

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

|   |             |
|---|-------------|
| <b>Resumen</b>  | <b>III</b>  |
| <b>Abstract</b>   | <b>v</b>    |
| <b>Dedicatoria</b>                                      | <b>VII</b>  |
| <b>Agradecimientos</b>                                  | <b>IX</b>   |
| <b>Índice de Figuras</b>                                | <b>XIII</b> |
| <b>Índice de Tablas</b>                                 | <b>xv</b>   |
| <b>Índice de Fórmulas</b>                               | <b>xvii</b> |
| <b>Lista de Abreviaciones</b>                           | <b>xix</b>  |
| <b>1 Introducción</b>                                   | <b>1</b>    |
| 1.1 Planteamiento del Problema . . . . .                | 1           |
| 1.2 Justificación . . . . .                             | 2           |
| 1.3 Objetivos . . . . .                                 | 2           |
| 1.3.1 Objetivo General . . . . .                        | 2           |
| 1.3.2 Objetivos Específicos . . . . .                   | 2           |
| 1.4 Alcances y Limitaciones . . . . .                   | 3           |
| 1.5 Estructura del Documento . . . . .                  | 3           |
| <b>2 Marco Teórico / Estado del Arte</b>                | <b>5</b>    |
| 2.1 Antecedentes . . . . .                              | 5           |
| 2.2 Conceptos Clave . . . . .                           | 6           |
| 2.2.1 El Problema en Optimización . . . . .             | 6           |
| 2.2.1.1 Optimización Continua . . . . .                 | 7           |
| 2.2.1.2 Optimización Discreta o Combinatoria . . . . .  | 7           |
| 2.2.2 Complejidad Computacional y Algorítmica . . . . . | 8           |
| 2.2.3 El Problema P vs NP . . . . .                     | 9           |
| 2.2.3.1 Clases P y NP . . . . .                         | 10          |
| 2.2.3.2 NP-Completo y NP-Difícil . . . . .              | 11          |
| 2.2.4 Problemas de Optimización Combinatoria . . . . .  | 12          |
| 2.2.4.1 Problema del Agente Viajero (TSP) . . . . .     | 12          |

|  |           |
|--|-----------|
| 2.2.4.2 Problema del Problema Ruteo de Vehículos (VRP) . . . . .       | 14        |
| 2.2.4.3 Problema de Ruteo de Vehículos con Ventanas de Tiempo (VRP-TW) | 17        |
| 2.2.5 Métodos de Solución para el VRPTW . . . . .                      | 20        |
| 2.2.5.1 Métodos Exactos . . . . .                                      | 21        |
| 2.2.5.2 Heurísticas . . . . .  | 21        |
| 2.2.5.3 Mateheurísticas . . . . .                                      | 22        |
| 2.2.5.4 Algoritmos Híbridos . . . . .                                  | 24        |
| 2.3 Referencias Relevantes . . . . .                                   | 24        |
| <b>3 Materiales y Métodos</b>  | <b>25</b> |
| 3.1 Materiales . . . . .   | 25        |
| 3.2 Métodos . . . . .  | 25        |
| <b>4 Resultados</b>  | <b>27</b> |
| <b>5 Conclusiones</b>  | <b>29</b> |
| <b>Apéndices</b>   | <b>35</b> |

# Índice de figuras

|     |  |    |
|-----|--|----|
| 2.1 | Tasa de crecimiento de varias funciones en análisis de algoritmos [23]. . . . .                            | 9  |
| 2.2 | Clasificación de los problemas computacionales en decidibles e indecidibles [21]. . . . .                  | 10 |
| 2.3 | Relación entre las clases de complejidad: P, NP, NP-completo y NP-difícil.                                 | 11 |
| 2.4 | Ejemplificación de una solución al Problema del Agente Viajero (TSP). . . .                                | 13 |
| 2.5 | Ejemplificación de una solución al Problema de Ruteo de Vehículos (VRP).                                   | 15 |
| 2.6 | Ejemplificación de una solución al Problema de Ruteo de Vehículos con Ventanas de Tiempo (VRP-TW). . . . . | 18 |



# Índice de tablas



# Índice de fórmulas

|  |    |
|--|----|
| 2.1 Función objetivo TSP: minimizar el costo total del recorrido . . . . .     | 13 |
| 2.2 Restricción de salida única de cada nodo . . . . .                         | 13 |
| 2.3 Restricción de entrada única a cada nodo . . . . .                         | 13 |
| 2.4 Restricción de eliminación de subciclos . . . . .                          | 13 |
| 2.5 Variables binarias que indican si la ruta va de $i$ a $j$ . . . . .        | 14 |
| 2.6 Función objetivo VRP: minimizar el costo total del recorrido . . . . .     | 15 |
| 2.7 Restricción: entrada única a cada cliente . . . . .                        | 16 |
| 2.8 Restricción: salida única desde cada cliente . . . . .                     | 16 |
| 2.9 Restricción: número de vehículos que regresan al depósito . . . . .        | 16 |
| 2.10 Restricción: número de vehículos que salen del depósito . . . . .         | 16 |
| 2.11 Restricción de conectividad: eliminación de subrutas . . . . .            | 16 |
| 2.12 Variables binarias: decisión de ruta entre nodos . . . . .                | 16 |
| 2.13 Función objetivo VRPTW: minimizar el costo total del recorrido . . . . .  | 18 |
| 2.14 Restricción: cada cliente es visitado exactamente una vez . . . . .       | 18 |
| 2.15 Restricción: cada vehículo inicia su ruta en el depósito . . . . .        | 18 |
| 2.16 Restricción: cada vehículo termina su ruta en el depósito . . . . .       | 19 |
| 2.17 Restricción: conservación de flujo para cada vehículo y cliente . . . . . | 19 |
| 2.18 Restricción: respeto a la ventana de tiempo y orden de servicio . . . . . | 19 |
| 2.19 Restricción: inicio del servicio dentro de la ventana de tiempo . . . . . | 19 |
| 2.20 Restricción: capacidad máxima del vehículo . . . . .                      | 19 |
| 2.21 Variables binarias y tiempos no negativos . . . . .                       | 19 |



# **Lista de Abreviaciones**

|              |  |
|--------------|--|
| <b>ACO</b>   | Ant Colony Optimization ( <i>Optimización por Colonias de Hormigas</i> )                                   |
| <b>ADE</b>   | Algoritmo Evolutivo Diferencial ( <i>Differential Evolution Algorithm</i> )                                |
| <b>AG</b>    | Algoritmo Genético ( <i>Genetic Algorithm</i> )  |
| <b>BT</b>    | Búsqueda Tabú ( <i>Tabu Search</i> )   |
| <b>NP</b>    | Nondeterministic Polynomial Time ( <i>Tiempo Polinomial No Determinista</i> )                              |
| <b>P</b>     | Polynomial Time ( <i>Tiempo Polinomial</i> )   |
| <b>SA</b>    | Simulated Annealing ( <i>Recocido Simulado</i> )   |
| <b>TSP</b>   | Travelling Salesman Problem ( <i>Problema del Agente Viajero</i> )   |
| <b>VRP</b>   | Vehicle Routing Problem ( <i>Problema de Ruteo de Vehículos</i> )  |
| <b>VRPTW</b> | Vehicle Routing Problem with Time Windows ( <i>Problema de Ruteo de Vehículos con Ventanas de Tiempo</i> ) |



# Capítulo 1

## Introducción

### 1.1. Planteamiento del Problema

Dentro de la investigación operativa y la logística, modelos clásicos como el **Problema del Agente Viajero** (**TSP**, *Traveling Salesman Problem*) y el **Problema de Ruteo de Vehículos** (**VRP**, *Vehicle Routing Problem*) representan el fundamento para entender y abordar problemáticas logísticas reales. Sin embargo, en entornos modernos donde existen restricciones temporales estrictas para la atención de clientes, se requieren modelos más complejos, como el **Problema de Ruteo de Vehículos con Ventanas de Tiempo** (**VRP-TW**, *Vehicle Routing Problem with Time Windows*).

