Centro de Investigación Científica y de Educación Superior de Ensenada, Baja California

Maestría en Ciencias en Ciencias de la Computación

Optimización multiobjetivo para el enrutamiento de vehículos con penalización ambiental en ciudades inteligentes

Tesis
para cubrir parcialmente los requisitos necesarios para obtener el grado de Maestro en Ciencias

Presenta:

Luis Bernardo Pulido Gaytán

Ensenada, Baja California, México
2019

Tesis defendida por

Luis Bernardo Pulido Gaytán

y aprobada por el siguiente Comité

Dr. Andrey Chernykh
Director de tesis

Dr. Israel Marck Martínez Pérez

Dr. Gustavo Olague Caballero

Dr. Raúl Rivera Rodríguez

Dr. Ubaldo Ruiz López
Coordinador del Posgrado en Ciencias de la Computación

Dra. Rufina Hernández Martínez
Directora de Estudios de Posgrado

Resumen de la tesis que presenta **Luis Bernardo Pulido Gaytán** como requisito parcial para la obtención del grado de Maestro en Ciencias en Ciencias de la Computación.

Optimización multiobjetivo para el enrutamiento de vehículos con penalización ambiental en ciudades inteligentes

Resumen aprobado por:

_____

Dr. Andrey Chernykh
Director de tesis

Las ciudades albergan a más de la mitad de la población en tan solo el dos porciento de la superficie terrestre, pero consumen el 75 porciento de los recursos que se extraen del planeta. Este abrupto crecimiento demográfico en áreas urbanas ha empeorado el nivel de contaminación en la ciudad, así como los problemas de congestionamientos viales. En este intenso proceso de urbanización, la gestión exitosa del crecimiento urbano se vuelve crucial para garantizar el desarrollo sostenible a nivel económico, social y ambiental. En este contexto, las ciudades inteligentes proponen la incorporación sensata y sistemática de tecnologías para optimizar el uso de la infraestructura existente y así, lograr una ciudad sustentable e inclusiva. En esta tesis se propone el diseño, implementación y análisis de un algoritmo genético celular de optimización multiobjetivo para la resolución del problema de enrutamiento de vehículos con penalización ambiental, permitiendo disminuir el impacto ambiental en áreas altamente concurridas, al tiempo que se provee de rutas alternativas capaces de minimizar los costos asociados de trasladarse de un lugar a otro, esto bajo la premisa de que no siempre el camino más corto representa la mejor solución. El algoritmo presentado minimiza simultáneamente tres importantes objetivos: tiempo de viaje, emisión de contaminantes y una penalización ambiental, donde esta última representa el costo implícito de trasladarse por un determinado segmento del mapa. Es decir, se obtiene un conjunto de soluciones no-dominadas, las cuales representan rutas alternativas que evitan desplazarse por zonas con un alto grado de contaminación y que, a su vez, minimizan el tiempo y la cantidad de emisiones al recorrer una determinada red de rutas. Como instancia del problema, se utilizó un grafo conexo dirigido compuesto por 6104 nodos, el cual representa la topología subyacente de la red de carreteras en la ciudad de Oldemburgo. Dicho grafo se segmentó en tres diferentes tipos de zonas, las cuales simbolizan áreas con distinta penalización. El análisis experimental realizado exhibe un rendimiento competitivo del enfoque propuesto en términos de convergencia y diversidad, esto con respecto a NSGA-II y SPEA2, dos algoritmos multiobjetivo bien conocidos en la literatura.

Palabras clave: Algoritmos evolutivos, ciudades inteligentes, enrutamiento de vehículos, metaheurísticas, optimización multiobjetivo.

Abstract of the thesis presented by **Luis Bernardo Pulido Gaytan** as a partial requirement to obtain the Master of Science degree in Computer Science.

Multiobjective optimization of vehicle routing with environmental penalty in smart cities

Abstract approved by:

_____

Dr. Andrey Chernykh

Thesis Director

Cities host more than half of the population in only two percent of the earth's surface and consume 75 percent of the resources extracted from the planet. This abrupt demographic growth in urban areas has worsened the level of pollution in the city, as well as the problems of road congestion. In this intense urbanization process, successful management of urban growth becomes crucial to ensure sustainable development at economic, social, and environmental levels. In this context, smart cities propose the reasonable and systematic incorporation of technologies to optimize the use of existing infrastructure and thus, achieve a sustainable and inclusive city. This thesis proposes the design, implementation, and analysis of a multiobjective optimization cellular genetic algorithm for the vehicle routing problem with environmental penalty, allowing to diminish the environmental impact in highly concurred areas, while providing alternative routes that minimize the associated costs of moving from one place to another, this under the premise that not always the shortest path represents the best solution. The presented algorithm simultaneously minimizes three important objectives: travel time, emission of pollutants, and an environmental penalty, where the latter represents the implicit ecological cost of moving through a certain segment of the map. That is, a set of non-dominated solution is obtained, which represent alternative routes that avoid traveling through areas with a high degree of pollution and that, in turn, minimize the time and quantity of emissions when traveling through a particular network of routes. As an instance of the problem, a directed graph consisting of 6104 nodes was used, which represents the underlying topology of the road network in the city of Oldenburg. This graph was segmented into three different types of zones, which symbolize areas with different penalties. The experimental analysis carried out exhibits a competitive performance of the proposed approach in terms of convergence and diversity, this with respect to NSGA II and SPEA2, well-known multiobjective algorithms in the literature.

Keywords: Evolutionary algorithms, metaheuristics, multiobjective optimization, smart cities, vehicle routing.

Dedicatoria

A mi amiga y

compañera de toda la vida, mi madre.

Agradecimientos

Primeramente, agradezco a mi familia por su inmensurable apoyo. A mi madre, por darme todo incondicionalmente y ser un ejemplo constante de trabajo, sacrificio y responsabilidad. Esto es más que un agradecimiento, es una muestra de admiración, respeto y amor. A mi hermano Manuel Pulido, por ser un modelo a seguir de tenacidad, superación y capacidad. A mi hija y a mi esposa, por compartir conmigo cada sueño, y ser mi principal motivación para seguir adelante.

No hay palabras para expresar mi agradecimiento a mi director de tesis, el Dr. Andrey Chernykh. Estoy profundamente agradecido con él por darme la oportunidad de contribuir a su trabajo de investigación y por su total disponibilidad para compartir sus conocimientos conmigo. Para él, mi interminable respeto y admiración.

A los miembros de mi comité de tesis, Dr. Israel Marck Martínez Pérez, Dr. Gustavo Olague Caballero y Dr. Raúl Rivera Rodríguez, por sus atinadas observaciones respecto a mi trabajo de investigación, gracias por el tiempo y dedicación que esto representa.

A mis compañeros y amigos, ya que gracias a su amistad y al apoyo mutuo hicieron de este camino algo grato e inolvidable. Pero principalmente por la sensatez y honestidad de sus consejos.

Al Centro de Investigación Científica y de Educación Superior de Ensenada, Baja California. Gracias por permitirme pertenecer a tan distinguida institución.

Al Consejo Nacional de Ciencia y Tecnología (CONACyT) por brindar el apoyo económico para realizar mis estudios de maestría. Gracias.

Tabla de contenido

Lista de figuras

Lista de tablas

Capítulo 1. Introducción

Las ciudades son consideradas piezas elementales para el futuro, ya que juegan un papel preponderante en aspectos socio-económicos de todo el mundo, estas zonas urbanas albergan a más de la mitad de la población en tan solo el 2% de la superficie terrestre, pero consumen el 75% de los recursos que se extraen del planeta (Bocquier, 2005). Este abrupto crecimiento conlleva a una extensa gama de problemas como dificultades en el manejo de desechos, escases de recursos, contaminación ambiental, problemas de salud humana, congestionamientos viales, entre muchos otros (Chourabi et al., 2012), por esta razón pensar en un desarrollo urbano sostenible es una discusión que ha recibido atención constante de investigadores y políticos por varios años.

En este intenso proceso de urbanización, la gestión exitosa del crecimiento urbano se vuelve crucial para garantizar el desarrollo sostenible a nivel económico, social y ambiental. En este contexto, las ciudades inteligentes ("*Smart Cities*") proponen la incorporación sensata y sistemática de tecnologías para la cohesión de todos los aspectos que involucran problemas de tipo social, institucional e infraestructural en un ecosistema urbano (Albino et al., 2014); áreas tan diversas como la educación, salud, seguridad pública, vivienda, energía, transporte y logística, pueden mejorarse, interconectarse y tornarse más eficientes gracias a la incorporación de tecnología (Washburn y Sindhu, 2010). Las ciudades inteligentes permiten reducir costos, hacer un uso eficiente de los recursos y fomentar la participación activa de los ciudadanos en los procesos de toma de decisiones, con el fin de alcanzar una ciudad sustentable e inclusiva.

La organización geográfica de los escenarios urbanos, aunada a la complejidad de las actividades desarrolladas en las ciudades modernas, imponen serios desafíos a la movilidad de los ciudadanos. De tal forma, a través del paradigma de las ciudades inteligentes, conocido comúnmente como movilidad inteligente, se pretende dar frente a problemáticas asociadas a la movilidad urbana en estas grandes ciudades, como los congestionamientos viales o los altos niveles de contaminación ambiental. Dichas dificultades suelen relacionarse con la incapacidad de los sistemas de transporte público para satisfacer las necesidades de un número creciente de usuarios (Peña et al., 2017), así como a la ineficacia de la infraestructura en estas zonas. Sin embargo, dado que modificar la infraestructura existente representa un alto costo monetario, se ha optado por aplicar diversas técnicas de inteligencia computacional para la generación de soluciones de gestión, como la introducción y aplicación de límites de velocidad variable, la imposición de precios diferenciados para carreteras o la optimización del tiempo en señales de tráfico,

todo esto para mejorar de cierta forma el rendimiento de la red de carreteras existente (De Coensel et al., 2012).

Debido a su potencial para mejorar la seguridad vial, reducir la congestión del tráfico y mejorar la movilidad de personas y bienes, los Sistemas de Transporte Inteligente (ITS: por sus siglas en inglés "*Intelligent Transportation Systems*"), también conocidos bajo el concepto de movilidad inteligente, han generado un entusiasmo considerable en la comunidad científica (Barth et al., 2015). Además, los ITS juegan un papel importante en la disminución de contaminantes, así como en la reducción en el consumo de energía (Massobrio et al., 2014). En tal contexto, en el marco de este proyecto se presenta el diseño, implementación y análisis de algoritmos bio-inspirados de optimización multiobjetivo para la resolución del problema de enrutamiento de vehículos con penalización ambiental, permitiendo disminuir el impacto ambiental en áreas altamente concurridas, al tiempo que se provee de rutas alternativas que minimizan los costos asociados de trasladarse de un lugar a otro.

1.1 Motivación

La urbanización mundial se ha llevado a cabo junto con una incorporación creciente de tecnologías de la información y comunicación a la infraestructura actual. En consecuencia, se acuñó un nuevo término que describe la interacción entre estos dos fenómenos; las ciudades inteligentes. De acuerdo con Barrionuevo et al. (2015):

> *"El desarrollo de una ciudad inteligente implica utilizar toda la tecnología y recursos disponibles de manera eficiente y coordinada para desarrollar centros urbanos que sean a la vez integrados, habitables y sostenibles".*

En esencia, el paradigma de las ciudades inteligentes propone el aprovechamiento de la tecnología para mejorar la calidad y eficiencia de los servicios urbanos. En este contexto, la incorporación de artefactos inteligentes a los sistemas físicos tradicionales implementados en zonas urbanas, aunado a la aparición de sensores ciudadanos como dispositivos móviles y vestibles, generan grandes volúmenes de datos que presentan oportunidades y desafíos sin precedentes. Siendo así, resulta evidente la necesidad de un análisis exhaustivo de dichos datos para la extracción de información significativa que funja como soporte en los procesos de toma de decisiones.

Una de las áreas más abordadas dentro del paradigma de las ciudades inteligentes es la movilidad urbana, línea de trabajo conocida comúnmente como movilidad inteligente. El concepto de movilidad inteligente o ITS se basa en un principio simple definido por la integración de tecnologías sinérgicas, inteligencia computacional y conceptos de ingeniería para mejorar el funcionamiento de los sistemas de transporte (Minister of Infrastructure and the Environment, 2016). Los avances tecnológicos, en especial en informática y sistemas de comunicación, han permitido recopilar grandes volúmenes de información en tiempo real sobre el estado del tráfico en determinadas zonas de la ciudad, movilidad de la población o el nivel de partículas en suspensión presentes en el aire. De esta manera, ha sido posible proveer a usuarios, operadores de diferentes servicios y organizaciones gubernamentales de nuevas herramientas y aplicaciones capaces de reducir la contaminación atmosférica o acústica y controlar los flujos de tráfico en áreas urbanas.

Los congestionamientos viales son uno de los principales problemas que enfrentan las ciudades en la actualidad y se sabe que empeorarán a medida que aumente el número y el tamaño de dichas zonas (Nambiar et al., 2018). Un aumento en los niveles de congestión es directamente proporcional a la cantidad de gases de efecto invernadero (GEI) emitidos por cada uno de los vehículos, así como a una disminución en la calidad de viaje percibida por los individuos presentes en dicha situación. En este sentido, es evidente la necesidad de una toma de decisiones logísticas y operativas que permitan reducir los niveles de congestión e índices de contaminación partiendo de una planeación táctica pensada para tal finalidad.

Por tanto, diversos estudios se han dedicado a dar solución a este problema desde diversas perspectivas, como coordinar el tiempo de señales de tránsito a lo largo de carreteras principales en función del flujo de tráfico y los niveles de congestión (Girianna y Benekohal, 2002; De Coensel et al., 2012), o al responder dinámicamente a incidentes de tránsito o de clima inclemente, conduciendo a múltiples vehículos hacia vías colindantes al área afectada (Nambiar et al., 2018). Asimismo, se ha optado comúnmente por fomentar los viajes compartidos en automóvil, conocidos bajo el término anglosajón "*car pooling*". Dicha alternativa no solo es económicamente menos costosa, sino también más respetuosa con el medio ambiente (Nambiar et al., 2018) a niveles individuales y colectivos. De esta forma, logrando que más personas usen el mismo vehículo hacia distintos destinos, es posible minimizar los costos de traslado y la cantidad de automóviles presentes en las calles, contribuyendo directamente a la reducción del impacto ambiental generado por la movilidad urbana (Fagúndez et al., 2014).

Los objetivos de los ITS se formulan mediante la parametrización de metas establecidas por entes interesados (Gordon, 2016). En general, un conjunto de metas amplias es: mejorar la seguridad de las redes de transporte, incrementar la eficiencia operativa y capacidad tanto de los sistemas de transporte como de la infraestructura existente, reducir el consumo de energía e impacto ambiental, mejorar la productividad económica tanto de usuarios como de operadores, por último, mejorar la movilidad individual, convivencia y comodidad de estas redes.

Sin embargo, el proceso de *optimizar* depende específicamente de la parte interesada de la que se esté hablando, estas tienden a tener objetivos diferentes y ocasionalmente contradictorios. Por ejemplo, el sector empresarial tiene como objetivo vender sus productos a los clientes mientras intentan maximizar sus niveles de servicio. Los transportistas desean minimizar los costos de transporte incurridos. Los residentes desean vivir en un ambiente agradable, libre de contaminantes, congestionamientos y accidentes. Los administradores, por otro lado, están más interesados en elevar las condiciones económicas y ambientales de la ciudad (Kim et al., 2015).

1.2 Antecedentes

El problema de enrutamiento de vehículos (VRP: por sus siglas en inglés "*Vehicle Routing Problem*") consiste en encontrar las rutas óptimas para un conjunto de vehículos al satisfacer una demanda de viajes. Este problema fue formulado como un modelo de programación matemática por Dantzig y Ramser (2008). Posteriormente, en 1964 Clarke y Wright propusieron un método heurístico seminal para su resolución. En 1981, Lenstra y Kan demostraron que VRP es NP-difícil.

Desde entonces diferentes modelos y algoritmos han sido desarrollados para problemas relacionados al enrutamiento de vehículos y problemáticas análogas, los cuales pueden clasificarse en dos grandes ramas: exactos y heurísticos. Los métodos exactos permiten encontrar el óptimo global a expensas de un alto costo computacional, presentando un desempeño ineficiente en problemas con alta dimensionalidad, discontinuos y/o NP-completos. Por otro lado, los métodos heurísticos o metaheurísticos son técnicas estocásticas capaces de proveer de "*buenas*" aproximaciones hacia las soluciones óptimas en tiempos razonables. Si bien se ha demostrado que estas eventualmente pueden llegar a alcanzar el caso óptimo para un problema dado, en la mayoría de las ocasiones esto no puede ser garantizado. No obstante, gracias a su facilidad de implementación, alta adaptabilidad, bajo consumo de recursos y velocidades de respuesta

aceptables, han sido objeto de estudio durante décadas. Por lo general, las metaheurísticas son capaces de proporcionar buenas soluciones a una amplia gama de problemas de optimización que los métodos determinísticos tradicionales encuentran complicados. La **Figura 1** muestra ejemplos comunes de cada grupo.

La información de tráfico en tiempo real ha sido un factor clave para el problema de enrutamiento de vehículos (Kim et al., 2015), donde los ITS se han introducido para el aprovechamiento de esta información y proporcionar rutas sensatas (Lee et al., 2009). Se han promulgado normativas sobre vehículos y tiempos de acceso para reducir la contaminación atmosférica o acústica y controlar los flujos de tráfico en las zonas urbanas (Kim et al., 2015). Recientemente, los temas relacionados al consumo de combustible y problemas ambientales han recibido una atención cada vez mayor por parte de la comunidad científica (Peña et al., 2017).

