leq f(i), \quad \forall i \in S,$$

para problemas de minimización, o bien:

$$f(i_{opt}) \geq f(i), \quad \forall i \in S,$$

en el caso de maximización.

2.2.2. Complejidad Computacional

La complejidad computacional estudia los recursos necesarios para resolver un problema, independientemente del algoritmo utilizado. A diferencia de la complejidad algorítmica, que se enfoca en el tiempo o espacio que un algoritmo específico consume, la complejidad computacional analiza la dificultad inherente del problema desde una perspectiva teórica, considerando el trabajo, la estructura matemática y lógica implicada en su resolución [6].

2.2.3. El Problema P vs NP

En teoría de la computación, los problemas pueden clasificarse en dos grandes grupos: decidibles e indecidibles. Un problema se considera *decidible* si existe un algoritmo que puede resolverlo en un número finito de pasos. Por el contrario, un problema es *indecidible* si no existe un algoritmo general que permita resolverlo, incluso con recursos ilimitados [6].

Desde otro enfoque, esta clasificación también puede asociarse con la noción de compresibilidad: un problema decidible puede expresarse o condensarse en una forma computacional comprensible, mientras que los indecidibles son inherentemente incompresibles [6].

Clases de Complejidad Básicas

Clase P: incluye todos los problemas que pueden resolverse en tiempo polinomial por una máquina determinista. Es decir, existe un algoritmo eficiente que resuelve cualquier instancia del problema en un tiempo acotado por una función polinomial respecto al tamaño de entrada [7].

Clase NP: contiene problemas para los cuales una solución puede verificarse (no necesariamente encontrarse) en tiempo polinomial por una máquina determinista. Se desconoce si todos los problemas de NP pueden resolverse también en tiempo polinomial, lo que da origen al famoso problema abierto en computación:

$$P \stackrel{?}{=} NP \tag{2.1}$$

Este dilema ha sido objeto de extensas investigaciones. Una de las formulaciones más comunes consiste en preguntarse si todo problema cuya solución puede verificarse eficientemente (NP) también puede resolverse eficientemente (P), es decir, si $P = NP$, o si, por el contrario, $P \neq NP$ [5].

NP-Completo y NP-Difícil

Dentro de NP, algunos problemas destacan por su complejidad: los *problemas NP-completos* son aquellos que, además de pertenecer a NP, son al menos tan difíciles como cualquier otro problema en NP. Si se encontrara un algoritmo polinomial para resolver un solo problema NP-completo, todos los problemas en NP también podrían resolverse eficientemente [4].

Por otro lado, los *problemas NP-difíciles* (NP-hard) son aún más generales: no necesariamente pertenecen a NP, pero cualquier problema en NP puede reducirse a ellos en tiempo polinomial. Como menciona Maldonado (2013), los problemas NP-duros implican un tratamiento en tiempo polinomial no determinista, mientras que los NP-completos son una subcategoría de aquellos que además pertenecen a NP [6].

2.2.4. Algoritmos Exactos

2.3. Heurísticas

Una heurística es un procedimiento de resolución de problemas que proviene del término griego *heuriskein*, que significa "encontrar o descubrir". En el contexto de la investigación operacional, se define como:

> "Un procedimiento para el que se tiene un alto grado de confianza en que encuentra soluciones de alta calidad con un coste computacional razonable, aunque no se garantice su optimalidad ni su factibilidad." [2]

Las heurísticas son métodos aproximados que se utilizan cuando:

- No existe un método exacto conocido para resolver el problema.
- Los métodos exactos son computacionalmente muy costosos o inviables.
- Se requiere mayor flexibilidad en el modelado del problema.
- El problema presenta características de alta complejidad, no linealidad o gran dimensionalidad.

Características de una Buena Heurística

Para que un algoritmo heurístico sea considerado de calidad, debe cumplir con tres propiedades fundamentales:

1. Eficiencia

- El esfuerzo computacional debe ser realista y proporcional al problema a resolver.
- Debe ejecutarse en un tiempo computacional razonable.

2. Calidad de la Solución

- La solución obtenida debe estar, en promedio, próxima al óptimo global.
- Debe proporcionar consistentemente soluciones de alta calidad.

3. Robustez

- La probabilidad de obtener soluciones de baja calidad (distantes del óptimo) debe ser mínima.
- Debe mantener un rendimiento estable ante variaciones en los parámetros del problema.

Ventajas de los Métodos Heurísticos

Los métodos heurísticos ofrecen mayor flexibilidad y adaptabilidad que las técnicas exactas para abordar las características complejas de los problemas del mundo real. Su principal valor radica en la capacidad de encontrar soluciones de buena calidad en tiempos computacionales breves, constituyendo una alternativa viable cuando los métodos exactos resultan impracticables.