El **VRP-TW** consiste en planificar rutas para una flota de vehículos, de manera que se minimice el costo total (distancia, tiempo o recursos), asegurando que cada cliente sea atendido dentro de un intervalo de tiempo predefinido, sin violar las capacidades de los vehículos. Este problema pertenece a la clase **NP-difícil** y presenta una alta complejidad combinatoria, lo que hace inviable resolverlo mediante algoritmos exactos cuando se trata de instancias de gran tamaño.

En este contexto, las **metaheurísticas** se presentan como una alternativa viable para encontrar soluciones de buena calidad en tiempos razonables. Entre ellas, la **Optimización por Colonia de Hormigas** (**ACO**, *Ant Colony Optimization*) ha demostrado ser eficaz en la construcción de rutas iniciales, mientras que el **Recocido Simulado** (**SA**, *Simulated Annealing*), apoyado en diversas heurísticas locales, permite mejorar las soluciones generadas. Aun así, ambos enfoques pueden beneficiarse de mecanismos adicionales de exploración y explotación del espacio de búsqueda, por lo que se propone incorporar un **Algoritmo Evolutivo Diferencial** (**ADE**, *Differential Evolution Algorithm*) como componente global de refinamiento.

El problema a resolver, por tanto, se centra en diseñar e implementar un algoritmo híbrido que combine de manera coordinada y efectiva la **ACO**, el **SA** y el **ADE**. En este esquema, el **ADE** se emplea para calibrar los parámetros de los otros métodos, permitiendo así abordar el **VRP-TW** partiendo de modelos más simples como el **TSP** y el **VRP**,

considerando tanto la calidad de la solución como su robustez y eficiencia computacional.

## 1.2. Justificación

El **VRP-TW** es un problema fundamental en la logística moderna, con aplicaciones directas en la optimización de rutas de distribución y transporte, lo que impacta significativamente en la reducción de costos operativos y la mejora del servicio al cliente. Sin embargo, debido a su complejidad combinatoria y a su pertenencia a la clase NP-difícil, las técnicas exactas resultan impracticables para resolver instancias de tamaño realista.

Las soluciones basadas en metaheurísticas han demostrado ser una alternativa eficiente; sin embargo, la mayoría de los enfoques existentes se centran en una sola técnica, lo que limita la exploración y explotación del espacio de soluciones. La integración de métodos híbridos, como la combinación de **ACO**, **SA** y **AED**, puede potenciar las capacidades de búsqueda y refinamiento, mejorando la calidad y robustez de las soluciones obtenidas.

Este trabajo se justifica en la necesidad de desarrollar un método que aporte mayor eficiencia y calidad para resolver el **VRP-TW**, utilizando instancias benchmark clásicas ampliamente reconocidas en la literatura, como las propuestas por Solomon, y así contribuir al avance de las técnicas metaheurísticas aplicadas en problemas logísticos complejos.

## 1.3. Objetivos

### 1.3.1. Objetivo General

Diseñar, implementar, probar y evaluar un algoritmo híbrido que combine la **ACO**, el **SA** con múltiples heurísticas locales y un **ADE**, con el objetivo de resolver de manera eficiente el **VRP-TW**.

### 1.3.2. Objetivos Específicos

1. Estudiar y analizar los problemas **TSP**, **VRP** y **VRP-TW** para identificar sus similitudes estructurales y diferencias, estableciendo así una base conceptual que permita abordar de forma progresiva el problema principal **VRP-TW**.
2. Diseñar un algoritmo híbrido que combine el **ACO** para la construcción de soluciones iniciales y el **SA** para la mejora de dichas soluciones, integrando el **AED** como un componente de autoajuste de parámetros o como mecanismo para generar configuraciones óptimas que guíen el comportamiento de las otras metaheurísticas.

3. Evaluar el rendimiento del algoritmo híbrido en términos de calidad de la solución y eficiencia computacional, utilizando instancias de prueba estándar aceptadas en la literatura, y comparando los resultados con enfoques clásicos y metaheurísticos individuales.

## 1.4. Alcances y Limitaciones

### Alcances

- El trabajo se centra en la resolución de los problemas **TSP**, **VRP** y **VRP-TW**, siendo este último el problema principal de estudio.
- Se desarrolla un enfoque híbrido que integra tres técnicas metaheurísticas: **ACO**, **SA** y un **ADE**, aplicadas de manera complementaria.
- El enfoque es probado con instancias clásicas propuestas por Solomon, ampliamente reconocidas en la literatura, lo que permite una validación objetiva y reproducible de los resultados obtenidos.
- Se analizan métricas relevantes como la calidad de la solución, el tiempo de ejecución y la estabilidad del algoritmo ante múltiples ejecuciones.

### Limitaciones

- El enfoque propuesto está diseñado para instancias estáticas de los problemas de ruteo; no se contempla el tratamiento de versiones dinámicas o en tiempo real.
- La calidad de los resultados depende en gran medida de la correcta calibración de los parámetros de **ACO** y **SA**; sin embargo, esta calibración no se realiza manualmente, sino que es llevada a cabo automáticamente por el **ADE**.
- No se consideran restricciones adicionales como múltiples depósitos, flotas heterogéneas o prioridades diferenciadas entre clientes.
- Las pruebas están limitadas a conjuntos de datos artificiales y benchmarks académicos, sin validación en entornos productivos reales.
- Al tratarse de métodos metaheurísticos, no se garantiza la obtención del óptimo global, sino soluciones cercanas al óptimo dentro de un tiempo razonable.

## 1.5. Estructura del Documento



# Capítulo 2

## Marco Teórico / Estado del Arte

### 2.1. Antecedentes

El estudio de problemas de optimización en rutas se remonta al desarrollo del **Problema del Agente Viajero (TSP)**, formulado a principios del siglo XX, cuyo objetivo es encontrar la ruta de costo mínimo que recorra un conjunto de ciudades exactamente una vez, regresando al punto de partida. A lo largo de las décadas, este problema se ha consolidado como uno de los pilares de la optimización combinatoria [25].

Más adelante, se propuso el **Problema de Ruteo de Vehículos (VRP)** como una generalización del **TSP**, considerando una flota de vehículos que deben atender a múltiples clientes desde un depósito central. Este problema fue introducido por primera vez por Dantzig y Ramser en 1959 para optimizar rutas de distribución de gasolina [6].

Durante la década de 1970, el interés por resolver variantes del **VRP** aumentó significativamente, y surgieron métodos heurísticos como el algoritmo de Clarke y Wright (1964), que se convirtió en un referente para aproximaciones iniciales [4].

Entre la década de 1970 y la de 1980, se desarrollaron versiones especializadas como el *Fleet Routing Problem*, sistemas *Dial-a-Bus* y el diseño de redes de transporte, así como enfoques probabilísticos y la incorporación de incertidumbre [11, 14, 17].

Dentro de estas variantes, el **Problema de Ruteo de Vehículos con Ventanas de Tiempo (VRP-TW)** surgió como una extensión crítica que incorpora restricciones temporales para reflejar mejor las condiciones reales de operación, tales como horarios de entrega y recogida. Este problema fue estudiado ampliamente a partir del trabajo de Solomon en 1987, quien introdujo instancias benchmark y algoritmos heurísticos clásicos [27].