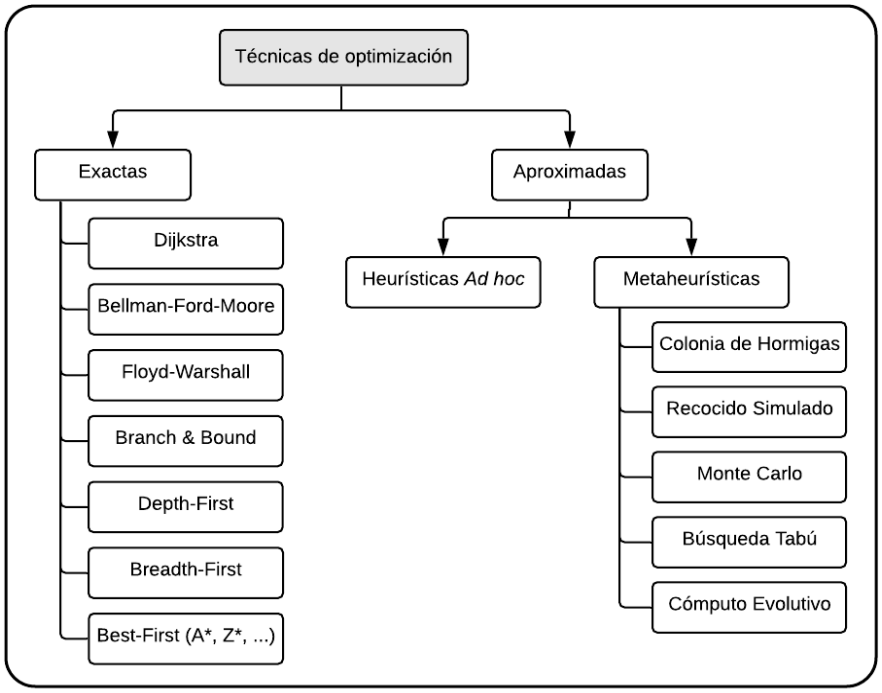


Figura 1. Clasificación de técnicas de optimización.

Un problema crucial en estos sistemas es el encontrar una ruta óptima que minimice los costos de ir de un lugar a otro. No obstante, en la mayoría de las ocasiones el camino más corto no siempre representa la mejor solución, esto desde el punto de vista de la satisfacción del usuario (Chakraborty et al., 2005). Bajo esta premisa enfoques de optimización mono-objetivo presentan una notable desventaja con respecto a enfoques multiobjetivo: su incapacidad para optimizar más de un aspecto a la vez al momento de ofrecer

soluciones al problema. Por otra parte, dada la magnitud del espacio de búsqueda al tratar con instancias realistas y altamente no determinísticas como lo son las redes de carreteras hoy en día, los algoritmos exactos no parecen ser una alternativa viable, ya que el objetivo principal yace en encontrar soluciones potenciales en tiempos computacionales reducidos. Por ello, técnicas como recocido simulado, métodos de Monte Carlo, búsqueda tabú o algoritmos evolutivos, son métodos comúnmente implementados para dar resolución este problema en particular. A continuación, se describen brevemente trabajos propuestos en la literatura para cada una de las técnicas previamente mencionadas, concentrando esfuerzos en tópicos relacionados a VRP.

1.2.1 Métodos exactos

VRP es considerado una generalización del conocido problema del agente viajero (TSP: por sus siglas en inglés "*Travelling Salesman Problem*"). Por asociación, es posible dar solución a dicha problemática a través del algoritmo Dijkstra, método exacto basado en la técnica de etiquetado, el cual consiste en encontrar la ruta con el costo más bajo (generalmente se refiere a la ruta más corta) desde un nodo a todos los nodos en una red de rutas. Su complejidad de cálculo es O(n^2) siendo n el número de nodos en la red (Mengyin Fu et al., 2004). Otros algoritmos exactos para el cálculo del camino más corto son: Floyd-Warshall (Floyd, 1962), Bellman-Ford-Moore, grafo incremental, umbral, ordenamiento topológico, entre muchos otros. No obstante, para encontrar el camino más corto del problema de uno a uno, vale la pena considerar el algoritmo Dijkstra, ya que este termina tan pronto el nodo destino sea etiquetado, es decir, se encuentre la ruta más corta. El resto de ellos solo pueden encontrar la solución óptima al calcular el camino más corto a cada uno de los nodos en el grafo (Nha et al., 2012).

Pallotino y Scutellà (1997) afirmaron que los problemas de transporte a menudo ofrecen una compensación entre dos o más objetivos, por ejemplo, minimizar el tiempo de llegada a un destino final y el costo de la ruta. Es decir, no existe una ruta simple que minimice simultáneamente dos o más criterios en conflicto. Müller-Hannemann y Schnee (2007) informaron sobre un problema de enrutamiento ferroviario que enfrenta un compromiso entre el costo monetario y el tiempo de viaje. En estos casos, es necesario encontrar un conjunto de soluciones que considere más de un objetivo simultáneamente, en lugar de encontrar una única solución centrada en un solo criterio. Si bien algunos de estos algoritmos exactos cuentan con una versión multiobjetivo, al examinar gran parte del grafo para el cálculo de cada una de las rutas, presentan un desempeño pobre en términos del tiempo computacional requerido.

1.2.2 Métodos metaheurísticos

Dada la magnitud del espacio de búsqueda al tratar con instancias realistas es necesario utilizar heurísticas o metaheurísticas que permitan calcular soluciones de calidad aceptable en tiempos razonables (Fagúndez et al., 2014). Esto se hace posible en vista de que VRP es un problema de optimización combinatoria, donde las variables de decisión son definidas dentro de un espacio discreto y, el número de alternativas es equivalente a las combinaciones posibles de todos los valores de las variables involucradas.

Por consiguiente, enfoques de optimización por colonias de hormigas (Bell y McMullen, 2004), recocido simulado (Possel et al., 2018), algoritmos genéticos simples (Nunes et al., 2011; Sharma y Mathew, 2011; Fagúndez et al., 2014) o paralelos (Massobrio et al., 2014), han propuesto soluciones para la optimización de rutas. Aunado a la complejidad antes mencionada, es posible encontrarse con problemáticas donde una solución de valor está dada por más de un factor, es decir, problemas que exhiben un comportamiento en donde dos o más de sus objetivos se encuentran en compromiso de manera natural. Por ejemplo, en el contexto de VRP, una solución que reduzca el congestionamiento vial se verá afectada en la distancia a recorrer para alcanzar la posición destino. Por ello, múltiples trabajos de investigación se han enfocado a la obtención de un conjunto de soluciones capaz de producir buenos compromisos entre dos o más objetivos en conflicto, esto a través de algoritmos de optimización multiobjetivo (Serafini, 1994; Peña et al., 2017, 2018), donde usualmente se ofrece un conjunto aproximado de soluciones no-dominadas.

La técnica de recocido simulado se caracteriza por perturbar una posible solución factible s seleccionada aleatoriamente, donde esta perturbación producirá una nueva alternativa s' que sustituirá a la anterior. Si la función *"fitness"* de s' es mejor que la del estado s, se sustituye s por s'. No obstante, dada una probabilidad de aceptación es posible elegir un estado peor al actual, permitiendo escapar de óptimos locales para alcanzar óptimos globales. El algoritmo itera hasta lograr la optimalidad o una aproximación aceptable. La analogía del método yace en el "*recocido simulado*" realizado en mecánica estadística para encontrar el límite inferior de energía de un material. La técnica consiste en calentar y enfriar el material en repetidas ocasiones, promoviendo cambios en la energía de los átomos; el calor permite alcanzar los valores iniciales (mínimo local del problema), mientras que el enfriamiento controlado posibilita alcanzar valores cuyo rendimiento tenga menor energía que los iniciales (mínimo global del problema). Serafini (1994) emplea algoritmos basados en esta técnica y sugiere su uso en problemas con múltiples objetivos; empleando reglas de agregación ponderada de valores a cada uno de los objetivos, diferente a los algoritmos mono-objetivo clásicos. Possel et al. (2018) implementan la técnica de recocido simulado para

dar solución a un problema de optimización de dos niveles relacionado al diseño de redes de transporte, considerando a las externalidades como objetivos a minimizar.

La búsqueda tabú es una meta-estrategia desarrollada por Glover en 1989 como una técnica iterativa que escapa de óptimos locales aceptando soluciones no tan buenas, moviéndose paso a paso hacia soluciones lo más cercanas posible a un óptimo global. A diferencia de métodos que tratan de evitar el estancamiento en óptimos locales, esta técnica utiliza una búsqueda inteligente basada en formas sistemáticas. Esta metaheurística se basa en la premisa de que para clasificar un procedimiento como "*inteligente*", este debe incorporar un mecanismo de memoria y exploración adaptativa. De esta forma, se mantiene un registro tanto de las soluciones visitadas como de los *"caminos"* que las alcanzaron, registro conocido como *lista tabú*. La búsqueda se detiene después de un número definido de iteraciones o después de una serie de iteraciones consecutivas sin ningún cambio significativo a la mejor solución conocida. Esta técnica ha sido ampliamente aplicada al problema de planificación de rutas (Liao y Hu, 2011).

Los métodos conocidos como optimización por colonias de hormigas (ACO: por sus siglas en inglés "*Ant Colony Optimization")* son metaheurísticas inspiradas en la naturaleza de estos formícidos, en el cómo es que cooperan entre sí para encontrar caminos cortos desde su nido hasta fuentes de alimento. Cuando una hormiga encuentra un camino potencial deposita un compuesto químico conocido como feromona. La feromona constituye la información que permite la coordinación de los esfuerzos del enjambre en busca de comida, una hormiga está más inclinada a seguir una trayectoria cuanto más fuerte sea el rastro de feromona depositada en ella. De esta manera, si otras hormigas encuentran ese camino, es más probable que lo sigan en lugar de vagar al azar, donde posteriormente dejarán más feromonas. Eventualmente la mayor parte de hormigas siguen dicha ruta hasta agotar la fuente de alimento (Nha et al., 2012). Bell et al. (2004) implementan ACO para minimizar el costo total de viaje en instancias realistas, donde cada hormiga almacena información sobre los nodos ya visitados y el tiempo estimado para alcanzarlos. En general, ACO ha sido ampliamente utilizado en la optimización de rutas, ya que es capaz de reaccionar a cambios dinámicos en las condiciones de tráfico (Tatomir et al., 2009).

Otra técnica metaheurística ampliamente utilizada para la resolución a VRP y en general a problemas de búsqueda y optimización combinatoria son los algoritmos genéticos (AG). Como bien su nombre lo indica, son métodos de búsqueda estocásticos inspirados en la forma en que las especies evolucionan y adaptan a su entorno, de acuerdo con el principio darwiniano de la selección natural (Kanoh, 2007). A través de estas técnicas se han realizado múltiples aportes para el diseño de redes de transporte (topología y trayectos de las rutas). De esta forma, bajo la premisa de que no siempre la ruta más corta representa la

mejor solución, Chakraborty et al. (2005) proponen un algoritmo genético donde es posible elegir rutas con respecto a diferentes criterios, como la congestión vial, problemas ambientales o comodidad de manejo, esto a pesar de un aumento implícito en el costo resultante. Fagúndez et al. (2014) proponen un AG capaz de minimizar el costo total al distribuir en varios vehículos a un determinado grupo de pasajeros que viajan desde el mismo origen hacia distintos destinos. Sharma et al. (2011) implementan un algoritmo evolutivo multiobjetivo para resolver un problema de diseño de redes de transporte, minimizando dos importantes objetivos: tiempo total de viaje y el costo de daños a la salud ocasionados por emisiones vehiculares. Utilizan una estructura basada en permutaciones de enteros y operadores genéticos clásicos para dicho tipo de codificación.

En la actualidad, las metaheurísticas para la resolución de problemas de optimización multiobjetivo han sido aplicadas extensivamente a una amplia variedad de dominios, los cuales pueden categorizarse en dos grandes grupos: industria y ciencia (Coello et al., 2007). En lo que respecta a las aplicaciones industriales, estas pueden encontrarse en las distintas disciplinas de la ingeniería, donde comúnmente se presentan problemas con modelos matemáticos bien estipulados, por lo que no resulta sorprendente la generación de gran cantidad de aplicaciones en este contexto. Muestras representativas en este dominio pueden encontrarse en: ingeniería eléctrica, hidráulica, aeronáutica, robótica, telecomunicaciones, entre muchas otras. Además, en la literatura es posible encontrar una amplia gama de trabajos relacionados al diseño y fabricación, o calendarización de procesos. Por otra parte, en el contexto científico, por obvias razones las aplicaciones informáticas son las más populares. No obstante, ejemplos pueden encontrarse en áreas como la química, física y medicina. Esta muestra ofrece una perspectiva del creciente interés por adoptar este tipo de técnicas en prácticamente todo tipo de disciplinas.

1.3 Objetivos

1.3.1 Objetivo general

Diseñar, implementar y analizar algoritmos bio-inspirados de optimización multiobjetivo para la planificación táctica en el enrutamiento de vehículos en el contexto de ciudades inteligentes, permitiendo minimizar simultáneamente tres importantes objetivos: tiempo de viaje, emisión de contaminantes y una penalización ambiental, donde esta última representa el costo ambiental implícito de trasladarse por un determinado segmento del mapa.

1.3.2 Objetivos específicos

- Realizar una revisión bibliográfica de las principales técnicas que se han utilizado para dar solución el problema de enrutamiento de vehículos, concentrando esfuerzos en la identificación de antecedentes, estado de desarrollo actual y tendencias.

- Modelar el problema de enrutamiento de vehículos con penalización ambiental con un enfoque de optimización combinatoria.

- Diseñar e implementar algoritmos metaheurísticos multiobjetivo que permitan encontrar soluciones no-dominadas a la problemática abordada, generando buenos compromisos entre objetivos en conflicto.

- Realizar una sintonización de parámetros a través de un análisis de varianzas multifactorial, permitiendo identificar aquella combinación de parámetros que posibilite encontrar soluciones potenciales de manera eficiente y a su vez, ofrezca una mejor exploración y explotación en el espacio de posibles soluciones.

- Evaluar el desempeño de las implementaciones con respecto a indicadores de calidad comúnmente utilizados en la literatura, permitiendo comparar la efectividad del enfoque propuesto frente a técnicas representativas del estado del arte.

1.4 Organización de la tesis

Este trabajo está divido en seis capítulos que desglosan la investigación desarrollada. En el Capítulo 2 se introduce la problemática objeto de esta tesis, así como la formulación matemática utilizada como base para su solución. En el Capítulo 3 se presentan los conceptos teóricos pertinentes a optimización, particularmente con objetivos múltiples, así como técnicas de resolución comúnmente utilizadas para este tipo de problemas. En el Capítulo 4 se revisan a fondo los algoritmos evolutivos enfocados a la resolución de problemas de optimización multiobjetivo, detallando su funcionamiento, características y componentes principales. Asimismo, en este capítulo se detalla el algoritmo elegido para dar respuesta a la problemática abordada. En el Capítulo 5 se muestran los resultados experimentales obtenidos,

describiendo paso a paso la metodología utilizada, desde las métricas empleadas para cuantificar la calidad de las soluciones encontradas, hasta los estudios estadísticos implementados para la parametrización de los algoritmos. De igual manera, se analizan los resultados para determinar la factibilidad de este tipo de técnicas en el enrutamiento de vehículos en distintas topologías y a su vez, se comparan los resultados frente a técnicas representativas del estado del arte. Finalmente, las conclusiones del trabajo se presentan en el Capítulo 6, conjuntamente con las numerosas propuestas que han surgido como posibles líneas de trabajo futuro.

Capítulo 2. Definición del problema

Este capítulo presenta el problema abordado en el marco del proyecto, revisando los principales componentes relacionados con el tema. La Sección 2.1 introduce de manera breve y concisa el problema de enrutamiento de vehículos con penalización ambiental. La Sección 2.2 presenta una caracterización formal de dicha problemática, discutiendo la formulación matemática a utilizar como base para su solución.

2.1 Planteamiento del problema

En años recientes, el proceso de urbanización ha llevado a la concentración de la población en zonas metropolitanas. Este abrupto crecimiento demográfico en áreas urbanas ha empeorado el nivel de contaminación en las ciudades, así como los problemas de congestionamiento vial. En este contexto, se propone el diseño, implementación y análisis de algoritmos bio-inspirados de optimización multiobjetivo para el enrutamiento adecuado de vehículos que permita disminuir el impacto ambiental en áreas altamente concurridas, al tiempo que se provee de rutas alternativas que minimicen los costos asociados de trasladarse de un lugar a otro.

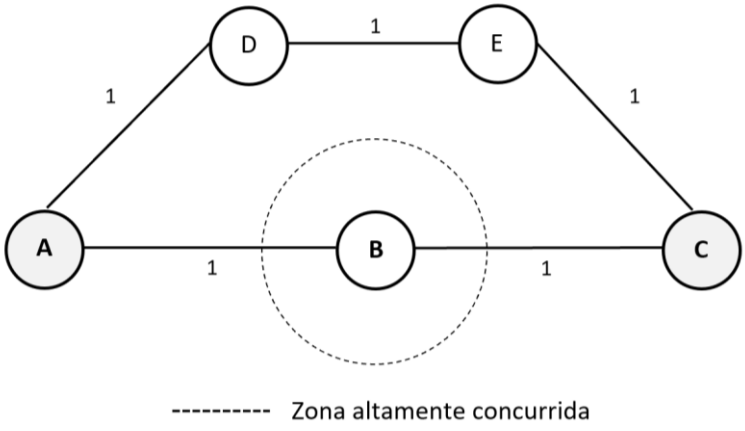


Figura 2. Ejemplo de grafo donde la mejor solución para ir del punto A al punto C no siempre está dada por el camino más corto. Si bien la ruta compuesta por la secuencia de vértices $[A, B, C]$ representa el camino más corto con un costo igual a 2, dicha ruta atraviesa una zona con alta concurrencia y, por ende, altamente contaminada. Por otra parte, la ruta alternativa compuesta por la secuencia de vértices $[A, D, E, C]$ evita desplazarse por este tipo de zonas a expensas de un mínimo aumento en el costo total.

El problema a resolver modela la siguiente situación: un determinado número de vehículos, ubicados en diferentes puntos de origen deciden trasladarse hacia distintos destinos. El problema de optimización consiste en enrutar adecuadamente estos vehículos bajo la premisa de que no siempre el camino más corto representa la mejor solución, esto desde el punto de vista de la satisfacción del usuario (véase **Figura 2**). De esta manera, se pretende minimizar simultáneamente tres importantes objetivos: el tiempo, la cantidad de contaminantes emitidos y una penalización ambiental, donde esta última representa el costo implícito de trasladarse por un determinado segmento del mapa. Es decir, se pretende encontrar un conjunto de soluciones no-dominadas, el cual contenga rutas alternativas que eviten desplazarse por zonas con un alto grado de contaminación y que, a su vez, minimicen el tiempo y la cantidad de emisiones al recorrer una determinada red de rutas.