En síntesis, una heurística representa un enfoque inteligente de resolución que equilibra la calidad de la solución con la eficiencia computacional, siendo especialmente útil para problemas de optimización combinatoria donde los métodos exactos no son factibles debido a su complejidad computacional.

2.4. Metaheurísticas

Esta sección se basa en la exposición realizada por [2], quien describe las metaheurísticas como una evolución natural de los métodos heurísticos, desarrolladas para superar las limitaciones de las heurísticas tradicionales. El término fue introducido por Fred Glover en 1986 y se define como:

> "Una clase de métodos aproximados diseñados para resolver problemas de difícil optimización combinatoria, en los que los heurísticos clásicos no son efectivos."

Características Fundamentales

Las metaheurísticas se caracterizan por:

- **Marco conceptual general:** Proporcionan estrategias de alto nivel que pueden aplicarse a diversos problemas de optimización con modificaciones mínimas.
- **Hibridación:** Combinan conceptos de diferentes campos como genética, biología, física, inteligencia artificial y neurología.
- **Flexibilidad:** Pueden adaptarse a distintos tipos de problemas manteniendo su estructura básica.
- **Mejora iterativa:** Evolucionan las soluciones a través de procesos iterativos guiados.

Propiedades Deseables de una Metaheurística

Una metaheurística de calidad debe poseer las siguientes características:

Propiedades Básicas

- **Simplicidad:** Basada en principios claros y comprensibles.
- **Precisión:** Pasos y fases formulados en términos concretos.
- **Coherencia:** Elementos que se derivan naturalmente de sus principios.

Propiedades de Rendimiento

- **Eficacia:** Alta probabilidad de alcanzar soluciones óptimas en casos realistas.
- **Eficiencia:** Aprovechamiento óptimo de recursos computacionales (tiempo y memoria).
- **Robustez:** Comportamiento estable ante alteraciones del modelo.

Propiedades de Aplicabilidad

- **Generalidad:** Utilizable con buen rendimiento en una amplia variedad de problemas.
- **Adaptabilidad:** Capacidad de adaptación a diferentes contextos de aplicación.
- **Interactividad:** Permite la incorporación de conocimiento del usuario.

Clasificación de Metaheurísticas

Las metaheurísticas pueden clasificarse según su inspiración:

- **Inspiradas en la Física:** *Recocido Simulado (Simulated Annealing)*, basado en el proceso de calentamiento y enfriamiento de metales para obtener estados de baja energía.
- **Inspiradas en la Evolución:** *Algoritmos Genéticos*, desarrollados por John Holland, basados en los mecanismos de selección natural y reproducción sexual.
- **Inspiradas en la Biología:** *Optimización por Colonias de Hormigas (Ant Colony Optimization)*, que simula el comportamiento de comunicación de las hormigas mediante feromonas para encontrar caminos óptimos.

Ventajas de las Metaheurísticas

Las metaheurísticas representan una de las mejores aproximaciones para abordar problemas de optimización combinatoria debido a:

- Su capacidad de escapar de óptimos locales.
- La posibilidad de manejar espacios de solución complejos.
- Su adaptabilidad a problemas del mundo real.

- El equilibrio entre exploración y explotación del espacio de búsqueda.

En resumen, las metaheurísticas constituyen marcos de trabajo de propósito general que guían el diseño de algoritmos de optimización, ofreciendo estrategias sofisticadas para la resolución de problemas complejos donde las heurísticas tradicionales resultan insuficientes.

2.5. Referencias Relevantes

Capítulo 3

Materiales y Métodos

Capítulo 4

Resultados

Capítulo 5

Conclusiones

Bibliografía

[1] Mokhtar S. Bazaraa, Hanif D. Sherali, and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms*. Wiley, 3rd edition, 2006.

[2] Orlando de Antonio Suárez. Una aproximación a la heurística y metaheurísticas. *INGE@UAN – Tendencias en la Ingeniería*, 1(2):–, 2014. Publicado el 4 de marzo de 2014.

[3] Sergio Gerardo de los Cobos Silva, John Goddard Close, Miguel Ángel Gutiérrez Andrade, and Alma Edith Martínez Licona. *Búsqueda y exploración estocástica*. Libros CBI, Universidad Autónoma Metropolitana Iztapalapa, México, 2010.

[4] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

[5] Richard J. Lipton. *The P = NP Question and Gödel's Lost Letter*. Springer, 2010.

[6] Carlos Maldonado. Un problema fundamental en la investigación: Los problemas p vs. np. *Revista Logos, Ciencia & Tecnología*, 4(2):10–20, 2013.

[7] Michael Sipser. *Introduction to the Theory of Computation*. Cengage Learning, 3rd edition, 2012.

Apéndices