Durante la década de 1990, se desarrollaron enfoques exactos basados en programación entera y técnicas como *branch-and-bound*, *branch-and-cut* y *column generation*, aplicados con éxito a instancias pequeñas [9, 5]. Sin embargo, la elevada complejidad del **VRP-TW** motivó el uso creciente de heurísticas y metaheurísticas, tales como:

- **Algoritmos Genéticos (AG)**, inspirados en la evolución natural, propuestos por Holland en 1975, que utilizan operadores genéticos para evolucionar una población de soluciones [15].
- **Algoritmo de Colonias de Hormigas (ACO)**, propuesto inicialmente por Dorigo en 1992, basado en el comportamiento colectivo de las hormigas para resolver problemas combinatorios [10].
- **Recocido Simulado (SA)**, inspirado en el proceso físico de enfriamiento de metales, introducido por Kirkpatrick, Gelatt y Vecchi en 1983, que permite escapar de óptimos locales mediante la aceptación controlada de soluciones peores [16].
- **Búsqueda Tabú (BT)**, una técnica heurística que explora iterativamente soluciones vecinas para mejorar la calidad de la solución, evitando ciclos mediante la prohibición temporal de movimientos [1].

Estas metaheurísticas han demostrado ser eficaces para obtener soluciones cercanas al óptimo en tiempos razonables [11].

En las últimas dos décadas, la investigación se ha enfocado en algoritmos híbridos, técnicas de aprendizaje automático aplicadas al **VRP**, optimización robusta y modelos dinámicos, buscando mejorar la eficiencia y adaptabilidad de las soluciones en contextos logísticos cada vez más complejos y cambiantes [29].

El presente trabajo se inscribe en esta línea de investigación, proponiendo un modelo basado en un algoritmo híbrido con metaheurísticas para abordar el **VRP-TW**, considerando específicamente restricciones operativas como la capacidad limitada de los vehículos y los intervalos estrictos de tiempo de servicio asignados a cada cliente. Con el objetivo de mejorar la calidad y eficiencia de las rutas generadas, se implementaron tres enfoques metaheurísticos: el Algoritmo Evolutivo Diferencial (**AED**), el Algoritmo de Optimización por Colonias de Hormigas (**ACO**) y el Recocido Simulado (**SA**).

## 2.2. Conceptos Clave

### 2.2.1. El Problema en Optimización

Los **problemas de optimización** constituyen una herramienta fundamental en ciencias computacionales, ingeniería, logística y muchas otras disciplinas. En el contexto de este trabajo de investigación, resultan especialmente relevantes porque permiten modelar situaciones en las que se busca obtener el **mejor resultado posible** bajo ciertas restricciones, como **minimizar costos, distancias o tiempos**.

De acuerdo con [8], los problemas de optimización se dividen de manera natural en dos categorías: problemas con **variables continuas** y problemas con **variables discretas**. A estos últimos se les conoce como problemas de **optimización combinatoria**.

La función  $f : S \rightarrow \mathbb{R}$ , donde el conjunto  $S$  es un subconjunto de  $\mathbb{R}^n$ , se denomina **función objetivo**, **función de costo** o **beneficio**, mientras que el conjunto  $S$  se identifica como el **conjunto factible** o el conjunto de **soluciones posibles** [8].

Como definición formal, se puede decir que el problema general de optimización consiste en encontrar un elemento  $x \in S$  que **optimice** la función objetivo. En el caso de un problema de **minimización**, se expresa de la siguiente manera:

$$\min_{x \in S} f(x).$$

Por su parte, en un problema de **maximización**, se utiliza la notación:

$$\max_{x \in S} f(x).$$

[8]

### 2.2.1.1. Optimización Continua

En la **optimización continua** para problemas de minimización, el objetivo es encontrar un punto  $x_{opt}$  dentro del conjunto factible  $S$  tal que el valor de la función objetivo en ese punto sea menor o igual que el valor en cualquier otro punto del conjunto, es decir,

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

Para problemas de maximización, se busca un punto  $x_{opt}$  en  $S$  donde la función objetivo tome un valor mayor o igual al de cualquier otro punto factible, esto es,

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

A este punto se le denomina **solución óptima global**, y el valor correspondiente  $f(x_{opt})$  se conoce como **costo óptimo**. Además, el conjunto de todas las soluciones óptimas se representa por  $S_{opt}$  [8].

### 2.2.1.2. Optimización Discreta o Combinatoria

Una instancia de un problema de **optimización combinatoria** puede representarse mediante una pareja  $(S, f)$ , donde  $S$  es un conjunto **finito** que contiene todas las soluciones posibles, y  $f$  es una función real, denominada **función objetivo** o **función costo**, definida como

$$f : S \rightarrow \mathbb{R}.$$

Para problemas de minimización, se busca una solución  $i_{opt} \in S$  tal que

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

mientras que para problemas de maximización, se requiere que

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

La solución  $i_{opt}$  se denomina **solución óptima global**, y el valor  $f(i_{opt})$  representa el **costo óptimo**. El conjunto de todas las soluciones óptimas se denota como  $S_{opt}$ . Un problema de optimización combinatoria se define entonces como el conjunto de todas sus instancias [8].

Comprender esta distinción es esencial para abordar el problema central de este trabajo **VRP-TW**, el cual se enmarca dentro de la optimización combinatoria. En este tipo de problemas, el número de soluciones posibles es **finito**, y el desafío consiste en identificar cuál de ellas representa la **mejor opción** según un criterio específico.

Esta base teórica permite introducir los problemas clásicos de enrutamiento, como el **TSP**, el **VRP** y sus variantes, como el **VRP-TW**, los cuales serán analizados más adelante por su relación directa con la planificación eficiente de rutas en contextos reales.

### 2.2.2. Complejidad Computacional y Algorítmica

El estudio de la **complejidad** en ciencias computacionales busca comprender qué tan difícil es resolver un problema, tanto desde su estructura teórica como desde los recursos requeridos para su solución. Esta dificultad se aborda desde dos enfoques: la **complejidad computacional**, que clasifica los problemas según el crecimiento de recursos necesarios en función del tamaño de la entrada, y la **complejidad algorítmica**, que evalúa el desempeño de algoritmos específicos al aplicarse sobre dichas entradas [13, 25].

De forma simplificada, la complejidad computacional se refiere al análisis general del problema sin importar el algoritmo, mientras que la algorítmica se enfoca en el tiempo y espacio consumido por un algoritmo concreto [21].

Esta distinción es clave al abordar problemas de optimización combinatoria como el **TSP**, el **VRP** y su variante principal en este trabajo: el **VRP-TW**, que presentan una explosión combinatoria (crecimiento factorial) de soluciones conforme crece el número de clientes.

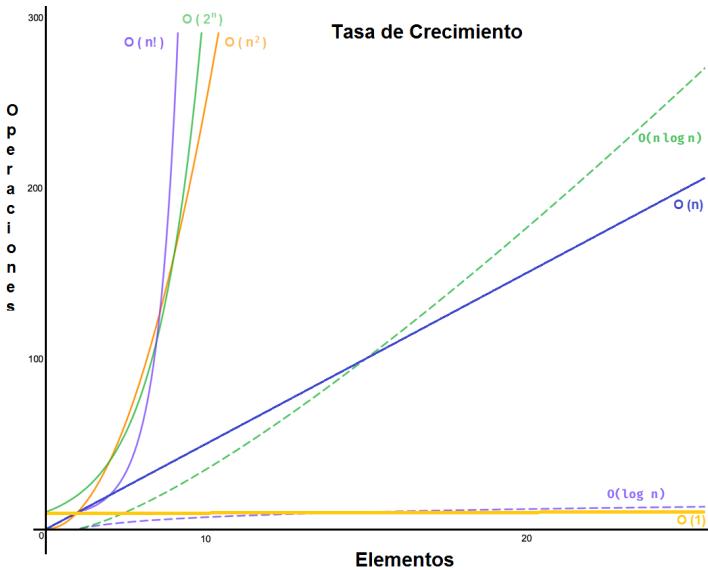


Figura 2.1: Tasa de crecimiento de varias funciones en análisis de algoritmos [23].