Como bien se observa en el ejemplo de la **Figura 2**, si bien la ruta compuesta por la secuencia de vértices $[A, B, C]$ representa el camino más corto con una distancia igual a 2, dicha ruta atraviesa una zona con alta concurrencia y, por ende, altamente contaminada. Por otra parte, la ruta alternativa compuesta por la secuencia de vértices $[A, D, E, C]$ evita desplazarse por este tipo de zonas a expensas de un mínimo aumento en la distancia total; considerando que el tiempo de viaje está dado por el cociente entre la distancia y la velocidad del vehículo en curso ($t_{ij}^{b_k} = d_{\mathrm{ij}}/s_k$), esto permite evidenciar el conflicto entre estos dos criterios: tiempo y penalización ambiental. Sin embargo, si bien una ruta alterna permite reducir considerablemente el costo ambiental resultante, al aumentar la distancia a recorrer incrementa directamente la cantidad de contaminantes emitidos, exhibiendo el segundo conflicto entre objetivos: contaminantes emitidos y penalización ambiental. Dicho efecto no sucede para los criterios tiempo y cantidad de emisiones, dado que estos son correlacionados, es decir, sus valores varían sistemáticamente con respecto a los del otro.

2.2 Definición formal del problema

La formulación matemática a utilizar como base para la resolución del problema de enrutamiento de vehículos con penalización ambiental se presenta a continuación. Dados los siguientes elementos:

- Sea $G(V, E)$ un grafo conexo acíclico que representa la topología subyacente de una red de rutas, donde $V = \{1, \dots, m\}$ es el conjunto de intersecciones (vértices) y $E = \{(i, j) | i, j \in V, i \neq j\}$ denota el conjunto de conexiones (aristas) entre vértices en G.

- Sea $e_{ij} = \{d_{ij}, v_{ij}, cont_{ij}, p_{ij}, incl_{ij}\}$ la conexión entre el vértice i y el vértice j, el cual tiene asociado cinco valores no negativos $d_{ij}, v_{ij}, cont_{ij}, p_{ij}$ e $incl_{ij}$, los cuales denotan la distancia, velocidad máxima permitida, contaminación ambiental, penalización ambiental e inclinación de la ruta $(i, j) \in E$, respectivamente.

- Un conjunto de vehículos $B = \{b_1, \ldots, b_n\}$, donde $b_k = \{\beta_k, s_k, m_k, FT_k\}$, β_k el coeficiente de consumo de combustible, s_k la velocidad (constante), m_k el peso y FT_k la fuerza de tracción del vehículo b_k, para $k = \{1, \ldots, n\}$.

- Un subconjunto de vértices de inicio $V_{in}^{b_k} \subseteq V$ y un subconjunto de vértices de destino $V_{out}^{b_k} \subseteq V$.

- $t_{ij}^{b_k} = d_{\mathrm{ij}}/s_k$ representa el tiempo requerido para recorrer la arista e_{ij} para el vehículo b_{k}, donde $s_{\mathrm{k}} \leq v_{ij}$.

- Una ruta $r_{ab}^{b_k}$ para cada $b_k \in B$ compuesta por aristas adyacentes $r_{ab}^{b_k} = < e_{a1}, e_{12}, \ldots, e_{zb} >$ donde $e \in E$ y l_{ab} representa la longitud de la ruta $r_{ab}^{b_k}$.

- Sea $T_{ab}^{b_k} = \sum_{(\mathrm{i,j}) \in r_{ab}^{b_k}} t_{ij}^{b_k}$ el tiempo para recorrer la ruta $r_{ab}^{b_k}$ para b_k.

- Sea $P_{ab}^{b_k} = \sum_{(\mathrm{i,j}) \in r_{ab}^{b_k}} p_{ij}^{b_k}$ la penalización ambiental al recorrer la ruta $r_{ab}^{b_k}$ para b_k.

- Sea $c_{ij}^{b_k} = t_{ij}^{b_k} \cdot \beta_k \cdot VSP_{ij}^{b_k}$ la cantidad de contaminantes emitidos por el vehículo b_k al ir del vértice i al vértice j, donde $VSP_{ij}^{b_k} = FT_k \cdot s_k/m_k$.

- Sea $C_{ab}^{b_k} = \sum_{(\mathrm{i,j}) \in r_{ab}^{b_k}} c_{ij}^{b_k}$ la cantidad de contaminantes emitidos por el vehículo b_k al recorrer la ruta $r_{ab}^{b_k}$.

- Sea $T = \sum_{k=1}^{n} T_{ab}^{b_k}$ el tiempo requerido para todos los vehículos al recorrer el conjunto de rutas.

- Sea $P = \sum_{k=1}^{n} P_{ab}^{b_k}$ la penalización ambiental para todos los vehículos al recorrer el conjunto de rutas.

- Sea $C = \sum_{k=1}^{n} C_{ab}^{b_k}$ las emisiones de todos los vehículos al recorrer el conjunto de rutas.

Se busca entonces identificar aquel conjunto de soluciones no-dominadas que permitan minimizar el tiempo necesario para cada uno de los vehículos al ir del punto de entrada al punto de salida, a la vez que se minimiza la cantidad de emisiones y la penalización ambiental, donde esta última representa el costo implícito de viajar por una determinada zona en G, contribuyendo así a la reducción del impacto ambiental o contaminación en esa región. En este sentido, es posible expresar dicha problemática como un problema de calendarización (bajo la notación de tres campos (Graham et al., 1979)), compuesto por procesadores, trabajos y criterios de optimización (véase **Ecuación 1**).

$$G \mid B \mid min\,\{T, C, P\}. \qquad (1)$$

Tomando en cuenta la complejidad del problema al utilizar instancias de tamaños realistas, los algoritmos exactos tradicionales no resultan útiles para una planificación eficiente. Por ello, es necesario la implementación de heurísticas o metaheurísticas capaces de calcular soluciones de calidad aceptable en tiempos razonables. En este contexto, con el fin de optimizar simultáneamente los tres objetivos en conflicto anteriormente mencionados, se hizo uso de un enfoque evolutivo multiobjetivo basado en la dominancia de Pareto, permitiendo encontrar un conjunto de soluciones no-dominadas que generen buenos compromisos entre objetivos en conflicto, reduciendo los costos asociados de trasladarse de un lugar a otro sin dejar de lado aspectos ecológicos.

Capítulo 3. Optimización multiobjetivo

En este capítulo se introducen los conceptos teóricos pertinentes a una clase particular de problemas, llamada optimización multiobjetivo. Asimismo, se presentan diversas técnicas comúnmente utilizadas en la literatura para su resolución.

3.1 Introducción

Áreas como la biología, medicina, economía, industria, ingeniería y otras muchas disciplinas, comúnmente presentan problemáticas donde una solución de valor está dada por más de un factor, es decir, estos problemas exhiben un comportamiento en donde dos o más de sus objetivos se encuentran en compromiso de manera natural. Por esta razón, identificar soluciones de calidad en tiempos razonables ha sido objeto de estudio por parte de comunidades de investigación de operaciones y optimización en general.

A diferencia de problemas donde únicamente se busca satisfacer un simple objetivo y la optimalidad está dada por una única solución que lo maximiza o minimiza, en los problemas de optimización multiobjetivo (MOOP: por sus siglas en inglés “Multiobjective Optimization Problem”), también llamados problemas de optimización multi-criterio o de optimización vectorial, se busca encontrar aquel conjunto de soluciones que satisfagan las restricciones y a su vez, optimicen dos o más funciones simultáneamente, permitiendo así a tomadores de decisiones seleccionar la solución que mejor se apegue a sus necesidades.

Una posible alternativa para dar solución a los problemas asociados a optimización multiobjetivo, es abordarlos como problemas mono-objetivo, es decir, asignar una función de calidad numérica a cada uno de los criterios de optimización (Serafini, 1994), y luego combinar estos puntajes en un única función a través de una ponderación. Este enfoque, comúnmente llamado escalarización o suma ponderada, se ha utilizado durante años dentro de las comunidades optimización heurística para hacer más *tratables* este tipo de problemáticas, sin embargo, este adolece de una serie de inconvenientes, como su incapacidad para capturar todas las preferencias del usuarios, o incluso conocer el rango de posibles soluciones (Eiben y Smith, 2015).

Por tal razón, múltiples enfoques multiobjetivo han sido propuestos en la literatura para abordar los problemas de decisión con múltiples criterios, capaces de identificar simultáneamente un conjunto diversos de soluciones con alta calidad, donde al evaluar aquel conjunto se produzcan vectores cuyos componentes representan compensaciones en el espacio objetivo.

3.2 Fundamentos teóricos

En un problema de optimización se conoce de antemano el modelo computacional a ejecutar, así como la salida esperada (o al menos su descripción). De esta manera, tomando como base el modelo clásico de caja negra en sistemas computacionales (véase **Figura 3**), la tarea consiste básicamente en encontrar la entrada ideal que conlleve satisfactoriamente a dicha salida o bien, la optimice. La mayor parte de los conceptos presentados en esta sección pueden encontrarse en los libros de Coello et al. (2007), Eiben y Smith (2015) y Olague (2016).

Figura 3. Problema de optimización. Ejemplo de problema de optimización tomando como base el modelo de caja negra de un sistema computacional.

No obstante, para poder definir a una solución como “óptima” es necesario previamente describir bajo qué criterios lo es o bien, a que restricciones está sujeta. Por tal razón, con el fin de facilitar el entendimiento de un MOOP y los requerimientos específicos que se pretenden abordar con técnicas de solución propuestas a lo largo de este documento, como primer paso en este capítulo se formulan los términos matemáticos de un problema en teoría de la optimización.

Un problema de optimización con un único objetivo se define como la minimización o maximización de una función $f(\mathbb{x})$ sujeta a $g_i(\mathbb{x}) \leq 0, i = \{1, \dots, m\}$, y $h_j(\mathbb{x}) = 0, j = \{1, \dots, p\}\, \mathbb{x} \in \Omega$. Una solución que minimice o bien, maximice la función escalar $f(\mathbb{x})$ donde $\mathbb{x}$ es un vector de variables de decisión n-dimensional $\mathbb{x} = (x_1, x_2, \dots, x_n)$ para algún universo Ω.

Se observa que $g_i(\mathbb{x}) \leq 0$ y $h_j(\mathbb{x}) = 0$ representan restricciones que deben ser satisfechas al momento de optimizar $f(\mathbb{x})$. Asimismo, el universo Ω contiene todas las posibles soluciones que pueden ser utilizadas para satisfacer una evaluación de la función $f(\mathbb{x})$ y las restricciones presentadas. Por otra parte, cabe mencionar que $\mathbb{x}$ puede ser un vector con variables continuas o discretas, así como f puede ser continua y discreta.

De este modo, un método para encontrar el óptimo global para alguna función $f(\mathbb{x})$ con un único criterio de optimización, se conoce como optimización global y puede ser definido como: dada una función $f: \Omega \subseteq \mathbb{R}^n \to \mathbb{R}, \Omega \neq \emptyset$, para un $\mathbb{x} \in \Omega$ el valor de $f^* \triangleq f(\mathbb{x}^*) > -\infty$ es llamado optimo global si y solo si $\forall \mathbb{x} \in \Omega: f(\mathbb{x}^*) \leq f(\mathbb{x})$, donde $\mathbb{x}^*$ es la solución mínima global, f es la función objetivo y el conjunto Ω es la región factible de $\mathbb{x}$. Sin embargo, si bien para un problema de optimización mono-objetivo existe una única solución óptima, en el caso de un MOOP se cuenta con un conjunto de soluciones de dimensión indefinida que, al ser evaluados, se producen unos vectores cuyos componentes representan los valores de Ω en el espacio de posibles soluciones.

De acuerdo con Osyczka (1978), un problema de optimización multiobjetivo puede ser definido como:

> *"Problema de encontrar un vector de variables de decisión que satisfagan un cierto conjunto de restricciones y optimice un conjunto de funciones objetivo. Estas funciones forman una descripción matemática de los criterios de desempeño que suelen estar en conflicto unos con otros y que se suelen medir en unidades diferentes. El término optimizar en este caso toma pues un significado diferente al del caso de problemas mono-objetivo."*

De esta forma, tomando como base la definición de un problema con un único objetivo previamente mencionada, es posible definir formalmente a un MOOP de la siguiente manera:

Encontrar el vector $\vec{x}^* = [x_1^*, x_2^*, \dots, x_n^*]^T$ que satisfaga las m restricciones de desigualdad

$$g(\vec{x}) \leq 0 \quad i = \{1, 2, \dots, m\}, \tag{2}$$

así como las p restricciones de igualdad

$$h_i(\vec{x}) = 0 \quad i = \{1, 2, \dots, p\}, \tag{3}$$

y optimice la función vectorial

$$\vec{f}(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x}), \dots, f_k(\vec{x})]^T. \quad (4)$$

Siendo así, un MOOP consiste en k objetivos reflejados en k funciones objetivo, $m + p$ restricciones y n variables de decisión. Si una solución satisface todas las $m + p$ restricciones es conocida como una solución factible o potencial, la cual forma parte de un conjunto o región factible $\delta \subseteq \Omega$ (**espacio de búsqueda**). Las k funciones objetivo pueden ser lineales o no lineales, así como de naturaleza continua o discreta. La función de evaluación $F: \Omega \rightarrow \Lambda$, es un mapeo del vector de variables de decisión ($\mathbb{x} = x_1, \dots, x_n$) a los vectores de salida ($y = a_i, \dots, a_k$), donde Ω representa el **espacio de decisión** y Λ el **espacio objetivo**. Específicamente, cada punto en el primer espacio (Ω) representa en sí una solución y a su vez, genera un cierto punto en el segundo espacio (Λ), lo que determina la calidad de dicha solución en términos de los valores obtenidos para la función objetivo. La **Figura 4** representa la relación entre estos dos espacios para el caso de $n = 2$, $m = 0$ y $k = 3$.

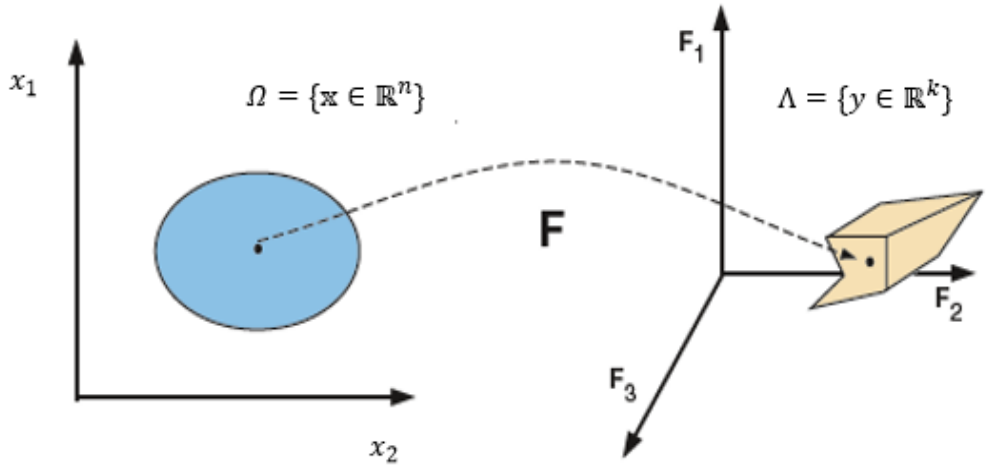


Figura 4. Representación gráfica de mapeo $F: \Omega \rightarrow \Lambda$. Espacio de decisión Ω (izquierda) y su correspondiente espacio objetivo Λ (derecha).

Para poder determinar la calidad de una cierta solución, es necesario contar con algún criterio para evaluarla. Estos criterios se expresan como funciones computables de las variables de decisión a las que se denomina funciones objetivo. En problemas del mundo real, estas funciones objetivo suelen estar en conflicto entre sí, y algunas deben minimizarse mientras otras han de maximizarse. Sin embargo, por

cuestiones de simplicidad, normalmente todas las funciones se convierten ya sea a un problema de maximización o a uno de minimización mediante el teorema de dualidad (Rao, 2009). Las funciones objetivo pueden ser conmensurables (medidas en las mismas unidades) o no conmensurables (medidas en unidades distintas).

Dado que no existe una única solución que optimice dos o más criterios de manera simultánea y en su lugar se generan múltiples soluciones que producen buenos compromisos (llamados "trade-offs" en inglés) entre objetivos en conflicto, es preciso contar con un tomador de decisiones capaz de seleccionar aquel o aquellos puntos que satisfaga sus requerimientos de mejor manera, esto con base a algún modelo de preferencias humanas conocido. En las siguientes subsecciones se introducen conceptos que permiten explicar teóricamente la forma del espacio objetivo y el concepto matemático utilizado para decidir si una solución que da frente a más de un objetivo simultáneamente es mejor con respecto a otra de la misma naturaleza.

3.2.1 Vector ideal

El término de vector ideal está asociado a un vector de variables de decisión correspondientes a los óptimos (factibles) considerando a cada uno de los criterios de optimización del problema de manera aislada (véase **Figura 5**). Nótese que el vector ideal es prácticamente inalcanzable, excepto en el caso de que las funciones objetivo no se encuentre en conflicto.

De esta manera, sea:

$$\mathbb{x}^{*(i)} = \left[x_1^{*(i)}, x_2^{*(i)}, \dots, x_n^{*(i)}\right]^T \qquad (5)$$

el vector de variables que optimiza (maximiza o minimiza) la i-ésima función objetivo $f_i(\mathbb{x})$ o bien, dicho de otra forma, el vector $x^{*(i)} \in \Omega$ es tal que:

$$f_i\left(\mathbb{x}^{*(i)}\right) = \underset{x \in \Omega}{\text{óptimo}}\, f_i(\mathbb{x}). \qquad (6)$$

Por lo tanto, el vector ideal $\mathbf{f}^*$ está dado por:

$$\mathbf{f}^* = [f_1^*, f_2^*, \dots, f_i^*]^T, \qquad (7)$$

donde f_i^* denota el valor óptimo de la i-ésima función objetivo, el punto en $\mathbb{R}^n$ que determina el vector $\mathbf{f}^*$ se considera la solución ideal para el problema propuesto.

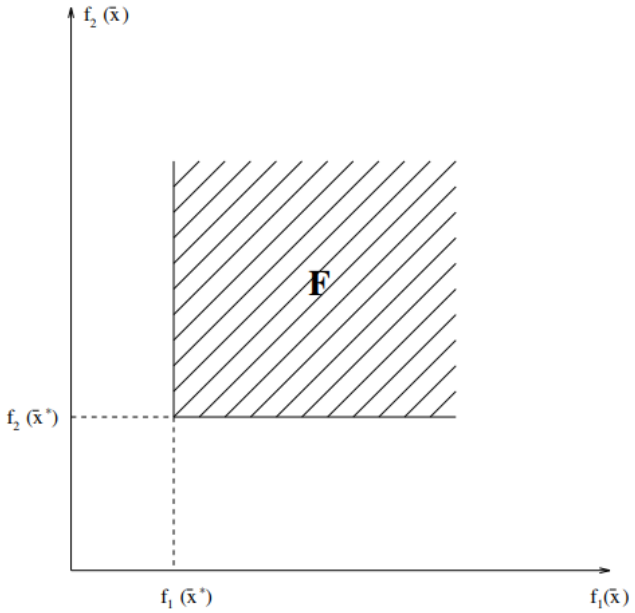


Figura 5. Representación del vector ideal. Sea F el espacio de posibles soluciones**,** dadas dos funciones objetivo f_1 y f_2, el punto en $\mathbb{R}^n$ que determina el vector ideal está representado por las coordenadas $[\, f_1(\bar{x}^*), f_2(\bar{x}^*)]$.

El vector ideal es utilizado por algunos métodos de programación matemática que normalmente tratan de minimizar la distancia (en el espacio de las funciones objetivo) entre una solución dada y el vector ideal. Determinar el vector objetivo ideal suele ser sencillo excepto cuando las funciones objetivo (vistas por separado) presentan multi modalidad.

En este contexto, otro concepto común es el vector objetivo utópico, definido comúnmente como $z^{**} \in \mathbb{R}^n$, el cual es un vector objetivo infactible cuyos componentes se forman mediante $z^{**} = z_i^* - \varepsilon_i$ para toda $i = 1, \dots, k$, donde z_i^* es un componente del vector ideal y $\varepsilon_i > 0$ es un escalar relativamente pequeño pero computacionalmente significativo.

3.2.2 Convexidad y concavidad

Un punto importante a considerar al abordar un problema de optimización multiobjetivo es sin duda alguna la naturaleza del espacio de posibles soluciones, ya que de ello depende directamente el desempeño de las técnicas de optimización a implementar para su resolución. Por ello, conocer de antemano el dominio de la función a optimizar resulta de gran utilidad al momento de generar soluciones factibles o potenciales para más de un objetivo de manera simultánea, capaces de identificar soluciones de alta calidad en espacios de búsqueda multi dimensionales con particularidades que complican la convergencia hacia aquel conjunto de puntos deseado.

Una función $\phi(x)$ es llamada convexa sobre el dominio de $\mathbb{R}$ si para dos vectores cualesquiera $\vec{x}_1$ y $\vec{x}_2 \in \mathbb{R}$,

$$\phi(\theta\vec{x}_1 + (1-\theta)\vec{x}_2) \leq \theta\phi(\vec{x}_1) + (1-\theta)\phi(\vec{x}_2), \qquad (8)$$

donde θ es un escalar en el rango $0 \leq \theta \leq 1$.

La función $\phi(\vec{x})$ es estrictamente convexa si, para $\vec{x}_1 \neq \vec{x}_2$, el signo $\leq$ de la ecuación (**8**) puede ser reemplazado con una desigualdad ($<$).

Una función convexa no puede tener ningún valor mayor que los valores de la función obtenidos mediante interpolación lineal entre $\phi(\vec{x}_1)$ y $\phi(\vec{x}_2)$.

Si la desigualdad reversa de la ecuación (**2**) se cumple, la función es cóncava. De tal forma, una función $\phi(\vec{x})$ es estrictamente cóncava si $-\phi(\vec{x})$ es estrictamente convexa. Las funciones lineales son convexas y cóncavas al mismo tiempo.

Un conjunto o región de puntos se define como un conjunto convexo en un espacio n-dimensional si, para todos los pares de puntos $\vec{x}_1$ y $\vec{x}_2$ en el conjunto, el segmento rectilíneo que los une está también enteramente dentro del conjunto. De tal forma, todo punto $\vec{x}$, donde

$$\vec{x} = \theta\vec{x}_1 + (1-\theta)\vec{x}_2,\ 0 \leq \theta \leq 1, \qquad (9)$$

está también en el conjunto. La **Figura 6** muestra un ejemplo de función convexa y no convexa (cóncava).

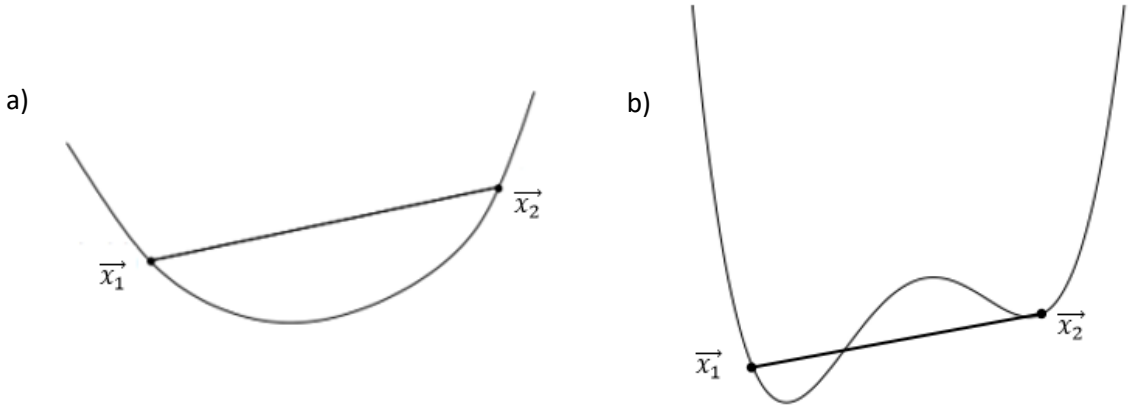


Figura 6. Ejemplo de función convexa y no convexa. a) Función convexa y b) función no convexa.

3.2.3 Dominancia y optimalidad de Pareto

Una noción particularmente importante dentro de la optimización multiobjetivo es sin duda alguna la **dominancia ($\succ$)** o **dominancia de Pareto**, la cual puede definirse como sigue: dadas dos soluciones (u y v), ambas con puntuaciones según algún conjunto de funciones objetivo, se dice que una solución domina a la otra si su puntaje es al menos igual de alto para todos los objetivos, y estrictamente mayor para al menos alguno de ellos (para un caso de maximización). Es posible representar las puntuaciones que obtiene una solución u como un vector n-dimensional $\vec{u} = (u_1, u_2, \dots, u_n)$. Usando el símbolo $\succ$ para indicar la dominancia, podemos definir formalmente $u \succ v$ como sigue:

$$u \succ v \;\Leftrightarrow\; \forall_i \in \{1, \dots, n\}\, u_i \geq v_i \;\wedge\; \exists_i \in \{1, \dots, n\}, u_i > v_i. \qquad (10)$$

Es decir, el concepto de dominancia de Pareto implica que, para que una solución domine a otra no debe ser peor en ninguno de los criterios de optimización y debe ser estrictamente mejor en al menos uno de ellos. En virtud de ello, al comparar dos soluciones u y v usando la relación de dominancia antes descrita, pueden haber tres posibles escenarios: 1) u domina a v ($u \prec v$), 2) u es dominada por v ($u \succ v$) o 3) u y v no se dominan entre sí. A continuación, se exponen las propiedades de relación de dominancia:

- *Reflexividad*: La relación de dominancia no es reflexiva, puesto que cualquier solución p no puede dominarse a sí misma.

- *Simetría*: La relación de dominancia no es simétrica, ya que $p \succ q$ no implica $q \succ p$. Por lo que el caso contrario es cierto, si p domina a q, entonces q no domina a p. Por tanto, la relación de dominancia es asimétrica.

- *Antisimetría*: Puesto que la relación de dominancia no es simétrica, tampoco puede ser antisimétrica.

- *Transitividad*: La relación de dominancia es transitiva. Esto es porque si $p \succ q$ y $q \succ r$, entonces $p \succ r$.

Otra noción importante dentro de la optimización multiobjetivo, misma que está directamente relacionada con el término de dominancia, es la **optimalidad de Pareto**. Como se mencionó en secciones anteriores, al contar con varias funciones objetivo el concepto de optimalidad cambia totalmente, ya que en los MOOP realmente se está buscando obtener buenos compromisos entre puntos encontradas en lugar de una única solución como en la optimización mono-objetivo. En este contexto, el concepto de óptimo de Edgeworth-Pareto, mejor conocido como óptimo de Pareto, fue propuesto inicialmente por Francis Ysidro Edgeworth (1881) en su trabajo relacionado con las curvas de indiferencia en economía a finales del siglo XIX y extendido en 1896 por Vilfredo Pareto. Podemos definir formalmente la optimalidad de Pareto como sigue: decimos que un vector de variables de decisión $\mathbb{x} \in \Omega$ es un óptimo de Pareto con respecto a Ω si y solo si no existe un $\mathbb{x}' \in \Omega$ tal que $v = F(\mathbb{x}') = (f_1(\mathbb{x}'), \dots, f_k(\mathbb{x}'))$ domine a $u = F(\mathbb{x}) = (f_1(\mathbb{x}), \dots, f_k(\mathbb{x}))$.

De esta forma, se sabe que para objetivos en conflicto no existe una única solución que domine al resto, pero si es posible que una solución no sea dominada por ninguna otra, a la que se denomina solución no dominada. Todas las soluciones no dominadas poseen el atributo de que su calidad no puede incrementarse con respecto a ninguna de las funciones objetivo sin afectar negativamente a otra de ellas. En presencia de restricciones, tales soluciones generalmente se encuentran en el borde de las regiones factibles del espacio de búsqueda. El conjunto de todas las soluciones no dominadas se le conoce como **conjunto de Pareto ($\mathcal{P}^*$)**, mientras que a la gráfica de las funciones objetivo cuyos vectores no dominados se encuentran en el conjunto de óptimos de Pareto se denomina **frente de Pareto ($\mathcal{PF}^*$).**

Para un problema multiobjetivo dado $f(\mathbb{x})$, el conjunto de óptimos de Pareto se define como:

$$\boldsymbol{\mathcal{P}}^* = \{\mathbb{x} \in \Omega | \neg \exists \mathbb{x}' \in \boldsymbol{\Omega} \ \boldsymbol{f}(\mathbb{x}') \prec \boldsymbol{f}(\mathbb{x})\}. \quad (11)$$

Las soluciones óptimas de Pareto son aquellas soluciones dentro del espacio de búsqueda de genotipos (espacio de decisión) cuyos componentes del vector objetivo del fenotipo correspondientes no se pueden mejorar simultáneamente. Estas soluciones también se denominan soluciones no inferiores, admisibles o eficientes, y pueden no tener ninguna relación aparente además de su pertenencia al conjunto óptimo de Pareto.

Por otra parte, al momento de graficar los vectores no dominados en el espacio objetivo, podemos definirlos colectivamente como frente de Pareto, definido formalmente como sigue: para un problema multiobjetivo dado $f(\mathbb{x})$ y un conjunto óptimo de Pareto $\boldsymbol{\mathcal{P}}^*$, el frente de Pareto ($\boldsymbol{\mathcal{PF}}^*$) se define como:

$$\boldsymbol{\mathcal{PF}}^* = \{u = f(\mathbb{x}) | \mathbb{x} \in \boldsymbol{\mathcal{P}}^*\}. \quad (12)$$

Por lo general, no resulta para nada sencillo encontrar una expresión analítica de la línea o superficie que represente los valores de los vectores no dominados en el espacio de las funciones objetivo y, en la mayor parte de los casos, resulta simplemente imposible. El procedimiento normal para generar un frente de Pareto es calcular todos los puntos factibles Ω y obtener sus valores correspondientes $f(\Omega)$. Cuando se cuenta con un número suficiente de estos puntos, es posible determinar los no dominados de entre ellos y así, definir un **frente artificial de Pareto ($\boldsymbol{\mathcal{PF}}$)**. En la **Figura 7**, se ilustra el frente de Pareto para dos objetivos en conflicto a minimizar.

Asimismo, una solución puede ser débilmente no dominada o bien, fuertemente (o estrictamente) no dominada, dichos conceptos pueden definirse formalmente como sigue: un punto $\mathbb{x}^* \in \Omega$ es una solución débilmente dominada si no existe $\mathbb{x} \in \Omega$ tal que $f_i(\mathbb{x}) < f_i(\mathbb{x}^*)$ para $i = 1, \dots, k$. Por el contrario, un punto $\mathbb{x}^* \in \Omega$ es una solución fuertemente no dominada si no existe $\mathbb{x} \in \Omega, \mathbb{x} \neq \mathbb{x}^*$ tal que $f_i(\mathbb{x}) \leq f_i(\mathbb{x}^*)$, para $i = 1, \dots, k$.

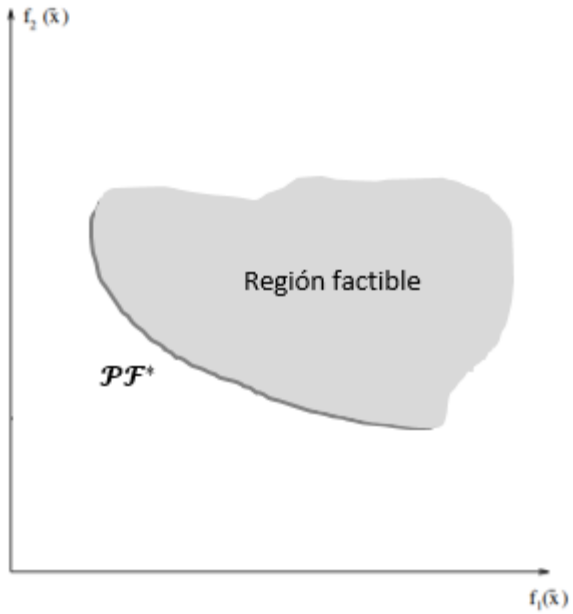


Figura 7. Frente de Pareto para dos objetivos en conflicto a minimizar.

3.3 Clasificación de técnicas de solución

Múltiples enfoques han sido propuestos en la literatura con el fin de poder dar frente a problemáticas con más de un criterio de optimización, los cuales van desde técnicas de búsqueda relativamente simples, pero altamente costosas en términos computacionales, como el evaluar cada posible solución factible dentro de algún espacio de decisión finito, realizando un ranking que permita seleccionar la solución con mejor rendimiento. O bien, métodos alternativos capaces de obtener una buena aproximación hacia el conjunto de Pareto, en lugar del óptimo como tal. En este sentido, es posible categorizar las técnicas de optimización en dos principales grupos: métodos exactos, mismos que garantizan la identificación de las mejores soluciones a expensas de un alto costo computacional, y las heurísticas, métodos estocásticos capaces de proveer "buenas" aproximaciones hacia las soluciones óptimas en tiempos razonables.

Sin embargo, dado que muchos problemas de optimización del mundo real son problemas con una alta dimensionalidad, discontinuos y/o NP-completos, se requiere de herramientas competentes que encuentren soluciones aceptables en tiempos razonables. Por ello, la implementación de métodos exactos para su resolución resulta inviable, por lo que diferentes enfoques estocásticos como recocido simulado, métodos de Monte Carlo, búsqueda tabú o algoritmos evolutivos, han sido ampliamente implementados

como técnicas alternativas para la resolución de estas problemáticas irregulares. Si bien se ha demostrado que estas eventualmente pueden llegar a alcanzar el caso óptimo para un problema dado, en la mayoría de las ocasiones esto no puede ser garantizado. Sin embargo, gracias a la facilidad de implementación, alta adaptabilidad, bajo consumo de recursos y velocidades de respuesta aceptables, han sido objeto de estudio durante décadas. Por lo general, las heurísticas son capaces de proporcionar buenas soluciones a una amplia gama de problemas de optimización que los métodos de búsqueda determinísticos tradicionales encuentran complicado.

No obstante, una aproximación por parte de estas técnicas debe satisfacer los siguientes dos criterios: convergencia y diversidad hacia el frente de Pareto. La convergencia hace referencia al hecho de minimizar la distancia entre el vector de soluciones no-dominadas identificadas y el frente de Pareto real, en otras palabras, se busca estar lo más cerca posible de las soluciones óptimas. Por otro lado, mediante la diversidad se busca obtener una muestra representativa del frente óptimo, esto implica una distribución aceptable de las soluciones del conjunto no dominado en el espacio objetivo. Sin embargo, dependiendo de la forma del espacio de la función objetivo, algunas regiones del frente pueden llegar a ser inalcanzables.

A continuación se presenta una de las clasificaciones más populares de técnicas de optimización multiobjetivo (Cohon y Marks, 1975) dentro de la comunidad de investigación de operaciones, la cual se basa en las aportaciones por parte del tomador de decisiones durante la búsqueda de una solución, que como bien su nombre lo indica, es el encargado de seleccionar cuál o cuáles de los vectores resultantes cuenta con una mejor compensación para cada uno de los objetivos en conflicto, capaz de satisfacer los requerimientos específicos propios del problema (conocimiento a *priori*), es decir, cuánto se está dispuesto a sacrificar en alguno de los criterios para ganar en otro de ellos. Asimismo, con el fin de facilitar la comprensión de la clasificación a presentar, se describen algunos de los métodos en investigación de operaciones más representativos, indicando cómo es que cada uno de ellos encaja dentro de estas categorías.

3.3.1 Métodos a *priori*

Este grupo de técnicas incluye aquellos enfoques de optimización que presuponen que un cierto conjunto de metas deseables (o un cierto pre-ordenamiento de los objetivos) pueden ser satisfechas por el tomador de decisiones antes de la realización de la búsqueda. Métodos representativos de esta categoría son el método del criterio global (Osyczka, 1984), ordenamiento lexicográfico (Fishburn, 1974), programación por metas (Charnes y Cooper, 1957) y optimización Min-Max (Solich, 1969).

Método del criterio global

En este método, el objetivo es minimizar una función definida como criterio global, la cual mide qué tanto puede acercarse un tomador de decisiones al vector ideal $\mathbf{f}^*$**,** definido formalmente (véase Capítulo 3 sección 3.2.1) como la solución óptima (factible) considerando a cada uno de los criterios de optimización del problema de manera aislada. La forma más común para evaluar esta cercanía es a través de la siguiente función (Osyczka, 1984):

$$f(\mathbb{x}) = \sum_{i=1}^{k} \left(\frac{f_i^* - f_i(\mathbb{x})}{f_i^*} \right)^p , \qquad (13)$$

donde k representa la cantidad de objetivos a optimizar. Sin embargo, la selección de la mejor p representa un problema en sí, ya que una mala elección de dicho valor puede resultar en soluciones inaceptables, múltiples autores sugieren un valor de $p = 1$ o $p = 2$.