En la Figura 2.1 se observa cómo las funciones constantes  $\mathcal{O}(1)$  y logarítmicas  $\mathcal{O}(\log n)$  crecen lentamente, incluso para entradas de gran tamaño. Las funciones polinomiales, como  $\mathcal{O}(n)$  y  $\mathcal{O}(n \log n)$ , presentan un crecimiento más acelerado pero aún manejable. En contraste, las funciones exponenciales  $\mathcal{O}(2^n)$  y factoriales  $\mathcal{O}(n!)$  crecen de manera extremadamente rápida, lo que vuelve intratables los problemas asociados a medida que aumenta el tamaño de la entrada. Por esta razón, problemas como el **TSP**, **VRP** y **VRP-TW** se consideran de alta complejidad computacional, ya que su espacio de búsqueda crece factorialmente.

### 2.2.3. El Problema P vs NP

En ciencias de la computación, todos los problemas pueden clasificarse en dos grupos principales: **problemas decidibles** y **problemas indecidibles**. Un problema es *indecidible* cuando no existe ningún algoritmo que pueda resolverlo, incluso si se dispone de tiempo y recursos ilimitados. En consecuencia, no es posible determinar si dicho problema terminará su ejecución. Por contraste, un problema es *decidible* cuando existe (o podría existir) un algoritmo capaz de resolverlo [21].

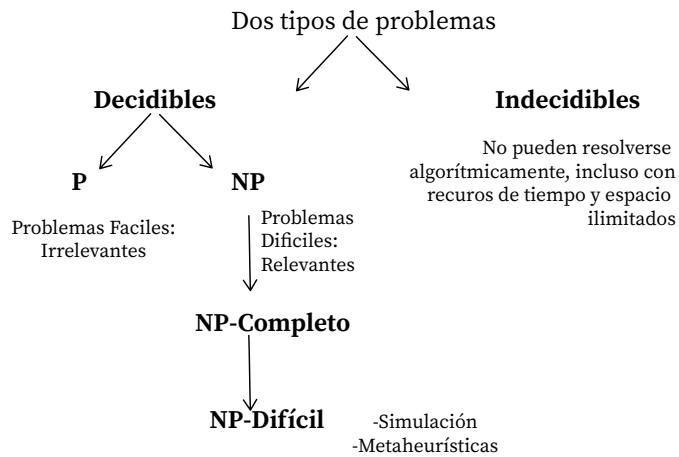


Figura 2.2: Clasificación de los problemas computacionales en decidibles e indecidibles [21].

En este trabajo, los problemas abordados —el **TSP**, el **VRP** y el **VRP-TW**— son considerados **problemas decídibles**. A continuación, se presenta una clasificación más detallada dentro de esta categoría, basada en la teoría de la complejidad computacional.

### 2.2.3.1. Clases P y NP

Dentro de los problemas decídibles, las clases **P** y **NP** juegan un papel fundamental [21].

#### Clase P

Un problema pertenece a la clase **P** si puede resolverse mediante un algoritmo cuyo tiempo de ejecución está acotado por una función polinomial respecto al tamaño de la entrada. Formalmente, existe un polinomio  $p(n)$  y constantes  $\alpha, n_0 > 0$  tales que:

$$D(n) \leq \alpha \cdot p(n) \quad \forall n \geq n_0,$$

lo que se expresa en notación asintótica como

$$D = O(n^k).$$

Aunque se considera que algoritmos con tiempo polinomial son “eficientes”, en la práctica el grado del polinomio importa, ya que un polinomio de grado muy alto puede ser inviable. Un ejemplo clásico de problema en **P** es la multiplicación de números naturales [12].

## Clase NP

Un problema pertenece a la clase ***NP*** si, dado un candidato a solución (certificado), es posible verificar su validez en tiempo polinomial. Es importante notar que esto no implica necesariamente que exista un algoritmo eficiente para encontrar la solución, sino que su validación es eficiente. Un ejemplo típico es el **TSP**: si se proporciona una ruta candidata, es fácil comprobar en tiempo polinomial la longitud total de la ruta para verificar si cumple cierta condición [12].

### La gran incógnita: $P = NP?$

Una de las preguntas abiertas más importantes en ciencias de la computación es si  $P = NP$  o  $P \neq NP$ , es decir, si todo problema cuya solución puede verificarse rápidamente también puede resolverse rápidamente. Esta cuestión tiene profundas implicaciones teóricas y prácticas [21].

#### 2.2.3.2. NP-Completo y NP-Difícil

Dentro de la clase ***NP*** existe una subclase de problemas llamados **NP-completos**, que son los problemas más difíciles en ***NP***. Si se encontrara un algoritmo eficiente para cualquiera de ellos, entonces todos los problemas en ***NP*** podrían resolverse eficientemente [21].

Por otro lado, los problemas **NP-difíciles** pueden ser incluso más complejos que los **NP-completos**, ya que no requieren pertenecer a ***NP*** (es decir, su solución no tiene por qué ser verificable en tiempo polinomial), pero son al menos tan difíciles como los problemas **NP-completos**. Pueden incluir problemas de decisión, búsqueda u optimización [21].

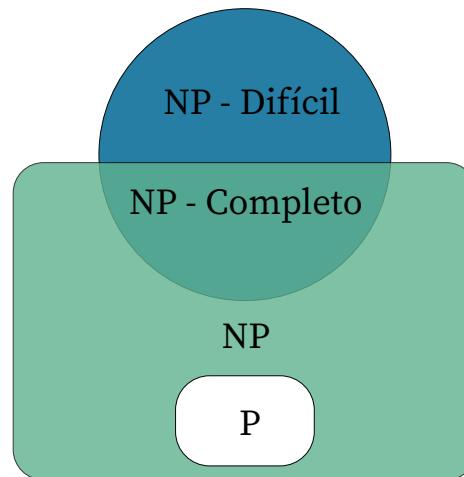


Figura 2.3: Relación entre las clases de complejidad: P, NP, NP-completo y NP-difícil.

El crecimiento exponencial del tiempo requerido para resolver problemas como el **TSP**, el **VRP** y el **VRP-TW** los ubica dentro de la clase **NP**. Estos problemas, aunque decidibles, presentan una complejidad tan elevada que no se conoce ningún algoritmo exacto que los resuelva eficientemente para instancias grandes. Por ello, más adelante se analizará su clasificación específica dentro de **NP**, así como las estrategias metaheurísticas empleadas para aproximar soluciones en tiempos razonables.

## 2.2.4. Problemas de Optimización Combinatoria

De manera complementaria a la definición formal presentada en la sección 2.2.1.2, Papadimitriou y Steiglitz [26] definen la **optimización combinatoria** como el estudio de problemas de optimización en los que el conjunto de soluciones factibles es discreto, o puede discretizarse mediante un proceso de enumeración, y donde se busca una solución que optimice una función objetivo definida sobre dicho conjunto.

Los problemas de optimización combinatoria constituyen una clase amplia de modelos aplicables a diversas áreas, especialmente en la planificación y el enrutamiento de vehículos. Dentro de este campo destacan el **TSP**, que representa la forma más básica de optimización de rutas, y sus extensiones, que incorporan restricciones adicionales propias de escenarios reales, como la capacidad limitada de los vehículos o las ventanas de tiempo en las que deben realizarse las entregas.

En este contexto, el **VRP** y el **VRP-TW** se han consolidado como dos de los desafíos más estudiados, debido a su relevancia práctica y su elevada complejidad computacional. A continuación, se describen estos problemas en detalle.

### 2.2.4.1. Problema del Agente Viajero (TSP)

El **Problema del Agente Viajero**, conocido por sus siglas en inglés como **TSP** (*Travelling Salesman Problem*), es uno de los problemas más reconocidos y complejos dentro de las ciencias computacionales. Ha sido estudiado desde diversas ramas de la ingeniería y por múltiples motivos. Su aplicación principal consiste en determinar rutas desde diferentes enfoques, ya sea en procesos que requieren una secuencia específica o en operaciones logísticas relacionadas con el transporte, con el objetivo de encontrar la ruta óptima considerando criterios de minimización de distancia o de costo [19].