Una variante del método del criterio global, denominada comúnmente como programación de compromisos, es la siguiente:

$$L_p(f) = \left[\sum_{i=1}^{k} |f_i^* - f_i(x)|^p \right]^{1/p} . \qquad (14)$$

Este método hace uso de métricas L_p, donde se busca básicamente minimizar el valor de L_p para $1 \leq p \leq \infty$ tal que $\mathbb{x} \in \Omega$, de tal manera que se puede obtener una solución óptima de Pareto para cada valor de p. Asimismo, Wierzbicki (1980) propone una variante más a esta técnica de optimización, donde la función global cuenta con una forma tal que penaliza las desviaciones con respecto a un punto de referencia dado. Por lo general, dicho punto de referencia corresponde al vector utópico $z^{**} \in \mathbb{R}^n$, el cual funge como un vector ficticio capaz de direccionar la búsqueda. Sin embargo, cualquier punto razonable o deseable en el espacio de las funciones objetivo puede elegirse como un objetivo de referencia.

Programación por metas

En este método, propuesto inicialmente por Charnes y Cooper (1957) e Ijiri (1965), el tomador de decisiones asigna metas que desea satisfacer por cada objetivo, donde estos valores se incorporan al problema como restricciones adicionales. La función objetivo intenta entonces minimizar las desviaciones absolutas con respecto a las metas establecidas para cada objetivo. La forma más simple del método es la siguiente:

$$\min \sum_{i=1}^{k} |f_i(\mathbb{x}) - T_i|, \ \mathbb{x} \in \Omega, \qquad (15)$$

donde T_i denota el conjunto de metas establecidas por el tomador de decisiones para la i-ésima función objetivo $f_i(x)$ y Ω representa la región factible o espacio de decisión. El criterio de optimización consiste básicamente en minimizar la suma de los valores absolutos de las diferencias entre los valores meta (o deseados) y los valores actualmente obtenidos. Una formulación más general de la programación por metas hace uso de una suma ponderada de las p-ésimas potencias de las desviaciones $|f_i(\mathbb{x}) - T_i|$, denominada comúnmente como programación por metas generalizada.

Método lexicográfico

En este método el tomador de decisiones debe ser capaz de asignar un orden de importancia (de mayor a menor) para todas las funciones objetivo a optimizar. La idea general consiste en buscar la mejor solución posible (u optima en algunos casos) para el primer objetivo del ranking y asignarla como restricción para el siguiente objetivo, asegurando así que la solución encontrada para el siguiente criterio no altere en lo absoluto la optimalidad previamente identificada. De esta manera, dicho proceso continúa así para cada uno de los objetivos hasta llegar al menos relevante.

Suponiendo que los subíndices de los objetivos indican no solo el número de la función objetivo sino también su prioridad. Se tiene entonces que, $f_1(\mathbb{x})$ y $f_k(\mathbb{x})$ denotan las funciones objetivo con mayor y menor prioridad de acuerdo al orden establecido por el tomador de decisiones, respectivamente. Por lo que el primer problema se formula como sigue:

$$\min f_1(\mathbb{x}) \tag{16}$$

sujeto a

$$g_j(\mathbb{x}) \leq 0; \quad j = \{1, 2, \dots m\}. \tag{17}$$

De esta forma se obtiene la solución $\mathbb{x}_1^*$ y $f_1^* = f(\mathbb{x}_1^*)$. Posteriormente, el segundo problema que se formula es como sigue:

$$\min f_2(\mathbb{x}) \tag{18}$$

sujeto a

$$g_j(\mathbb{x}) \leq 0; \quad j = \{1,2, \dots m\} \tag{19}$$

y

$$f_1(\mathbb{x}) = f_1^*. \tag{20}$$

Cabe destacar que la solución previamente identificada f_1^* se añadió como restricción en la obtención de la segunda solución $\mathbb{x}_2^*$ y $f_2^* = f(\mathbb{x}_2^*)$. Este procedimiento se repite hasta que se hayan considerado las k funciones objetivo. El i-ésimo problema está dado por:

$$\min f_i(\mathbb{x}) \tag{21}$$

sujeto a

$$g_j(\mathbb{x}) \leq 0; \quad j = \{1,2, \dots m\} \tag{22}$$

y

$$f_l(\mathbb{x}) = f_l^*, \quad l = \{1,2, \dots, i-1\}. \tag{23}$$

La solución identificada en la última iteración $\mathbb{x}_{\mathrm{k}}^*$ se toma como la solución resultante al problema dado. El ordenamiento lexicográfico corresponde a un método de ponderaciones en el que los valores de los pesos difieren considerablemente. No obstante, el hecho de involucrar al tomador de decisiones a tal grado puede llegar a representar un problema, ya que este puede no contar con los elementos suficientes como para priorizar adecuadamente. A pesar de ello, dada su extrema simplicidad este tipo de técnicas han sido ampliamente utilizadas por parte de grandes comunidades de investigación de operaciones durante décadas.

3.3.2 Métodos a *posteriori*

Estas técnicas no requieren información preliminar sobre las preferencias del tomador de decisiones o de algún modelo de preferencias humanas. Este grupo de técnicas engloba a algunos de los enfoques multiobjetivo más antiguos que existen. La idea principal consiste básicamente en presentar un conjunto no dominado de soluciones al tomador de decisiones y así, este seleccione uno o varios puntos con base en el conocimiento propio del problema. Los algoritmos más representativos de esta categoría son el método "$\boldsymbol{\epsilon}$ −constraint" (Haimes et al., 1971), la combinación lineal de pesos (Zadeh, 1963) o bien, los enfoques evolutivos.

Combinación lineal de pesos

De acuerdo con Zadeh (1963) es posible obtener un conjunto de soluciones no-dominadas a través de la resolución de un problema de optimización escalar, en la cual la función objetivo sea una suma ponderada de los componentes del problema multiobjetivo original. En otras palabras, dado un vector de pesos $\mathbb{w} =$

$(\omega_1, \dots, \omega_n)^T$ tal que $\omega_i \geq 0$ para todos los $i = 1, \dots, n$ y a su vez, $\sum_{i=1}^{n} \omega_i = 1$, es posible realizar una combinación lineal de los n objetivos a optimizar, buscando así resolver un problema multi-criterio mediante técnicas utilizadas en la resolución de problemas mono-objetivo, minimizando (o maximizando, según sea el caso) la suma ponderada de los componentes de la función vectorial objetivo $\mathbb{F}$. Es decir:

$$\min \sum_{i=1}^{k} \omega_i f_i(\mathbb{x}) \qquad (24)$$

sujeto a

$$\mathbb{x} \in \Omega. \qquad (25)$$

El conjunto de soluciones no dominadas puede generarse variando paramétricamente los pesos ω_i en la función objetivo. Nótese que los pesos no reflejan proporcionalmente la importancia relativa de las funciones objetivo como en otras técnicas, sino que sólo son factores que permiten generar el conjunto óptimo de Pareto al ser variados.

Una vez que se obtiene una solución, esta es revisada por el tomador de decisiones quien evalúa la viabilidad de seleccionarla o por el contrario, proceder a ajustar los valores de $\mathbb{w}$ y realizar el proceso nuevamente. Sin embargo, una limitante importante a considerar en la combinación lineal de pesos es su incapacidad para generar porciones cóncavas de un frente de Pareto, sean cualesquiera los pesos utilizados.

Método restricción – ϵ

El método de restricción épsilon o método de compensación, mejor conocido como "ϵ-constraint" en inglés, basa su funcionamiento en la linealidad de un problema de optimización multiobjetivo. La idea de esta técnica consiste en minimizar una función objetivo (la principal o preferida) a la vez, considerando al resto de los criterios como restricciones al problema, las cuales deben ser satisfechas dentro de ciertos niveles permisibles ϵ_l. De esta forma, dado un problema donde se busca minimizar todas sus funciones objetivo, este método obtiene soluciones no-dominadas resolviendo (Messac et al., 2003):

$$\min f_r(\mathbb{x}) \qquad (26)$$

sujeto a

$$f_l(\mathbb{x}) \leq \epsilon_l \text{ para } l = \{1, 2, \dots, k\} \text{ y } l \neq k, \qquad (27)$$

donde ϵ_l representan las cotas superiores definidas por los valores supuestos para cada una de las funciones objetivo. Por tanto, al variar estos valores ϵ_l y/o seleccionar diferentes funciones objetivo f_j a optimizar, es posible obtener las soluciones no-dominadas al problema dado. Un punto importante a destacar es la necesidad de un análisis preliminar para la identificación de los valores iniciales adecuados para ϵ_l.

Optimización evolutiva multiobjetivo

El cómputo evolutivo, también conocido como computación evolutiva, es un área de investigación multi disciplinaria para la optimización de objetivos mediante un proceso de prueba y error, misma que es fundamentada en el paradigma de la evolución artificial. En este enfoque, considerado comúnmente como un subcampo de la inteligencia artificial, se propone automatizar la resolución a una gran cantidad de problemas de una manera humana (Olague, 2016).

Las primeras ideas relacionadas al potencial de los algoritmos evolutivos para resolver problemas de optimización multiobjetivo o bien, ligadas al desarrollo de técnicas de optimización multiobjetivo que simulen el mecanismo de selección natural (supervivencia del más apto), se remontan a la tesis doctoral de Rosenberg en el año de 1960 "*Simulation of genetic populations with biochemical properties"*, donde se sugería utilizar múltiples propiedades (cercanía a alguna composición química especifica) en la simulación de una población de organismos unicelulares. Sin embargo, Resenberg realmente no presenta en ella una implementación multi-criterio como tal, ya que el problema bi-objetivo de su interés fue transformado a un problema mono-objetivo donde el objetivo adicional fue empleado como una restricción más.

No obstante, la primera implementación real de lo que ahora se conoce como algoritmos evolutivos para la resolución de problemas multi-criterio (MOEA: por sus siglas en inglés *"Multiobjective Evolutionary*

Algorithms") se le atribuye a David Schaffer, quien propuso el algoritmo genético de evaluación vectorial (VEGA, por sus siglas en inglés) en 1985. VEGA implementaba los tres operadores genéticos clásicos (selección, cruzamiento y mutación) de tal manera que iterativamente evolucionaban las soluciones haciendo uso de un criterio de selección basado en las funciones objetivo, por lo que para cada uno de los objetivos se realizaba una selección y posteriormente se aplicaba cruzamiento y mutación (Schaffer, 1986). Particularmente, en este algoritmo la población fue dividida aleatoriamente en subpoblaciones a las que luego se les asignó un valor *"fitness"* según una función objetivo diferente, pero la selección y la recombinación parentales se realizaron globalmente. Esta modificación fue suficiente para preservar una aproximación al frente de Pareto por algunas generaciones, pero no indefinidamente (Eiben y Smith, 2015). Después de esto, Goldberg (1989) sugirió el uso del *"fitness"* basado en dominancia en lugar de puntuaciones objetivas absolutas, junto con los métodos de *"niching"* o especiación para preservar la diversidad.

Los algoritmos evolutivos multiobjetivo son métodos a *posteriori* fundamentados en los principios de los algoritmos evolutivos para optimización mono-objetivo. Dichos métodos son estrategias particularmente atractivas para la resolución de MOOP dada su capacidad para tratar simultáneamente con un conjunto de posibles soluciones (llamada población), mismas que pueden pertenecer al conjunto óptimo de Pareto y es posible obtenerlas en una sola ejecución, esto a diferencia de métodos tradicionales donde las soluciones se alcanzarían en ejecuciones independientes. Además, los MOEA son menos susceptibles a la forma y continuidad del frente de Pareto, es decir, pueden operar con frentes cóncavos o discontinuos (Coello, 2018). Asimismo, estos cuentan con la capacidad de buscar en espacios parcialmente ordenados para compensaciones alternativas.

El Capítulo 4 está dedicado únicamente a los algoritmos evolutivos multiobjetivo, enfatizando en sus componentes principales y cómo es que estos interactúan entre sí, como la función objetivo o *"fitness",* representaciones de problemas, operadores genéticos, selección y reemplazo de padres, por mencionar algunos. Asimismo, se introducen los algoritmos genéticos de tipo celular (AGC), mismos que fueron implementados para la resolución del problema de enrutamiento de vehículos (definido en el Capítulo 2), objeto de estudio del presente trabajo.

3.3.3 Métodos interactivos

Estas técnicas normalmente operan en tres etapas, la primera de ellas consiste en obtener una solución no-dominada. Posteriormente, se obtiene la reacción del tomador de decisiones con respecto a dicha solución para así, modificar las preferencias de cada objetivo. Por último, se repiten los dos pasos anteriores hasta que el tomador de decisiones quede satisfecho o hasta que no sea posible obtener mayores mejoras (Coello et al., 2007). En esta categoría normalmente se incluyen técnicas adaptativas que dirigen su ejecución con respecto a una retroalimentación continua.

3.4 Estado actual de la optimización multiobjetivo

La gran mayoría de las técnicas de optimización multiobjetivo en la literatura de investigación de operaciones están limitadas a frentes de Pareto con ciertas características (p.ej., convexos), suelen requerir un punto inicial de búsqueda y suelen imponer restricciones adicionales (p.ej., que las funciones objetivo sean diferenciables). Adicionalmente, los algoritmos de programación matemática existentes suelen generar una sola solución por ejecución. De esta forma, se pueden identificar varias áreas promisorias de investigación dentro de optimización multiobjetivo (Miettinen, 1998; Coello et al., 2007; Coello, 2018):

1. La generación de conceptos más generales de optimalidad basados en preferencias que satisfagan ciertas condiciones que no se limiten a la reflexividad y la transitividad.

2. El desarrollo de nuevos algoritmos que sean más eficientes y que puedan generar soluciones compromiso para problemas más generales y requiriendo menos restricciones (p.ej., las metaheurísticas).

3. La implementación de más aplicaciones del mundo real que involucren problemas a gran escala (con muchas variables de decisión, funciones objetivo costosas de evaluar y procesos de toma de decisiones de alta complejidad).

Capítulo 4. Algoritmos evolutivos multiobjetivo

Este capítulo presenta los algoritmos evolutivos multiobjetivo (MOEA: por sus siglas en inglés *"Multiobjective Evolutionary Algorithms"*), enfatizando en sus principales componentes y cómo es que estos interactúan entre sí, entre ellos la función objetivo o *"fitness"*, representaciones de problemas, operadores genéticos, selección y reemplazo de padres, por mencionar algunos. Asimismo, diversos tópicos selectos son cubiertos con el fin de indagar en aquellas técnicas estocásticas bio-inspiradas propuestas en la literatura, así como técnicas representativas del estado del arte en cómputo evolutivo. Finalmente, se describe el algoritmo de optimización multiobjetivo con penalización ambiental propuesto para la resolución del problema de enrutamiento de vehículos, objeto de estudio del presente trabajo.

4.1 Introducción

El cómputo evolutivo es un área de investigación dentro de las ciencias de la computación, que como bien su nombre lo indica, son métodos de búsqueda estocásticos inspirados en la naturaleza que emulan el proceso de evolución de las especies para resolver problemas de optimización, el cómputo evolutivo propone técnicas como los **algoritmos genéticos (AG)**, **programación genética (PG)**, **estrategias evolutivas (EE)** y **programación evolutiva (PE)**, cada una de ellas con particularidades que las hacen útiles para cierto tipo de problemas. Estás técnicas evolutivas difieren principalmente en la forma en que representan a los individuos, por ejemplo, en los AG es común utilizar caracteres de un alfabeto finito, máquinas de estado finito para la programación evolutiva clásica o bien, representación con árboles en la programación genética.

Esta idea de emplear a la evolución natural como fuente de inspiración en la resolución de problemas complejos o bien, imitar a "solucionadores de problemas naturales" se remonta al año de 1948, cuando Alan Turing en su reporte titulado *"Intelligent Machinery"* identificó tres principales maneras en que la inteligencia de maquina está ampliamente relacionada con el objetivo de la programación de computadores para resolver problemas de manera automática, enfatizando en que la inteligencia de dicha maquina podría lograrse mediante el proceso de selección natural y evolución (Olague, 2016), es decir, a través de una búsqueda genética o evolutiva. Sin embargo, si bien Turing no especificaba cómo realizar o qué serie de pasos seguir para la ejecución de esta búsqueda genética, si permitió abstraer la idea de

aplicar principios darwinianos para la resolución de problemas de manera autónoma antes de la aparición de las computadoras como tal.

Dos décadas después, tres diferentes áreas de investigación estaban siendo desarrolladas tomando como base la idea general de Turing. En EEUU, L. Fogel, Owens y Walsh introducen la programación evolutiva y John Holland en colaboración con sus colegas y estudiantes de la Universidad de Michigan desarrollan los algoritmos genéticos. Mientras tanto, en Alemania, Rechenberg y Schwefel inventan las estrategias evolutivas. Si bien estas tres líneas de investigación continuaron por separado por más de 15 años, desde principios de la década de 1990 se les ha visto como pioneros de una tecnología emergente llamada cómputo evolutivo. A la par de dicho surgimiento, Koza propone un cuarto enfoque que seguía las ideas centrales, la programación genética.

No resulta complicado entender por qué gran cantidad de investigadores han optado por utilizar este tipo de técnicas, ya que el poder de la evolución en la naturaleza es evidente en las diversas especies que conforman el planeta, cada una diseñada minuciosamente para sobrevivir en su propio entorno. O bien, cómo especies relativamente "débiles" o con un número reducido de individuos, han perdurado a través de los años y a su vez, continúan mejorando con el pasar de las generaciones. La metáfora fundamental de la computación evolutiva relaciona esta poderosa evolución con un estilo particular de resolución de problemas: el proceso de prueba y error (Eiben y Smith, 2015).

Antes de introducir la idea subyacente detrás de estas técnicas, es importante destacar el concepto de selección natural propuesto por Darwin en 1859 en su obra *"El origen de las especies"*, donde prevalece la supervivencia del más apto y a su vez, se complementa con la necesidad de adaptación al cambio como método de supervivencia. Asimismo, otro concepto importante también proveniente del darwinismo, sugiere que, a través del uso de la herencia, nuevas generaciones tienen en su ADN toda la información inherente que ha permitido que dicha especie evolucione y sobreviva. El material genético heredado puede llegar a sufrir cambios o mutaciones ligado al proceso de prueba y error, generando así nuevas características y habilidades.

De esta forma, un algoritmo evolutivo (EA: por sus siglas en inglés *"Evolutionary Algorithm"*) en su configuración más general funciona con base a la siguiente analogía: dada una población de individuos en un entorno que tiene recursos limitados, la competencia por esos recursos causa la selección natural (supervivencia del más apto), lo que provoca a su vez un aumento en la condición física de la población. Asimismo, dada una función de calidad que debe maximizarse, es posible crear aleatoriamente un

conjunto de soluciones candidatas, es decir, elementos en el dominio de la función, para luego aplicar una función de calidad a estos como una medida de aptitud abstracta (Eiben y Smith, 2015). De la misma forma, la mayoría de los EA usan la recombinación, mezclando información de dos o más soluciones candidatas para crear nuevos individuos, así como un operador de mutación para fomentar la diversidad dentro de la población. A estos se les evalúa y generalmente compiten con los individuos más antiguos por un lugar en la próxima generación. Dicho proceso puede iterar hasta que se encuentre un candidato con calidad suficiente o bien, se alcance un límite computacional preestablecido (normalmente un número máximo de generaciones).

De tal manera, los algoritmos evolutivos resultan ser una herramienta útil en la resolución de problemas en los cuales la solución no puede ser obtenida de forma analítica o el tamaño del espacio de posibles soluciones es tan grande que un método determinístico no resulta factible. Por otro lado, dada su estrategia de búsqueda basada en poblaciones, es posible orientar la búsqueda en diferentes direcciones y así, escapar de óptimos locales para obtener un mejor rendimiento a la hora de aproximarse a la o las soluciones óptimas.

4.2 Componentes y principios básicos

El desarrollo de solucionadores de problemas autónomos es uno de los temas centrales de las ciencias de la computación y las matemáticas, donde se ha optado comúnmente por emular el comportamiento de *"solucionadores de problemas naturales"* dentro de estas herramientas. Al buscar al solucionador natural más poderoso, existen dos posibles candidatos: el cerebro humano y el proceso evolutivo. Si bien el primer candidato ha sido capaz de crear grandes invenciones como la rueda, grandes ciudades, guerras, etc., resultaría más conveniente tratar de diseñar al solucionador capaz de producir propiamente al cerebro humano, es decir, al proceso evolutivo. De tal forma, aquel conjunto de técnicas computacionales diseñadas a base de los principios básicos del darwinismo, se conoce como cómputo evolutivo.

Cualquier algoritmo evolutivo está compuesto por componentes basados en dos principios bien conocidos de la evolución: la selección natural y la herencia genética (Olague, 2016). De esta manera, resulta apropiado definir algunos conceptos desde un punto de vista microscópico. Según la definición de la Real Academia Española la genética es la *parte de la biología que trata de la herencia y de lo relacionado a ella*, misma que precisa de varios términos importantes análogos en el cómputo evolutivo. La genética aborda

temas relacionados con la herencia de características de una generación a otra y como esta información esta codificada en el ADN.

Según la observación fundamental de la genética molecular, es posible definir a un individuo como una entidad dual, es decir, un ente con propiedades fenotípicas representadas a un nivel genotípico. En otras palabras, el **genotipo** de un individuo codifica su **fenotipo** (véase **Figura 8**). Los **genes** por otra parte, son las unidades funcionales de la herencia que codifican las características fenotípicas. Por ejemplo, las propiedades visibles de un individuo como el color de cabello, el color de ojos o la estatura pueden ser determinadas por los genes. Siendo así, un genotipo se compone de uno o más cromosomas (filamentos de ADN), a su vez cada cromosoma está compuesto por un conjunto de genes separados, donde cada uno de ellos toma cierto valor (**alelos**) de algún alfabeto genético. Un **locus** identifica la posición de un gen dentro del cromosoma (véase **Figura 9**). Otro término importante es el **genoma**, el cual representa la información genética completa de un organismo, misma que está dispuesta en varios cromosomas; 46 para el caso de los humanos. Todos estos conceptos se ilustran en la **Figura 9** (para cromosomas binarios).

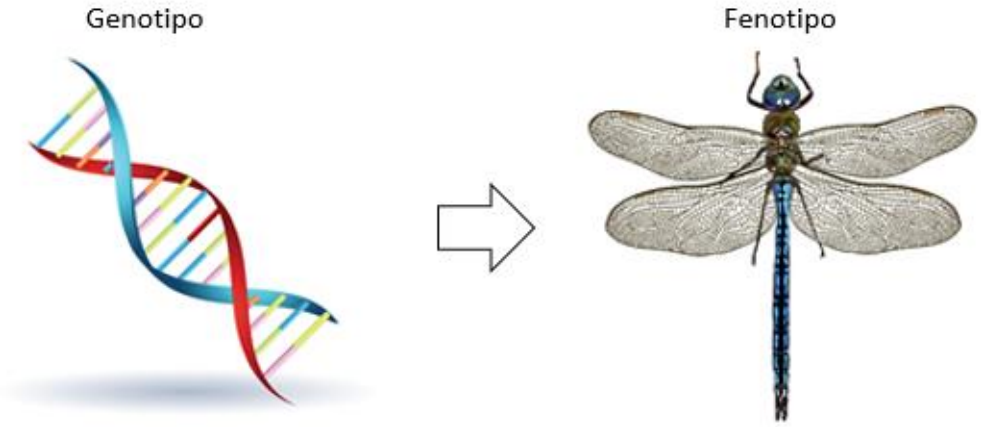


Figura 8. Representación de relación entre genotipo y fenotipo. El genotipo contiene toda la información inherente para construir un fenotipo en particular.

Las variaciones fenotípicas siempre son causadas por variaciones genotípicas, que a su vez son las consecuencias de las **mutaciones** de los genes, o de la **recombinación** de genes por la reproducción sexual o asexual (Coello et al., 2007). En cómputo evolutivo la combinación de características de uno o más individuos en su descendencia se le conoce como **cruzamiento**. No obstante, es importante tener en cuenta que esto no es análogo al funcionamiento de los organismos, donde la combinación de material

genético ocurre únicamente durante la formación de gametos (células sexuales), proceso al cual se le conoce como meiosis (Eiben y Smith, 2015).

De tal forma, dado un genotipo que puede ser modificado mediante diversos **operadores genéticos**, es posible generar una población con n cantidad de estos individuos, los cuales corresponden a una misma especie o bien, comparten el mismo hábitat. En este contexto, dichos operadores alterarán el material genético con el fin de fomentar la mejora en las próximas generaciones.

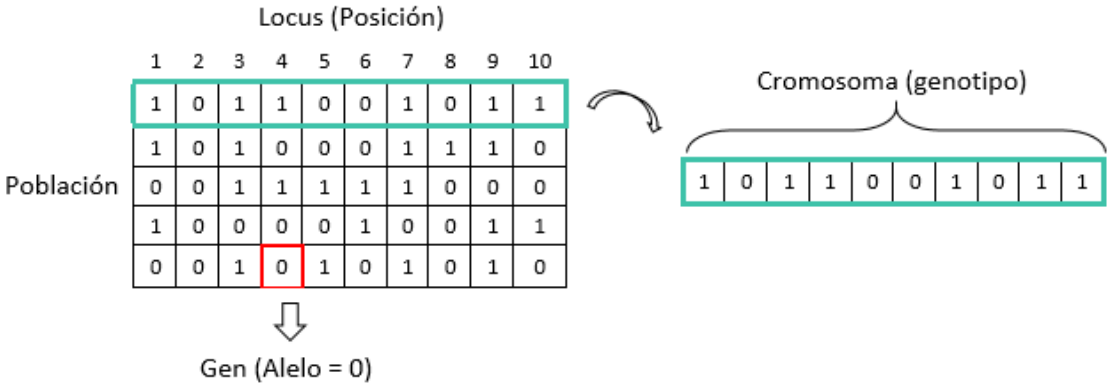


Figura 9. Estructura de datos y terminología utilizada en cómputo evolutivo. Analogía entre conceptos computacionales y su contraparte biológica.

4.2.1 Componentes principales

En esta sección se presentan a detalle los diversos componentes, procedimientos y operadores que conforman un algoritmo evolutivo, indagando en la relación que existe entre ellos y cómo su conjunción puede conllevar a la resolución de una gran cantidad de problemas complejos.

De acuerdo con Eiben y Smith (2015) si bien existen diversas variantes de los EA, la idea subyacente detrás de todas es la misma: Dada una población de individuos en un algún ambiente o contexto especifico con recursos limitados, la competencia por esos recursos causa la selección natural (supervivencia del más apto), lo que provoca a su vez un aumento en la condición física de la población. Asimismo, dada una función de calidad a maximizar (**función *"fitness"***), es posible crear aleatoriamente un conjunto de soluciones candidatas, es decir, elementos en el dominio de la función, para luego aplicar una función de calidad a todos ellos como una medida de aptitud abstracta (el valor más alto es el mejor individuo). Con base en estos valores de calidad, las mejores soluciones son seleccionadas como semillas para la siguiente

generación, a las cuales posteriormente se les aplica algún operador genético de recombinación y/o mutación. El proceso de recombinación se ejecuta en dos o más candidatos seleccionados (también conocidos como padres o semillas), produciendo así una o más posibles nuevas soluciones (hijos o descendencia). Por otro lado, la mutación tiene la capacidad de modificar cada posible solución generando nuevos candidatos.

De modo que, después de aplicar los operadores genéticos de recombinación y mutación se produce un conjunto de nuevas soluciones potenciales (descendencia), mismas que cuentan con un valor de aptitud que les permite competir por un lugar en la siguiente generación (supervivencia). Dicho proceso itera hasta que un individuo alcance la aptitud deseada (solución óptima) o bien, presente suficiente calidad como para ser considerado como solución. Adicionalmente, es posible preestablecer algún límite computacional que funja como criterio de parada, este puede ser un número máximo de generaciones, cantidad de memoria o CPU utilizada, entre otros.

Dos objetivos en conflicto intrínsecos son comunes en todas las búsquedas en los algoritmos evolutivos: exploración y explotación. Estos dos criterios forman la base de los sistemas evolutivos, donde un objetivo se logra solamente a expensas del otro. Para dar frente a ellos, dos tipos de operadores son implementados: los **operadores de variación** y de **selección**. Los primeros (recombinación y mutación) son los encargados de fomentar la diversidad necesaria dentro de la población. Por otra parte, el operador genético de selección actúa como una fuerza para aumentar la calidad media de las soluciones.

Algoritmo 1. Algoritmo evolutivo.

Inicio

Inicializar aleatoriamente la población ($\mathbb{X}$) con n soluciones candidatas (x). $x_1, x_2, \dots, x_n \in \mathbb{X}$.

Evaluar el desempeño de cada individuo dentro de $\mathbb{X}$ con base a una función de aptitud. $x_i \rightarrow f_{aptitud}(x_i), i = \{1, \dots, n\}$.

Mientras (Criterio de parada ≠ verdadero) **hacer**

Seleccionar j padres de $\mathbb{X}$. $\mathbb{P} = \{p_1, \dots, p_j\}$.

Recombinar pares de padres para generar k descendientes. $D = \{d_i, \dots, d_k\}$.

Mutar la descendencia resultante.

Evaluar el desempeño de los nuevos candidatos. $d_i = f_{aptitud}(d_i), i = \{1, \dots, k\}$.

Seleccionar individuos para la próxima generación $\mathbb{X}'$.

$\mathbb{X} \rightarrow \mathbb{X}'$.

Fin

Cabe señalar que gran parte de los componentes inmersos en este proceso evolutivo son estocásticos. Por ejemplo, durante el proceso de selección los individuos no son elegidos de manera determinista e incluso los individuos débiles cuentan con cierta posibilidad de convertirse en padres o de sobrevivir. Del mismo modo, durante la recombinación y mutación, la elección de qué genes serán tomados en cuenta se realiza al azar. A grandes rasgos, todas las técnicas que comprenden al cómputo evolutivo siguen este esquema, diferenciándose únicamente en detalles técnicos. Un ejemplo del algoritmo básico se muestra en el **Algoritmo 1**. De igual manera, la **Figura 10** ilustra la estructura general de un EA.

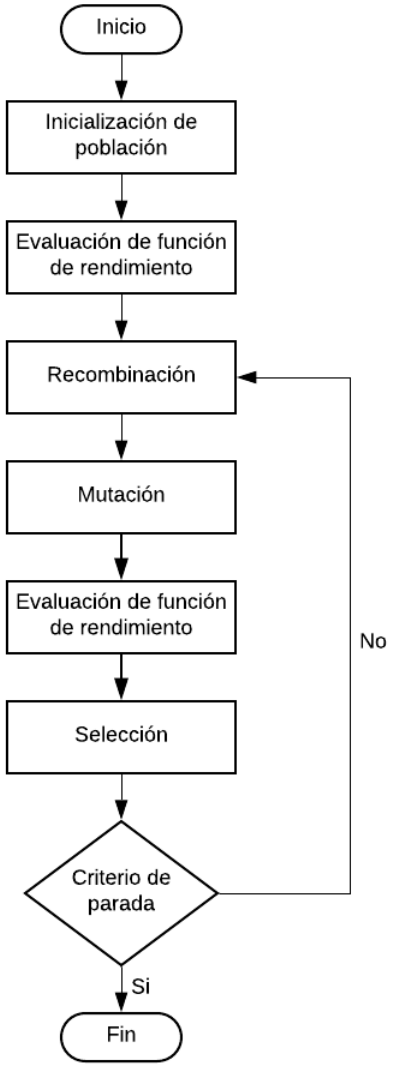


Figura 10. Flujo general de un algoritmo evolutivo.

Representación del individuo

Dentro de los EA, una estructura o individuo es una solución potencial a un problema en cuestión, normalmente representada como una cadena (o una cadena de cadenas) cuya analogía corresponde a un genotipo biológico. Este genotipo define un organismo individual cuando se expresa (decodifica) en un fenotipo.

Uno de los principales retos al momento de diseñar un EA, es precisamente representar el mundo real dentro del mismo algoritmo, lo que significa crear una estructura de datos que represente las características del problema y su contexto, la cual debe estar diseñada de tal manera que un sistema computacional pueda manipularla, en otros términos, toda la información inherente al problema y su contexto son los fenotipos y su codificación, crea una estructura de datos en un entorno de EA que contiene los genotipos (Eiben y Smith, 2015). De esta manera, la generación de una codificación que permita mapear adecuadamente del espacio de fenotipos al de genotipos es de vital importancia en el desempeño de un algoritmo evolutivo, ya que de ella dependerá si la búsqueda se ejecuta en todo el espacio de posibles soluciones o en un subconjunto de él.

Así, dada la naturaleza del problema a abordar, es posible definir qué tipo de técnica evolutiva implementar y más específicamente, que tipo de cromosoma utilizar. Como bien se ha mencionado anteriormente, los múltiples enfoques de EA existentes en la literatura presentan notables diferencias a la hora de codificar sus individuos, particularmente para los AG es común utilizar caracteres de un alfabeto finito, máquinas de estado finito para la programación evolutiva clásica o bien, representación con árboles en la programación genética.

Función de evaluación de aptitud

Un EA requiere tanto de una función objetivo como de una función de aptitud (función *"fitness"*), las cuales fundamentalmente son distintas. La función objetivo define la condición de optimalidad del EA, misma que es una característica del dominio del problema, mientras que la función *"fitness"* (en el dominio del algoritmo) mide qué tan bien una solución en particular satisface dicha condición de optimalidad y a su vez, le asigna un valor real (Coello et al., 2007). Cabe destacar que, en caso de no requerir ninguna modificación en la función *"fitness"* ambas funciones representan el mismo valor (Eiben y Smith, 2015).

El papel de la función *"fitness"* es representar los requisitos que la población debe cumplir para adaptarse idóneamente a su entorno, es decir, define el significado de mejora dentro del EA. O bien, visto desde la perspectiva de la resolución de problemas, indica la tarea a resolver en el contexto evolutivo. Básicamente, la función *"fitness"* es un procedimiento que asigna una medida de calidad a cada genotipo y su diseño es crucial para alcanzar la optimalidad deseada.

Población

En el contexto de los algoritmos evolutivos se denomina población a un multiconjunto de genotipos. Dado que los individuos como tal son objetos estáticos que no pueden evolucionar por si solos y necesitan la interacción con sus pares para lograr cambios o adaptarse, se considera a la población como la unidad del proceso evolutivo. De esta forma, dada una codificación, definir una población puede ser tan simple como especificar cuántos individuos hay en ella o definir su tamaño.

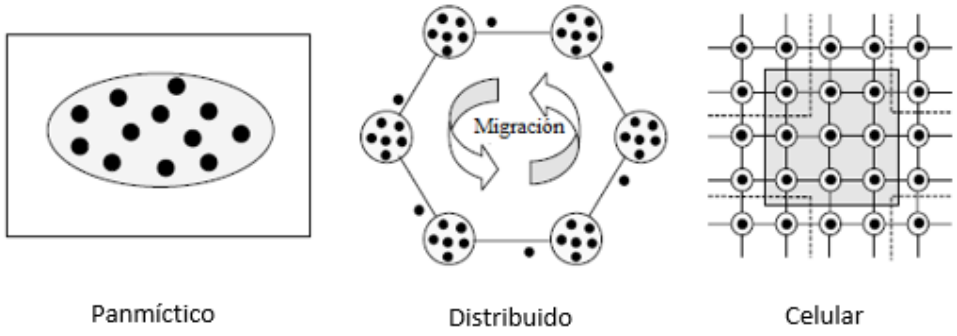


Figura 11. Tipos de población en EA.

En la mayoría de las aplicaciones de EA el tamaño de la población se mantiene constante a lo largo de toda la búsqueda evolutiva, lo cual incrementa la competencia debido a la necesidad de sobrevivencia en el recambio generacional. Una métrica común relacionada a este componente es la diversidad, la cual pretende cuantificar el número de diferentes soluciones presentes. Sin embargo, dicha medida puede hacer referencia a distintos factores como a la cantidad de individuos con distinto valor *"fitness"*, la cantidad de fenotipos o genotipos diferentes presentes en la población o bien, a medidas estadísticas como la entropía. Un punto importante a considerar es que la presencia de un único valor *"fitness"* dentro de la población no implica necesariamente que solo se cuente con un único fenotipo, ya que diferentes genotipos pueden llegar a contar con un mismo valor de aptitud.