En la Figura 2.4 se presenta una representación gráfica de una posible solución al **TSP**. En ella, un vehículo parte de un nodo inicial(depósito), recorre una serie de ciudades o puntos de entrega siguiendo una secuencia determinada, y regresa al punto de partida, completando así un circuito cerrado que minimiza la distancia total recorrida.

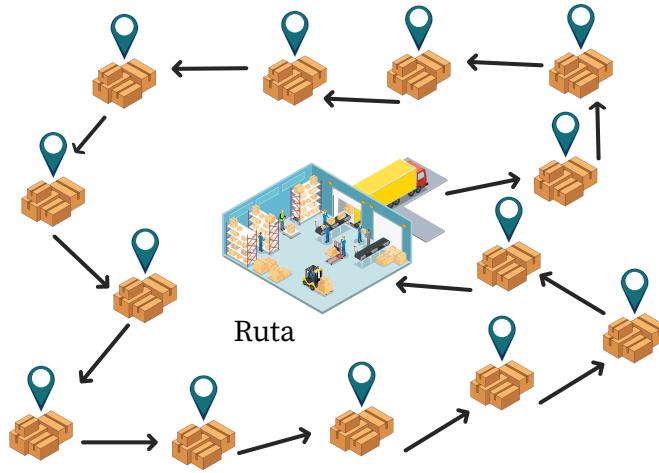


Figura 2.4: Ejemplificación de una solución al Problema del Agente Viajero (TSP).

Según [28], el **TSP** se define sobre un grafo  $G = [N, A, C]$ , donde  $N$  es el conjunto de nodos,  $A$  es el conjunto de arcos y  $C = [c_{ij}]$  es la matriz de costos (distancias) entre nodos  $i$  y  $j$ . El objetivo es encontrar un ciclo hamiltoniano de costo mínimo que recorra todos los nodos una sola vez y regrese al punto de partida.

### Modelo Matemático del Problema del Agente Viajero (TSP)

$$\min \sum_{i=1}^N \sum_{j=1}^N c_{ij} x_{ij} \quad (2.1)$$

sujeto a:

$$\sum_{j=1}^N x_{ij} = 1, \quad \forall i = 1, \dots, N \quad (2.2)$$

$$\sum_{i=1}^N x_{ij} = 1, \quad \forall j = 1, \dots, N \quad (2.3)$$

$$u_i - u_j + Nx_{ij} \leq N - 1, \quad \forall i, j = 2, \dots, N, \quad i \neq j \quad (2.4)$$

$$x_{ij} \in \{0, 1\}, \quad \forall i, j = 1, \dots, N \quad (2.5)$$

donde:

- $x_{ij} = 1$  si la ruta va de  $i$  a  $j$ , y 0 en caso contrario;
- $u_i$  son variables auxiliares usadas para eliminar subciclos, con  $i = 2, \dots, N$  [28].

La **función objetivo** (2.1) representa la suma total de los costos (o distancias) de los arcos que conforman el recorrido. La **restricción** en (2.2) garantizan que, al salir de cada nodo, sólo se dirija a un único nodo siguiente. La **restricción** en (2.3) aseguran que a cada nodo únicamente se llegue una vez. La **restricción** (2.4) es fundamental, ya que previene la formación de subciclos que no incluyan a todos los nodos, asegurando así un recorrido único y completo. Finalmente, la **restricción** (2.5) indica que las variables  $x_{ij}$  son binarias, es decir, sólo pueden tomar los valores 0 o 1, lo que permite representar la inclusión o exclusión de una ruta entre nodos [28].

### Complejidad del TSP

Como explica [26], el **TSP** es *NP-difícil*, y su versión de decisión es *NP-completa*. Esto implica que la búsqueda de un algoritmo eficiente para resolverlo representa un desafío fundamental en la teoría de la computación, pues encontrar una solución polinomial para el **TSP** permitiría resolver eficientemente toda la clase *NP*. Esta dificultad justifica la utilización de algoritmos heurísticos y metaheurísticos, dado que las técnicas exactas sólo resultan prácticas para instancias

El **TSP** ha sido tradicionalmente considerado como el punto de partida para el estudio de problemas más complejos de enrutamiento, como el **VRP** y sus variantes, entre ellas el **VRP-TW**. Su análisis proporciona una base conceptual sólida para comprender aspectos fundamentales como la construcción de rutas, la minimización de costos y el manejo de restricciones operativas. Estudiar el **TSP** permite introducir de forma gradual nuevas variables y condiciones, lo que lo convierte en un modelo introductorio ideal para el desarrollo de enfoques logísticos más realistas.

#### 2.2.4.2. Problema del Problema Ruteo de Vehículos (VRP)

El **Problema del Ruteo de Vehículos**, conocido por sus siglas en inglés como **VRP** (*Vehicle Routing Problem*), puede interpretarse como la convergencia de dos problemas clásicos de optimización combinatoria: el **Problema del Agente Viajero (TSP)** mencionado anteriormente en la sección 2.2.4.1, en el que se asume una capacidad infinita de los vehículos, y el **Problema de Empaqueamiento en Compartimentos (BPP)** [7]. En su formulación más básica, el **VRP** considera un depósito central desde donde parte una flota de vehículos para atender a un conjunto de clientes distribuidos geográficamente. Cada vehículo debe visitar un subconjunto de clientes exactamente una

vez, respetando su capacidad de carga y satisfaciendo la demanda de los clientes asignados. El objetivo es minimizar el costo total asociado a las rutas, las cuales inician y finalizan en el depósito [24].

La Figura 2.5 muestra una representación gráfica de una solución al **VRP**. En este ejemplo, varios vehículos parten desde un depósito central para atender a distintos clientes distribuidos geográficamente. Cada ruta ha sido asignada de manera que se respeten las restricciones de capacidad de carga de los vehículos, y se minimice la distancia total recorrida, cumpliendo así con los objetivos del problema.

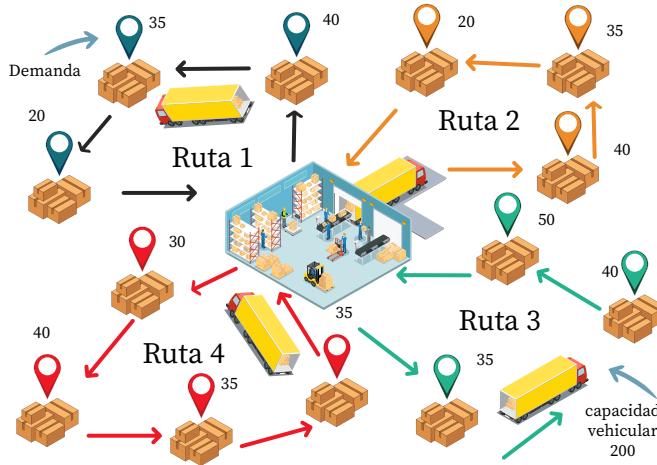


Figura 2.5: Ejemplificación de una solución al Problema de Ruteo de Vehículos (VRP).

Según [29], el **VRP** se define sobre un grafo  $G = (V, A)$ , donde  $V = \{0, 1, \dots, n\}$  representa el conjunto de nodos, con el nodo 0 correspondiente al depósito y los nodos restantes a los clientes,  $A$  es el conjunto de arcos, y  $C = [c_{ij}]$  es la matriz de costos asociados a los arcos  $(i, j)$ . El objetivo es determinar un conjunto de rutas de costo mínimo para una flota de vehículos idénticos, de modo que: cada cliente sea visitado exactamente una vez por un solo vehículo, cada ruta comience y termine en el depósito, y la demanda total de los clientes en cada ruta no exceda la capacidad del vehículo.