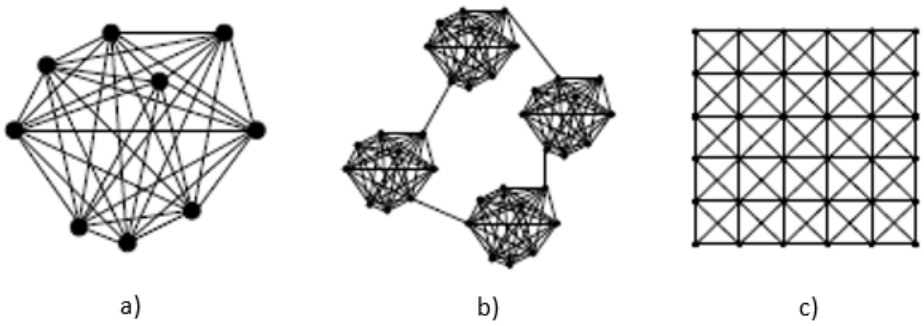


Figura 12. Grafo de conectividad entre individuos en una población a) panmíctica, b) distribuida y c) celular (Alba y Dorronsoro, 2008).

Mantener una buena diversificación de los individuos dentro de la población a lo largo de la ejecución del algoritmo evolutivo es un factor clave para lograr buenos resultados, ya que de ello depende directamente que el algoritmo escape de óptimos locales, o bien, dicho en términos técnicos, explote y explore adecuadamente el espacio de posibles soluciones. Por ello, diversos tipos de poblaciones han sido propuestos en la literatura, entre los cuales se encuentran: poblaciones panmícticas, distribuidas o celulares. En implementaciones clásicas en común utilizar poblaciones de tipo panmíctico, en donde sus individuos pueden interactuar sin restricción alguna. Sin embargo, en algoritmos más sofisticados, típicamente se utilizan poblaciones distribuidas en islas, donde cada una de ellas evoluciona de manera independiente e intercambian material genético cada cierta cantidad de generaciones. Por otro lado, como enfoque hibrido, se presentan las poblaciones de tipo celular, donde la distribución se da por medio de vecindarios definidos dentro de una única población, donde se limita el número de individuos con los que cada solución puede interactuar y de igual forma, cada cierto tiempo definido se intercambia información genética, emulando así la migración de individuos entre distintos ecosistemas (véase **Figuras 11 y 12**).

Selección de padres

La operación de selección de padres imita el proceso darwiniano de selección natural para crear poblaciones a través de generaciones, para una revisión del tema se recomienda revisar el libro de Olague (2016). Dicha operación consiste básicamente en permitir que los mejores individuos intercambien material genético para generar descendencia. Junto con el mecanismo de selección de sobrevivientes o reemplazo, la selección de padres es responsable de impulsar la mejora en la calidad de las soluciones. En cómputo evolutivo, esta selección es comúnmente probabilística, por lo que individuos con un alto valor

"fitness" tienen más posibilidades de procrear que aquellos con baja calidad. Sin embargo, con el fin de evitar estancamientos en óptimos locales, a menudo a individuos con bajo rendimiento se les da oportunidad (pequeña en la mayoría de los casos) de ser seleccionados, de no ser así la diversidad en la población se vería afectada y la búsqueda se reduciría a un enfoque voraz.

Ejemplos de técnicas de selección propuestas en la literatura son: selección por torneo o selección por ranking. La selección por torneo opera aleatoriamente eligiendo q individuos de la población generacional y seleccionando a los mejores de ellos para sobrevivir en la próxima iteración. La selección por ranking asigna probabilidades de selección únicamente en la clasificación de un individuo, ignorando por completo los valores *"fitness"* absolutos. Otras dos técnicas de selección son las estrategias $(\mu + \lambda)$ y (μ, λ), donde μ representa el número de soluciones que fungirán como padres y λ la cantidad de descendientes resultantes.

Mutación

Los operadores genéticos imitan el proceso de herencia de los genes para crear nuevos descendientes en cada generación. En el contexto de los EA, esto equivale a generar nuevas soluciones candidatas a partir de las más antiguas. El uso de los diferentes operadores genéticos juega un papel preponderante en el rendimiento de cualquier enfoque evolutivo (Gen et al., 2008). Estos operadores comúnmente se dividen en dos principales ramas según su aridad: unarias (mutación) y n-arias (recombinación).

El operador genético de mutación es un operador unario que modifica ligeramente el material genético de un genotipo. Esta variación es totalmente estocástica: su salida depende directamente de los resultados de una serie de elecciones aleatorias e imparciales (Eiben y Smith, 2015). Asimismo, al igual que otros componentes clave de los EA, la manera en que se implementan estos operadores genéticos está directamente relacionada con la técnica evolutiva utilizada. Por ejemplo, en la programación evolutiva es el único operador utilizado, mientras que en los algoritmos genéticos se ha considerado tradicionalmente a dicho proceso como una operación de fondo, capaz de aportar nuevo material genético a la población, e incluso se puede llegar a prescindir de ello, como bien sucede en la programación genética.

En un sentido teórico, la mutación permite garantizar que el espacio de búsqueda está conectado y por lo tanto, repercute en la factibilidad de alcanzar la optimalidad, ya que la generación de un descendiente es equivalente a pasar de un punto a otro dentro de este espacio. En este sentido, si se cuenta con un espacio conexo donde cada genotipo representa una posible solución, la forma más sencilla de satisfacer la

optimalidad para un problema dado es permitir que el operador de mutación salte a todas partes: por ejemplo, al permitir que cualquier alelo pueda mutar a cualquier otro con una probabilidad mayor a cero ($p_m > 0$) (Eiben y Smith, 2015).

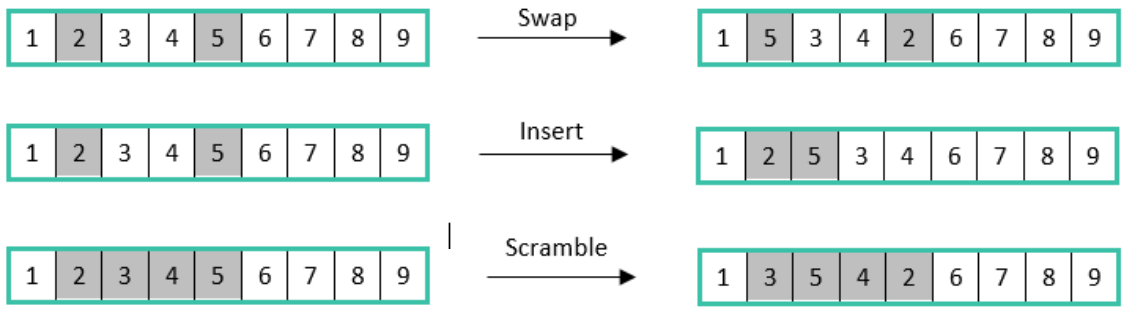


Figura 13. Mutación *"swap"* (arriba), *"insert"* (en medio) y *"scramble"* (abajo) para un cromosoma definido como una permutación.

Para el caso particular de representaciones de permutaciones no es posible considerar cada gen de forma independiente como bien sucede para otras codificaciones, ya que es necesario generar mutaciones que permitan únicamente mover alelos a lo largo del cromosoma. El método de mutación conocido como *"swap"*, propone la selección aleatoria de dos posiciones (genes) en el cromosoma y el intercambio de valores entre ellos. Otro ejemplo es la mutación *"insert"*, la cual selecciona dos alelos al azar y el segundo de ellos se mueve al lado del primero. Asimismo, otro de los operadores de mutación comúnmente utilizados en la literatura es la mutación *"scramble",* la cual simplemente intercambia todos o algún subconjunto de genes dentro del cromosoma (véase **Figura 13**). Básicamente, estos operadores funcionan realizando pequeños cambios en el orden en que aparecen los valores.

Recombinación

El operador genético de recombinación o cruzamiento es un operador binario, que como bien su nombre lo indica, combina información genética de dos genotipos en uno o más descendientes (hijos). A pesar de su naturaleza estocástica, este proceso busca distribuir los genes "*buenos*" de cada uno de los padres en los nuevos descendientes o bien, garantizar que cualquier información llevada en ambos padres esté presente en estos nuevos cromosomas.

Al igual que en el operador de mutación, el papel de recombinación difiere entre las distintas técnicas evolutivas. Por ejemplo, en la programación genética usualmente es el único operador de variabilidad implementado, mientras que en los algoritmos genéticos es considerado el operador de exploración principal; incluso se puede llegar a prescindir de él, como bien sucede en la programación evolutiva. La mayoría de estos métodos están restringidos al uso de dos padres, es decir con una aridad igual a dos, aunque se ha demostrado que el uso de multi-padres, así no se tenga una contraparte biológica, provee efectos positivos en la evolución (Eiben y Smith, 2015).

Es importante recordar que los operadores de variabilidad dependen directamente del tipo de representación implementado, por lo que para distintas codificaciones deben definirse diferentes operadores de recombinación. En el presente trabajo de investigación nos enfocaremos en los métodos utilizados para algoritmos genéticos con permutaciones de enteros como representación de individuos.

En este sentido, las representaciones basadas en permutaciones presentan dificultades particulares para el diseño de operadores de recombinación, ya que generalmente no es posible simplemente intercambiar subcadenas entre padres y aun así mantener las propiedades inherentes de una permutación. Por lo tanto, se han propuesto múltiples operadores genéticos especializados de cruce, que tienen como objetivo primordial transmitir la mayor cantidad posible de información genética contenida en los padres y, de cierta forma dar frente al inconveniente previamente mencionado, entre ellos se encuentran PMX ("*Partially Mapped Crossover*"), OX ("*Order Crossover*"), CX ("*Cycle Crossover*"), EX ("*Edge Crossover*"), PX ("*Position Crossover*"), cruzamientos heurísticos, entre muchos otros (Gen et al., 2008).

OX (Lawrence, 1991) es uno de los operadores de cruce mayormente utilizados en la literatura para codificaciones de permutaciones dada su capacidad para transmitir información en común de padres a descendientes. Dicho operador funciona de la siguiente manera (véase **Figura 14**) (Eiben y Smith, 2015):

1. Dados dos padres P_1 y P_2, se eligen dos puntos de cruce al azar y se copia el segmento entre ellos desde el primer padre (P_1) a la primera descendencia.

2. A partir del segundo punto de cruce en el segundo padre (P_2), se copian los números restantes no utilizados en el primer hijo en el orden en que aparecen en el segundo padre. Se genera la segunda descendencia de manera análoga, con los roles invertidos para P_1 y P_2.

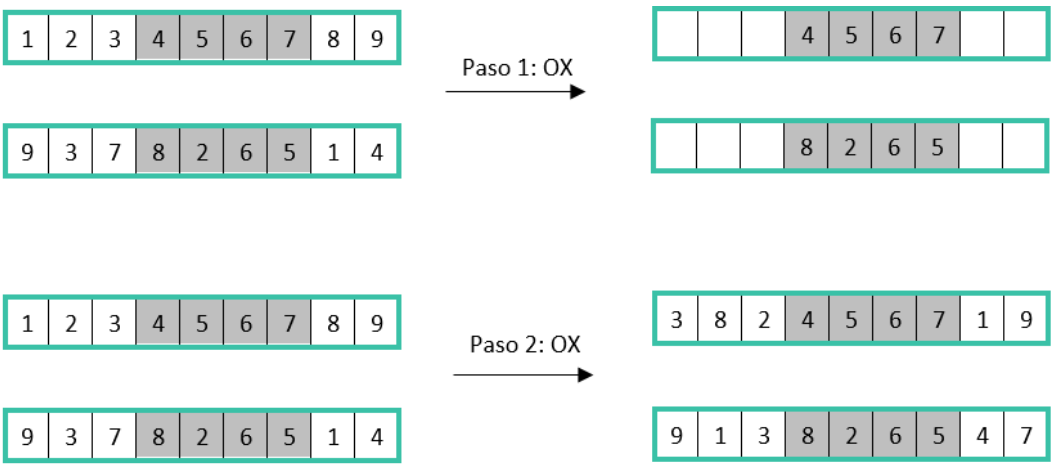


Figura 14. Ejemplo de "*Order Crossover*" para representación con permutación de enteros, donde los puntos de cruce son 3 y 7 para $n = 9$, donde n representa lo longitud del cromosoma.

Dentro de cualquier codificación con permutaciones el espacio de búsqueda es $(n-1)!$, donde dicho proceso puede corresponder a dos principales enfoques: basados en orden (OX, CX, etc.), en las cuales se busca preservar la mayor cantidad posible de información acerca de la posición absoluta en que los elementos aparecen o bien, basados en adyacencias (PMX, EX, etc.). Se recomienda revisar (Eiben y Smith, 2015) para mayor información acerca de métodos adicionales de recombinación para representaciones con permutación de enteros.

Reemplazo

Al igual que en la selección de padres, en el proceso de reemplazo o selección de sobrevivientes se distingue entre los individuos según su calidad para decidir cuáles de ellos deben dejar la población y así dar paso a nuevo material genético. La diferencia yace en que este sucede en una etapa distinta del ciclo evolutivo: el mecanismo de selección de sobrevivientes se llama después de la generación de descendencia por parte de los padres seleccionados.

En contraste con la selección de padres, la cual suele ser estocástica, el proceso de reemplazo es a menudo determinístico, considerando los valores de la función *"fitness"* para clasificar al multiconjunto unificado de progenitores y descendientes para así seleccionar al segmento superior. De igual forma, otro factor comúnmente utilizado es el concepto de edad, es decir, considerar la antigüedad generacional para

determinar a qué individuos se les permitirá o no formar parte de la próxima generación en el proceso evolutivo.

Criterio de parada

Dada la naturaleza estocástica de los algoritmos evolutivos, no hay garantías de optimalidad global para cualquier problemática a tratar. Bajo esta premisa, con el propósito de evitar que un algoritmo de esta índole se ejecute de manera indefinida ante la posibilidad de un estancamiento en un óptimo local, es necesario definir alguna condición capaz de detener su ejecución, previniendo así un alto costo computacional innecesario. Alternativas comúnmente utilizadas para dicho propósito son: definir un número máximo de generaciones, un límite de evaluaciones de la función *"fitness"*, acotar la cantidad de memoria o CPU utilizada o bien, preestablecer un umbral para identificar cuándo el algoritmo se ha atascado en un mismo valor de calidad durante un periodo determinado, o cuándo la diversidad de la población ha caído debajo de dicho umbral. En términos generales, se busca garantizar la detención oportuna del algoritmo ante el desconocimiento o incapacidad para alcanzar un resultado óptimo.

4.3 Algoritmos evolutivos para optimización multiobjetivo

Como se mencionó anteriormente, los algoritmos evolutivos multiobjetivo están fundamentados en los principios de los algoritmos evolutivos para optimización mono-objetivo, extendiendo en dos aspectos principales: los mecanismos de selección y conservación de diversidad. En lo que respecta al proceso de selección, a diferencia de un EA tradicional (véase **Algoritmo 1**) donde se busca una única solución que maximice o minimice un valor de aptitud directamente relacionado con una única medida de calidad subyacente, en los MOEA el objetivo es seleccionar un conjunto de soluciones no-dominadas dentro de la población (considerándolas como soluciones de igual calidad), las cuales permitan generar buenos compromisos entre dos o más objetivos en conflicto. En el proceso de conservación de diversidad, la diferencia yace principalmente en el bloqueo parcial del mecanismo de selección para evitar estancamientos en óptimos locales dado el ruido estocástico intrínseco, posibilitando así la generación de varios elementos del conjunto óptimo de Pareto en una única ejecución.

Básicamente, dado que en los problemas de optimización multiobjetivo la calidad de una solución está dada por su desempeño en relación con varios objetivos en conflicto, se desea que un MOEA genere

soluciones para problemas de esta índole ofreciendo un buen compromiso de desempeño para dos o más objetivos específicos (costo/beneficio, restricciones, etc.) comprometidos entre sí. Por ejemplo, en el problema multiobjetivo de enrutamiento de vehículos, una solución que optimice la congestión del tráfico se verá afectada en la distancia a recorrer para alcanzar la posición destino, es decir, para objetivos en conflicto, la optimalidad de uno de ellos afectará directamente el desempeño en el resto. Por este motivo, científicos, ingenieros, gobiernos e industrias ciertamente se han centrado en la investigación e implementación de este tipo de técnicas en problemas reales que impliquen la optimización de múltiples criterios.

En general, los MOEA se diseñan teniendo en cuenta dos objetivos principales:

- **Converger** al frente de Pareto, conservando en todo momento los puntos no-dominados en el espacio objetivo y los puntos asociados en el espacio de decisión (elitismo).

- **Diversificar** la población en lugar de converger a una sección reducida del frente de Pareto, permitiendo así proveer al tomador de decisiones una cantidad "suficiente", pero limitada de puntos en el frente.

Siendo así, es posible definir un buen "*trade-off*" por parte de un algoritmo evolutivo multiobjetivo como la generación de un conjunto de soluciones no-dominadas que permitan converger hacia el frente de Pareto y que, a su vez, estén propagadas a lo largo del mismo. No obstante, en la ejecución alcanzar estos dos criterios implica la aplicación de diversas técnicas capaces de direccionar adecuadamente la búsqueda y fomentar tanto la exploración como la explotación del espacio de posibles soluciones.

Con respecto a la preservación de la diversidad en un MOEA, una amplia gama de métodos ha sido propuesta en la literatura especializada. Entre ellos se encuentran "*niching*", los esquemas geográficos, el uso de entropía, coordenadas paralelas, entre muchos otros. La idea central detrás de todos estos métodos es penalizar las soluciones que se encuentren cerca unas de otras, ya sea en el espacio de decisión o en el espacio objetivo.

Otro componente primordial dentro de los MOEA modernos es el elitismo, el cual normalmente consiste en utilizar un archivo externo (también llamado población secundaria) que puede o no interactuar con la población principal durante el proceso de selección. El propósito de este archivo es almacenar todas las soluciones no-dominadas generadas a lo largo del proceso evolutivo. Por tanto, la aproximación al

conjunto de Pareto resultante es el contenido final de dicho archivo. Un punto importante a enfatizar es que el uso del elitismo no solo es aconsejable, sino también necesario, ya que se ha demostrado teóricamente que este componente es indispensable para garantizar la convergencia de un MOEA al conjunto óptimo de Pareto. En la práctica los archivos externos normalmente están limitados a un número máximo de soluciones, ya que de no ser así el archivo podría aumentar demasiado su tamaño, diluyendo en gran medida la presión de selección, lo que tiene un efecto negativo en el rendimiento de cualquier técnica de optimización.