### Modelo Matemático del Problema de Ruteo de Vehículos (VRP)

$$\min \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij} \quad (2.6)$$

sujeto a:

$$\sum_{i \in V} x_{ij} = 1, \quad \forall j \in V \setminus \{0\} \quad (2.7)$$

$$\sum_{j \in V} x_{ij} = 1, \quad \forall i \in V \setminus \{0\} \quad (2.8)$$

$$\sum_{i \in V} x_{i0} = K \quad (2.9)$$

$$\sum_{j \in V} x_{0j} = K \quad (2.10)$$

$$\sum_{i \in S} \sum_{j \in S} x_{ij} \geq r(S), \quad \forall S \subseteq V \setminus \{0\}, S \neq \emptyset \quad (2.11)$$

$$x_{ij} \in \{0, 1\}, \quad \forall i, j \in V \quad (2.12)$$

donde:

- $V$  es el conjunto de nodos (clientes más depósito), con el depósito representado por el nodo 0;
- $c_{ij}$  es el costo (o distancia) de viajar del nodo  $i$  al nodo  $j$ ;
- $x_{ij} = 1$  si se viaja directamente del nodo  $i$  al nodo  $j$ , 0 en otro caso;
- $K$  es el número de vehículos disponibles en el depósito;
- $S$  es un subconjunto de clientes, y  $r(S)$  representa el número mínimo de vehículos necesarios para satisfacer la demanda total de los clientes en  $S$  [29].

La **función objetivo** (2.6) minimiza el costo total de todas las rutas. Las **restricciones** (2.7) y (2.8) aseguran que cada cliente sea visitado exactamente una vez. Las **restricciones** (2.9) y (2.10) controlan el número de vehículos que entran y salen del depósito. La **restricción** (2.11) evita la formación de subrutas que no incluyan al depósito, garantizando conectividad. Finalmente, la **restricción** (2.12) indica que las variables de decisión son binarias [29].

## Complejidad del VRP

Como señalan Laporte y Nobert [18], el **VRP** es *NP-difícil*, incluso en su versión más simple con un solo vehículo (es decir, el **TSP**). Esto significa que no se conoce ningún algoritmo que lo resuelva de forma exacta en tiempo polinomial, y su complejidad crece exponencialmente con el número de clientes. Por ello, la mayoría de las instancias prácticas requieren el uso de técnicas heurísticas o metaheurísticas, ya que los métodos exactos son computacionalmente inviables para problemas de tamaño medio o grande.

El **VRP** representa la extensión natural del **TSP** hacia escenarios logísticos más realistas y complejos, incorporando restricciones operativas que reflejan las limitaciones del mundo real. Su estudio es fundamental para comprender cómo las restricciones de capacidad, múltiples vehículos y la gestión de flotas impactan en la optimización de rutas. El **VRP** sirve como base conceptual para abordar problemas logísticos avanzados como la distribución con ventanas de tiempo, el ruteo con múltiples depósitos y la optimización de última milla en el comercio electrónico. Analizar el **VRP** permite desarrollar una comprensión profunda de los trade-offs entre minimización de costos, utilización eficiente de recursos y satisfacción de restricciones operativas, estableciendo los fundamentos teóricos necesarios para abordar las complejidades de los sistemas de distribución modernos.

### 2.2.4.3. Problema de Ruteo de Vehículos con Ventanas de Tiempo (VRP-TW)

El **Problema de Ruteo de Vehículos con Ventanas de Tiempo**, conocido por sus siglas en inglés como **VRP-TW** (*Vehicle Routing Problem with Time Windows*), es una extensión del clásico **VRP** mencionado anteriormente en la sección 2.2.4.2 que incorpora restricciones temporales. En este problema, además de satisfacer las demandas específicas de cada cliente, es imprescindible que las visitas se realicen dentro de una franja horaria determinada para cada uno de ellos [29]. Específicamente, el servicio a un cliente debe comenzar en un instante mayor o igual al inicio de su ventana de tiempo, y el arribo al punto de servicio no puede exceder el límite superior de dicha ventana. Asimismo, en caso de que un vehículo llegue antes del inicio de la ventana, deberá esperar hasta el momento indicado para poder atender al cliente [24].

En este contexto, el objetivo fundamental del **VRP-TW** es determinar un conjunto de rutas para una flota de vehículos homogéneos que partan y regresen al depósito, visiten cada cliente exactamente una vez dentro de su respectiva ventana temporal, y sin exceder la capacidad de los vehículos. Todo esto debe lograrse minimizando el costo total del recorrido, usualmente expresado en términos de distancia o tiempo [27].

En la Figura 2.6 se muestra una solución típica al **VRP-TW**. En este caso, las rutas de múltiples vehículos parten desde un único depósito, deben cumplir con las capacidades máximas y además respetar las ventanas temporales de atención en cada cliente, optimizando el recorrido total.

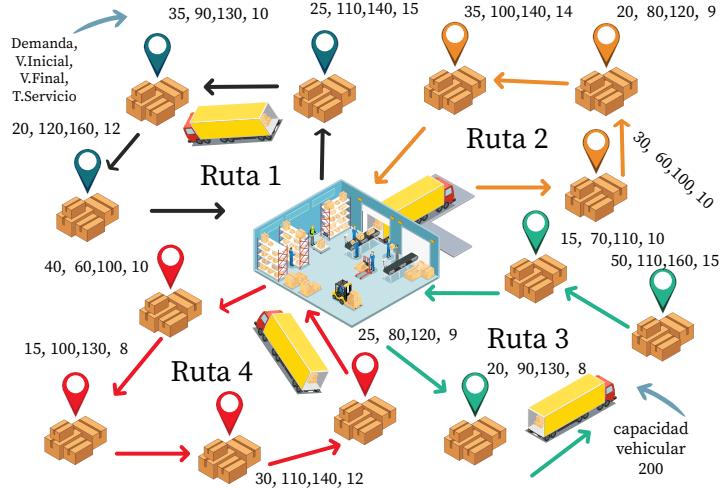


Figura 2.6: Ejemplificación de una solución al Problema de Ruteo de Vehículos con Ventanas de Tiempo (VRP-TW).

Según [29], el **VRP-TW** se define sobre un grafo dirigido  $G = (V, A)$ , donde  $V = \{0, 1, \dots, n, n + 1\}$  representa el conjunto de nodos, siendo 0 y  $n + 1$  el depósito (nodos origen y destino, respectivamente), y  $N = V \setminus \{0, n + 1\}$  el conjunto de clientes. Cada arco  $(i, j) \in A$  tiene asociado un costo  $c_{ij}$  y un tiempo de viaje  $t_{ij}$ . Además, cada cliente  $i$  tiene una demanda  $d_i$ , un tiempo de servicio  $s_i$ , y una ventana de tiempo  $[a_i, b_i]$  dentro de la cual debe iniciarse su servicio.

### Modelo Matemático del Problema de Ruteo de Vehículos con Ventanas de Tiempo(VRP-TW)

$$\min \sum_{k \in K} \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ijk} \quad (2.13)$$

sujeto a:

$$\sum_{k \in K} \sum_{j \in V} x_{ijk} = 1, \quad \forall i \in N \quad (2.14)$$

$$\sum_{j \in V} x_{0jk} = 1, \quad \forall k \in K \quad (2.15)$$

$$\sum_{i \in V} x_{i,n+1,k} = 1, \quad \forall k \in K \quad (2.16)$$

$$\sum_{j \in V} x_{ijk} - \sum_{j \in V} x_{jik} = 0, \quad \forall k \in K, \forall i \in N \quad (2.17)$$

$$w_{jk} \geq w_{ik} + s_i + t_{ij} - M(1 - x_{ijk}), \quad \forall k \in K, \forall i, j \in V \quad (2.18)$$

$$a_i \leq w_{ik} \leq b_i, \quad \forall k \in K, \forall i \in V \quad (2.19)$$

$$\sum_{i \in V} \sum_{j \in V} d_i x_{ijk} \leq C, \quad \forall k \in K \quad (2.20)$$

$$x_{ijk} \in \{0, 1\}, \quad w_{ik} \geq 0, \quad \forall k \in K, \forall i, j \in V \quad (2.21)$$

donde:

- $V = \{0, 1, \dots, n, n + 1\}$  es el conjunto de nodos, donde 0 y  $n + 1$  representan el depósito;
- $N = V \setminus \{0, n + 1\}$  es el conjunto de clientes;
- $K$  es el conjunto de vehículos disponibles;
- $A$  es el conjunto de arcos permitidos entre nodos;
- $c_{ij}$  es el costo o distancia de viajar del nodo  $i$  al nodo  $j$ ;
- $x_{ijk}$  es variable binaria que indica si el vehículo  $k$  viaja directamente del nodo  $i$  al nodo  $j$ ;
- $w_{ik}$  es el tiempo de inicio del servicio del vehículo  $k$  en el nodo  $i$ ;
- $s_i$  es el tiempo de servicio requerido en el nodo  $i$ ;
- $t_{ij}$  es el tiempo de viaje entre los nodos  $i$  y  $j$ ;
- $[a_i, b_i]$  es la ventana de tiempo para el nodo  $i$ ;
- $d_i$  es la demanda del cliente  $i$ ;

- $C$  es la capacidad máxima de cada vehículo;
- $M$  es un número grande usado para activar o desactivar restricciones condicionales.

La **función objetivo** (2.13) minimiza el costo total de todas las rutas. Las **restricciones** (2.14) garantizan que cada cliente sea visitado exactamente una vez. Las **restricciones** (2.15) y (2.16) aseguran que cada vehículo comience y termine su ruta en el depósito. La **restricción** (2.17) mantiene el balance de flujo para cada vehículo y cliente. La **restricción** (2.18) impone el orden y respeto a las ventanas temporales mediante los tiempos de servicio y de viaje, con ayuda del parámetro  $M$ . La **restricción** (2.19) asegura que el servicio inicie dentro de la ventana asignada. La **restricción** (2.20) limita la capacidad máxima que puede transportar cada vehículo. Finalmente, la **restricción** (2.21) define los dominios de las variables de decisión [29].

### Complejidad del VRP-TW

Sabemos que la complejidad del **VRP** es *NP-difícil*, como se señala en la Subsubsección 2.2.4.2 y en [18]. El **VRP-TW** es una generalización del **VRP** clásico, y también pertenece a la clase de problemas *NP-difícil*. De hecho, incluso si se ignoran las ventanas de tiempo, el problema sigue siendo difícil de resolver. Al incorporar restricciones temporales, el **VRP-TW** se vuelve aún más complejo, tanto desde el punto de vista computacional como algorítmico. Según Bräysy y Gendreau [3] y Toth y Vigo [29], las ventanas de tiempo reducen significativamente el conjunto de soluciones viables, aumentando así la dificultad del problema.

El **VRP-TW** representa una evolución del **VRP** orientada a entornos donde las restricciones temporales son determinantes para la calidad del servicio. Esta variante refleja de forma más precisa los desafíos logísticos actuales, donde los clientes no solo deben ser visitados, sino también atendidos dentro de intervalos de tiempo específicos, lo cual impone una coordinación precisa entre la planificación de rutas, la gestión de recursos y el cumplimiento de niveles de servicio.

### 2.2.5. Métodos de Solución para el VRPTW

Los problemas de enrutamiento como el **TSP**, el **VRP** y su extensión con ventanas de tiempo, el **VRP-TW**, mencionados en secciones anteriores, pertenecen a la clase de problemas *NP-difíciles* [26, 18, 29]. Esta clasificación se debe al crecimiento exponencial del tiempo requerido para resolverlos a medida que aumenta el número de nodos o clientes en la instancia.

Dada su complejidad computacional, estos problemas han sido abordados mediante diversas estrategias de solución que tienen como objetivo minimizar el costo total del recorrido, respetando las restricciones impuestas. Entre estas estrategias se encuentran los *métodos exactos* y las *técnicas heurísticas y metaheurísticas*.

A continuación, se describen estos enfoques, junto con ejemplos representativos aplicados en los problemas mencionados.

#### 2.2.5.1. Métodos Exactos

Son aquellos que parten de una formulación matemática del problema, generalmente como modelos de programación lineal entera o similares, y alcanzan soluciones factibles mediante algoritmos que acotan el conjunto de soluciones posibles [20]. Estos métodos garantizan la obtención de una solución óptima; sin embargo, su principal limitación radica en el alto costo computacional que implican, por lo que su eficiencia se restringe a instancias de tamaño reducido, de hasta aproximadamente 50 clientes en problemas de enrutamiento[24].

Con base en la clasificación presentada por [24], los métodos exactos pueden agruparse en las siguientes categorías:

1. **Técnicas de relajación:** consisten en simplificar el modelo relajando ciertas restricciones complejas, lo que permite obtener cotas inferiores o superiores que guían la búsqueda de soluciones.
2. **Búsqueda directa en árbol:** algoritmos como *Branch and Bound* y *Branch and Cut* exploran el espacio de soluciones mediante estructuras jerárquicas, descartando ramas no prometedoras.
3. **Programación dinámica:** resuelve el problema descomponiéndolo en subproblemas más pequeños que se abordan de forma recursiva y eficiente.
4. **Programación lineal entera:** utiliza modelos enteros mixtos como formulación clásica para problemas como el VRPTW, y sirve frecuentemente de base para otros enfoques híbridos.

Dado que el presente trabajo se enfoca en técnicas *heurísticas y metaheurísticas*, no se abordará en detalle la implementación de estos métodos exactos. Para una revisión más amplia, se remite al lector a [24].

#### 2.2.5.2. Heurísticas

Heurística es un concepto cuyo origen se remonta a la Grecia clásica, derivado de la palabra griega *heuriskein*, que significa encontrar o descubrir. Según la historia, se asocia con *eureka*, la famosa exclamación atribuida a Arquímedes [2].

Para exemplificar este concepto, se expone la siguiente definición:

Según Zanakis et al. (citado en [22]), las heurísticas son “*procedimientos simples, a menudo basados en el sentido común, que se supone que obtendrán una*

*buenas soluciones (no necesariamente óptimas) a problemas difíciles de un modo sencillo y rápido”.*

Resumiendo la subsubsection 2.2.3.1, los problemas de decisión que pertenecen a la clase NP corresponden a aquellos para los que no se puede garantizar encontrar una mejor solución en un tiempo polinómico razonable.

En este contexto, los métodos heurísticos se convierten en herramientas valiosas para abordar problemas de optimización complejos, como el **TSP**, el **VRP** y su variante con ventanas de tiempo, el **VRP-TW**. Estos problemas, por su naturaleza combinatoria y su complejidad computacional, son difíciles de resolver mediante métodos exactos en tiempos razonables cuando el tamaño de la instancia crece.

Así, las heurísticas permiten encontrar soluciones satisfactorias de forma eficiente, priorizando tanto la calidad del resultado como la rapidez con que se obtiene. Aunque no garantizan optimalidad, su uso es especialmente útil en escenarios donde las soluciones exactas son computacionalmente inviables.

### 2.2.5.3. Metahurísticas

El término *metaheurística* fue introducido por Fred Glover en 1986 [2]. Etimológicamente, deriva de la composición de dos palabras de origen griego, que son “meta” y “heurística”. El segundo término ha sido descrito en la sección anterior, mientras que el prefijo meta puede traducirse como “más allá de” o “en un nivel superior” [22].

Con este término, Glover pretendía definir un *procedimiento maestro de alto nivel que guía y modifica otras heurísticas para explorar soluciones más allá de la simple optimalidad local* [22].

Para ilustrar este concepto, cabe mencionar la definición propuesta por J.P. Kelly et al., quien es citado por [22]:

*Las metaheurísticas son métodos aproximados especialmente diseñados para abordar problemas complejos de optimización combinatoria donde los heurísticos tradicionales resultan ineficaces. Estas técnicas ofrecen un marco flexible para desarrollar algoritmos híbridos que integran conceptos provenientes de la inteligencia artificial, la evolución biológica y procesos estadísticos.*

Para los problemas **TSP**, **VRP** y **VRP-TW** se han implementado diversas metaheurísticas. Según [2], estas son estrategias de búsqueda diseñadas para mejorar procedimientos heurísticos, y se clasifican de acuerdo con la fuente de inspiración que las fundamenta. A continuación, se describen brevemente tres categorías relevantes:

## Metaheurísticas inspiradas en la física

Este tipo de metaheurísticas toma como referencia principios y fenómenos de la física para guiar el proceso de búsqueda de soluciones óptimas. Un ejemplo representativo, es el **Recocido Simulado** (*Simulated Annealing, SA*) [2].

### Recocido Simulado

El recocido simulado (*Simulated Annealing, SA*) es una metaheurística inspirada en el proceso físico de recocido de los metales, donde se calienta un sólido a altas temperaturas y luego se enfria de manera controlada, permitiendo que sus partículas se reorganicen y alcancen una estructura de mínima energía.

De forma análoga, en el contexto de la optimización combinatoria, cada solución posible del problema se considera un “estado” del sistema, y su calidad se mide como si fuera una “energía”. El objetivo es encontrar el estado (solución) con la menor energía (mejor valor de la función objetivo).

El algoritmo comienza con una solución inicial y, en cada iteración, genera una nueva solución vecina. Si la nueva solución es mejor, se acepta; si es peor, puede ser aceptada con una cierta probabilidad que depende de un parámetro llamado “temperatura”. A medida que el algoritmo avanza, la temperatura disminuye, reduciendo así la probabilidad de aceptar soluciones peores. Este mecanismo permite escapar de óptimos locales y explorar mejor el espacio de soluciones [8].

## Metaheurísticas inspiradas en la evolución: Algoritmos Genéticos (Genetic Algorithms, GA)

Los Algoritmos Genéticos, propuestos por John Holland en los años 60, están inspirados en los procesos de evolución natural y selección biológica. Estas técnicas trabajan con una población de soluciones codificadas genéticamente, que evolucionan mediante operadores como la selección, el cruce (recombinación) y la mutación. La idea principal es que las soluciones más aptas tienen mayor probabilidad de reproducirse y transmitir sus características a las siguientes generaciones, guiando así la búsqueda hacia soluciones cada vez más óptimas.

## Metaheurísticas inspiradas en la biología: Optimización basada en Colonias de Hormigas (Ant Colony Optimization, ACO)

La Optimización basada en Colonias de Hormigas es una metaheurística bioinspirada que simula el comportamiento colectivo de las hormigas reales al buscar rutas eficientes hacia fuentes de alimento. Estas se comunican de forma indirecta mediante el depósito de feromonas sobre los caminos recorridos. En su versión computacional, cada agente

(hormiga) construye soluciones mediante un proceso probabilístico que se ve influenciado por la cantidad de feromona acumulada y por una heurística local (por ejemplo, la cercanía entre nodos). Las rutas exitosas refuerzan la feromona, favoreciendo su reutilización en iteraciones posteriores.

#### **2.2.5.4. Algoritmos Híbridos**

### **2.3. Referencias Relevantes**

# Capítulo 3

## Materiales y Métodos

### 3.1. Materiales

### 3.2. Métodos



# **Capítulo 4**

## **Resultados**



# **Capítulo 5**

## **Conclusiones**



# Bibliografía

- [1] Emile Aarts and Jan K Lenstra. *Local Search in Combinatorial Optimization*. John Wiley & Sons, 1988.
- [2] Orlando Antonio Suárez. Una aproximación a la heurística y metaheurísticas. *INGE@UAN – Tendencias en la Ingeniería*, 1(2), 2014.
- [3] Olli Bräysy and Michel Gendreau. Vehicle routing problem with time windows, part i: Route construction and local search algorithms. *Transportation Science*, 39(1):104–118, 2005.
- [4] G Clarke and J W Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Operations research*, 12(4):568–581, 1964.
- [5] Jean-François Cordeau and Gilbert Laporte. A unified tabu search heuristic for vehicle routing problems with time windows. *Journal of the operational research society*, 53(5):528–536, 2002.
- [6] George B Dantzig and John H Ramser. The truck dispatching problem. *Management science*, 6(1):80–91, 1959.
- [7] Julio Mario Daza, Jairo R. Montoya, and Francesco Narducci. Resolución del problema de enrutamiento de vehículos con limitaciones de capacidad utilizando un procedimiento metaheurístico de dos fases. *Revista EIA*, (12):23–38, diciembre 2009.
- [8] 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.
- [9] Martin Desrochers, Jacques Desrosiers, and Marius M Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Operations research*, 40(2):342–354, 1992.
- [10] Marco Dorigo. *Optimization, Learning and Natural Algorithms*. Phd thesis, Politecnico di Milano, Milán, Italia, 1992.
- [11] Burak Eksioglu, Arif Volkan Vural, and Arnold Reisman. The vehicle routing problem: A taxonomic review. *Computers & Industrial Engineering*, 57(4):1472–1483, 2009.

- [12] Xavier Alejandro Flores Cabezas. Problemas del milenio: P vs np. *Revista de Divulgación Amarun*, 1:1–15, 2014.
- [13] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.
- [14] Bruce L. Golden and Arjang A. Assad. Vehicle routing: Methods and studies. In *Studies in Management Science and Systems*. North-Holland, 1988.
- [15] John H Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.
- [16] Scott Kirkpatrick, C. Daniel Gelatt Jr., and Mario P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.
- [17] Gilbert Laporte. The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59(3):345–358, 1992.
- [18] Gilbert Laporte and Yves Nobert. Exact algorithms for the vehicle routing problem. *Annals of Discrete Mathematics*, 31:147–184, 1987.
- [19] Erasmo López, Óscar Salas, and Álex Murillo. El problema del agente viajero: Un algoritmo determinístico usando búsqueda tabú. *Revista de Matemática: Teoría y Aplicaciones*, 21(1):127–144, 2014.
- [20] Armin Lüer, Magdalena Benavente, Jaime Bustos, and Bárbara Venegas. El problema de rutas de vehículos: Extensiones y métodos de resolución, estado del arte. In *Workshop Internacional EIG2009*, Temuco, Chile, 2009. Universidad de La Frontera, Departamento de Ingeniería de Sistemas.
- [21] Carlos Maldonado. Un problema fundamental en la investigación: Los problemas p vs. np. *Revista Logos, Ciencia & Tecnología*, 4(2):10–20, 2013.
- [22] Abraham Duarte Muñoz, Juan José Pantrigo Fernández, and Micael Gallego Carrillo. *Metaheurísticas*. Dykinson, Madrid, España, 2007.
- [23] Numerentur. Gráfico de complejidad computacional. <https://numerentur.org/wp-content/uploads/2019/08/grafico-complejidad-computacionala3.png>, 2019. Imagen descargada el 15 de julio de 2025.
- [24] Edwin Montes Orozco. Metaheurísticas para el problema de ruteo de vehículos con ventanas de tiempo (vrp-tw), 2017.
- [25] Christos H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.
- [26] Christos H. Papadimitriou and Kenneth Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Dover Publications, New York, 1998.
- [27] Marius M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35(2):254–265, 1987.

- [28] J. Torres-Jiménez, A. Aguilar-Melendéz, and M. G. Cedillo-Campos. El problema del agente viajero con restricciones de ventanas de tiempo: revisión y clasificación de métodos de solución. *Revista Iberoamericana de Automática e Informática Industrial*, 15(4):440–452, 2018.
- [29] Paolo Toth and Daniele Vigo. *Vehicle Routing: Problems, Methods, and Applications*. SIAM (Society for Industrial and Applied Mathematics), 2nd edition, 2014.



# Apéndices