Asimismo, cuantificar el desempeño de las soluciones bajo múltiples funciones objetivos es una tarea clave en el desarrollo de estos enfoques evolutivos multi-criterio. De tal forma, Goldberg (1989) introduce el término de dominancia (véase Capitulo 3) y "*niching*" (operador utilizado para controlar la diversidad de la población) con el propósito de subdividir la población en grupos en los que no es posible definir cuál es mejor para todos los objetivos en conflicto. Dicho de otra forma, propone el uso de conjuntos no-dominados como metodología para guiar la búsqueda y evolución en un MOEA. Lograr el frente óptimo de Pareto de un problema arbitrario suele ser una tarea difícil en la mayoría de los casos, por lo que se ha optado comúnmente por alcanzar buenas aproximaciones en tiempos computables. En este contexto, si bien las metodologías basadas en EA presentan un buen rendimiento para una gran cantidad de problemas, de acuerdo con el "*No free lunch theorem (NFL)"* (Wolpert y Macready, 1997) no es posible considerarlas como métodos de resolución universal.

En términos generales, el NFL asegura que, al comparar el rendimiento de diferentes algoritmos para todos los problemas posibles, en promedio todos presentarán resultados iguales. Es decir, al generar un nuevo algoritmo capaz de resolver algún problema en específico de manera óptima, este seguramente presentara un desempeño ineficiente en algunos otros (Eiben y Smith, 2015). No obstante, los MOEA poseen una implementación sencilla que permite guiar la búsqueda con información obtenida en el dominio del problema, lo que acelera su desarrollo, comprensión y evaluación. Además, estos cuentan con la capacidad para identificar soluciones de alta calidad en espacios de búsqueda de alta dimensionalidad con características complicadas, múltiples restricciones y discontinuidades. Cuando se combina con su naturaleza basada en poblaciones y su capacidad para encontrar y conservar diversos conjuntos de buenas soluciones, no es sorprendente que los MOEA sean actualmente el estado del arte en muchos MOOP.

4.3.1 Clasificación de los MOEA

Gran cantidad de publicaciones relacionadas con los MOEA pueden ser encontradas en la literatura, donde usualmente se proponen variantes de los enfoques ya existentes o bien, se intenta dar frente a un problema en particular con un algoritmo relativamente nuevo. A grandes rasgos, es posible enmarcar todos estos enfoques en dos generaciones marcadas por el uso de un mecanismo de selección conocido como elitismo, donde su uso e implementación es la diferencia principal entre los algoritmos que forman parte de la segunda generación con respecto a la primera. La inclusión de este proceso proporciona un rendimiento no degradante, dado que mantiene los mejores individuos durante cada recambio generacional.

Recientemente, dadas las múltiples aportaciones en el área, se ha optado por categorizar a este tipo de métodos evolutivos multiobjetivo en tres grandes ramas, cada una de ellas destacando su operatividad intrínseca, estas son: enfoques basados en dominancia, en descomposición y en indicadores. La primera categoría incluye aquellos algoritmos basados en el concepto propuesto por Goldberg en 1989 (véase Capitulo 3) como: *Strength Pareto Evolutionary Algorithm (SPEA, SPEA2)*, *Nondominated Sorting Genetic Algorithm (NSGA, NSGA-II)*, *Pareto Envelope-based Selection Algorithm (PESA, PESA II)*, entre algunos otros algoritmos populares de segunda generación. Por otra parte, los enfoques basados en descomposición como MOEA/D ("*Multi-Objective Evolutionary Algorithm Based on Decomposition*"), presentan un mejor desempeño con respecto a los basados en dominancia para problemas con más de cuatro objetivos en conflicto. Estas técnicas funcionan a través de la segmentación de un MOOP en diversos problemas con un único objetivo y resolviéndolos de manera simultánea. Por último, los enfoques basados en indicadores como bien su nombre lo indica, adoptan una medida de desempeño en el mecanismo de selección de padres o sobrevivientes. Un claro ejemplo de ello es HyPE ("*Hipervolume Estimation Algorithm for Multi-Objective Optimization*"), el cual propone un método rápido de búsqueda a través de simulaciones de Monte Carlo para aproximar los valores exactos del hipervolumen, cuya premisa yace en que el valor real del indicador no es lo más importante, sino la jerarquización de las soluciones que este produce.

En el contexto de los enfoques basados en dominancia, de acuerdo con Coello et al. (2007) existen tres posibles métodos que hacen uso de este concepto para hacer ordenamiento. Recordemos que la dominancia permite relacionar dos soluciones mediante un operador binario, es decir, dada una solución A que domina o no a una solución B, se tiene $domina(A, B) = 1$ o $domina(A, B) = 0$, respectivamente. Estos métodos son los siguientes:

- ***Ranking por dominancia:*** en este método se define un nivel para cada una de las soluciones pertenecientes a la población actual, esto mediante el cálculo de la cantidad de individuos por los cuales está dominada. Algoritmos representativos: "*Multi-Objective Genetic Algorithm*" (MOGA) y "*Niched Pareto Genetic Algorithm*" (NPGA).

- ***Dominancia por conteo:*** método basado en la cuantificación de las posibles soluciones dominadas por algún individuo. Algoritmos representativos: "*Strength Pareto Evolutionary Algorithm*" (SPEA y SPEA2)

- ***Dominancia por profundidad***: método basado en la distribución de los individuos en diversos *frentes de dominancia*, de tal manera que al ordenarlos el primer frente contenga las mejores soluciones, es decir, aquellas soluciones no-dominadas. Algoritmos representativos: "*Nondominated Sorting Genetic Algorithm*" (NSGA y NSGA-II).

Tabla 1. Comparativa de dos algoritmos evolutivos multiobjetivo representantes del estado del arte (NSGA-II y SPEA2) con respecto al algoritmo genético celular MOCell, objeto de estudio en este trabajo de investigación.

	MOCell	**NSGA-II**	**SPEA2**
Función de aptitud	Clasificación por no-dominancia y distancia de aglomeración	Clasificación por no-dominancia y distancia de aglomeración	Clasificación basada en información de densidad
Representación	{0,1} o $\mathbb{R}$	{0,1} o $\mathbb{R}$	{0,1} o $\mathbb{R}$
Mecanismo de diversidad	"*Crowding distance*" o distancia de aglomeración	"*Crowding distance*" o distancia de aglomeración	Mecanismo de truncamiento llamado "*método de enlace promedio*"
Mecanismo de elitismo	Si	Si	Si
Población externa	Si	No	Si
Ventajas	Rápida convergencia	Ampliamente utilizado	Buen desempeño en espacios objetivos con alta dimensionalidad
Desventajas	Poco popular	Bajo rendimiento en espacios de búsqueda con alta dimensionalidad.	Alto costo computacional
Método de dominancia	Dominancia por conteo	Dominancia por profundidad	Dominancia por conteo
Soporte	JMetal	JMetal, MOEA Framework	JMetal, MOEA Framework

Si bien ha habido muchos enfoques para la optimización multiobjetivo utilizando algoritmos evolutivos, solo un puñado de ellos en realidad son utilizados por un número significativo de investigadores y profesionistas de todo el mundo. Siendo así, en este trabajo de investigación se decide utilizar una versión adaptada de MOCell (*MultiObjective Cellular genetic algorithm*) (Nebro et al., 2009) como propuesta inicial y comparar su desempeño con respecto a dos algoritmos evolutivos multiobjetivo representantes del estado del arte ampliamente utilizados en problemas reales, NSGA-II (Deb et al., 2002) y SPEA2 (Zitzler et al., 2001). A continuación, se analizan a detalle cada uno de los algoritmos implementados en este trabajo. Asimismo, la **Tabla 1** presenta una comparativa de estos algoritmos, enfatizando en los componentes principales de un enfoque evolutivo.

4.3.2 Nondominated Sorting Genetic Algorithm II (NSGA-II)

El algoritmo de optimización multiobjetivo NSGA en su primera versión fue propuesto por N. Srinivas y Kalyanmoy Deb en 1994 como una variante más del enfoque de ordenamiento presentado originalmente por Goldberg. Este algoritmo se basa principalmente en el uso de varias capas de clasificación de los individuos, de tal manera que aquellos no-dominados con respecto al resto de la población son temporalmente removidos y dotados de una función de aptitud ficticia, esto con el fin de proporcionar un potencial reproductivo igual para todos ellos. Subsecuentemente, se obtienen los individuos no-dominados en la población restante, es decir, aquellos pertenecientes a la segunda capa, para luego asignarles una función de aptitud ficticia menor a la asignada a los individuos de la capa anterior. Posteriormente este grupo ya clasificado se ignora y se procede a generar una nueva capa; dicho proceso continúa hasta que todos los individuos hayan sido clasificados. Sin embargo, además de su alto costo computacional, en la práctica este enfoque multiobjetivo presentó un desempeño pobre con respecto a algoritmos representativos de primera generación, es decir, MOGA y NPGA.

Por ello, en 2002 Deb et al. proponen una versión mejorada del algoritmo NSGA, llamado NSGA-II, mismo que actualmente es uno de los MOEA más populares en la literatura. NSGA-II hace uso del principio de elitismo (MOEA de segunda generación) así como de la relación de dominancia en profundidad, además de un mecanismo capaz de preservar la diversidad conocido como "*crowding distance*" o distancia de aglomeración (véase **Figura 15**). Como se ilustra en la **Figura 16**, el algoritmo combina la población actual (P_t) de tamaño n con la descendencia (Q_t) resultante en cada iteración t (n número de hijos), obtenidos mediante el uso de los operadores genéticos tradicionales (selección, recombinación y mutación). De esta

forma, se cuenta con una nueva población R_t de tamaño $2n$, misma que es clasificada en diversos frentes no-dominados. Posteriormente, se calcula la distancia de aglomeración en cada uno de ellos con la finalidad de clasificar a las soluciones de un mismo frente de acuerdo a su aportación a la diversidad del frente en curso, permitiendo así mantener una población con alta diversidad y facilitar la exploración en el espacio de posibles soluciones; por tanto, cada uno de los individuos cuenta con un vector con k valores de aptitud, uno para cada criterio de optimización, un frente asociado y un valor "*crowding distance*". Por último, se construye una nueva población P_{t+1} seleccionando a los individuos de mayor a menor calidad, es decir, aquellos que pertenecen a los mejores frentes y que aportan mayor diversidad.

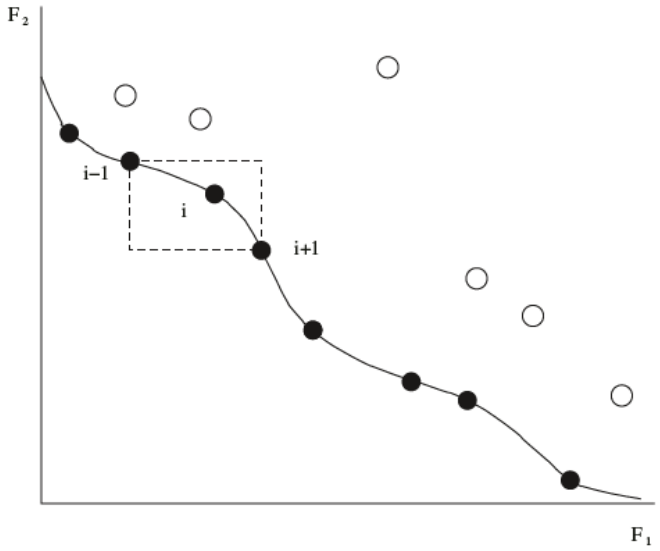


Figura 15. Ejemplo de "*crowding distance*" o distancia de aglomeración.

Aunque NSGA-II es un algoritmo relativamente viejo, sigue siendo muy utilizado para problemas con pocos objetivos en conflicto, ya que recientes estudios proponen que el uso del indicador de dominancia es inútil al trabajar con una alta cantidad de objetivos (Coello et al., 2007). En este caso la presión de selección que utiliza NSGA-II (y algoritmos similares) solo puede encontrar un grupo reducido de soluciones, estancándose sin posibilidad de enfocar la búsqueda hacia el conjunto óptimo de Pareto. Por lo cual, en años recientes se propone NSGA-III, diseñado principalmente para tratar problemas de optimización con más de cuatro objetivos en conflicto. Su principal diferencia yace en el mecanismo de preservación de diversidad implementado, el cual basa su funcionamiento en una serie de puntos de referencia bien distribuidos capaces de actualizarse de forma adaptativa.

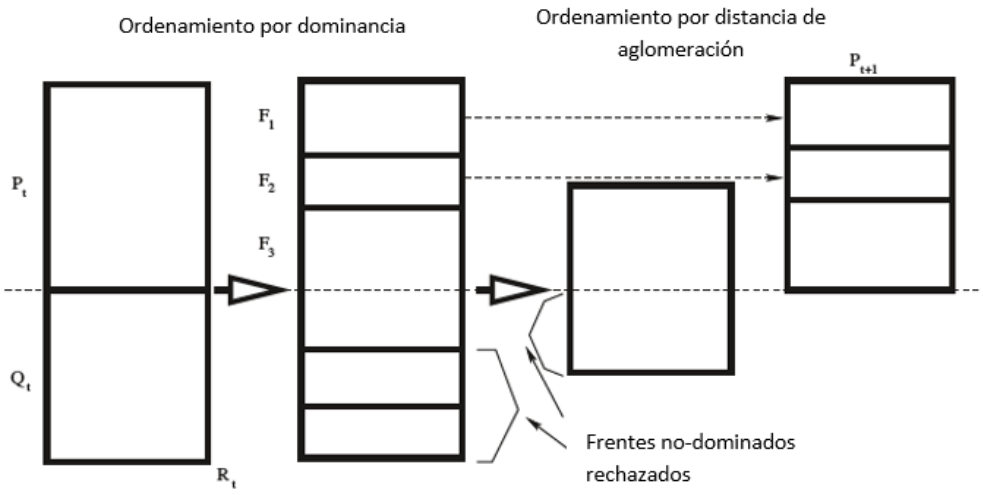


Figura 16. Diagrama de flujo que muestra la manera en que trabaja NSGA-II. P_t es la población de padres y Q_t la población de descendientes en la generación t. F_1 es el mejor frente no-dominado al combinar ambas poblaciones (padres e hijos). F_2 es el segundo mejor frente no-dominado y así sucesivamente (Coello et al., 2007).

4.3.3 Strength Pareto Evolutionary Algorithm 2 (SPEA 2)

El algoritmo evolutivo de optimización multiobjetivo SPEA en su primera versión fue propuesto por Zitzler y Thiele en 1998 como una alternativa para integrar diferentes MOEA, buscando combinar lo mejor de cada uno de ellos en una única herramienta. SPEA utiliza un archivo externo que contiene soluciones no-dominadas previamente obtenidas, comúnmente conocido como conjunto externo no-dominado. En cada generación, los individuos no-dominados se copian a este archivo, a la vez que se obtiene un valor de "*fortaleza*" para cada uno de ellos, valor directamente proporcional al número de soluciones a las que domina. De esta forma, la aptitud de cada miembro de la población actual se calcula de acuerdo a las "*fortalezas*" de todas las soluciones del archivo externo no-dominado que lo dominan, en otras palabras, hace uso de una dominancia por conteo. Adicionalmente, dado que el conjunto externo no-dominado participa proactivamente en el proceso de selección, en caso de aumentar considerablemente su tamaño podría llegar a reducir la presión de selección, lo que ralentizaría la búsqueda. Por tanto, se optó por adoptar una técnica de clustering llamada "*average linking method*" o método de enlace promedio, capaz de podar el conjunto del archivo externo para que su tamaño permanezca por debajo de un cierto umbral definido, preservando así la diversidad en la población. En general, SPEA marca el inicio de una segunda generación de MOEA al introducir el uso de elitismo. Se ha podido demostrar teóricamente que el elitismo es necesario para garantizar la convergencia de un algoritmo evolutivo multiobjetivo al verdadero frente de Pareto.

En 2001, Zitzler et al. proponen una versión mejorada de SPEA, llamada SPEA2, misma que actualmente es uno de los algoritmos evolutivos de optimización multiobjetivo más utilizados en la literatura para comparar el diseño de algún nuevo MOEA. SPEA2 (Zitzler et al., 2001) presenta tres diferencias fundamentales con respecto a su predecesor, las cuales son: 1) incorpora una estrategia de grano fino para asignación de aptitudes, la cual toma en cuenta la cantidad de individuos que cada solución domina y el número de individuos que la dominan, 2) hace uso de una técnica de estimación de densidad de vecinos que guían la búsqueda de manera más eficiente, y 3) presenta un esquema de truncamiento para el archivo externo que garantiza la preservación de soluciones de frontera.

4.3.4 Multiobjective Cellular genetic algorithm (MOCell)

A diferencia de muchos otros algoritmos evolutivos multiobjetivo, donde se cuenta con una única población y los individuos pueden interactuar entre sí sin ningún tipo de restricción, en un algoritmo evolutivo de tipo celular se estructura a la población en diversas subpoblaciones de menor tamaño con comunicación ocasional. Dicha topología es análoga al comportamiento de las especies al separarse en distintos nichos, apareándose y compitiendo ocasionalmente con aquellos individuos que se encuentren a cierta distancia de ellos, permitiendo así una lenta difusión del material genético inmerso y, por ende, una alta diversificación dentro de la población.

Los algoritmos genéticos de tipo celular (AGC) han destacado en la literatura por su rendimiento para casos de optimización multiobjetivo, tanto en distintos *"benchmark"* o problemas de referencia (Zitzler et al., 2000; Deb et al., 2001), como en implementaciones reales de ingeniería y calendarización (Guzek et al., 2014; Zavala et al., 2016). De igual manera, dada su naturaleza estos pueden ser implementados en paralelo, reduciendo en gran medida su tiempo de ejecución, a la vez que se aumenta su versatilidad y rendimiento (Peña et al., 2018).

MOCell es un algoritmo genético celular multiobjetivo propuesto por Nebro et al. (2009), el cual distribuye los individuos de una población en una malla cuadrada, para luego asignar un vecindario a cada uno (el tamaño del vecindario puede variar según la naturaleza del problema, véase **Figura 17**). El objetivo de emplear una malla toroidal es garantizar que toda solución tenga la misma cantidad de vecinos, sea cual sea su posición en la cuadricula. De esta forma, al restringir la cantidad de individuos que pueden interactuar entre sí, se promueve directamente la exploración del espacio de posibles soluciones, ya que

al transmitir genes entre vecindarios no es necesario enfocarse únicamente en el mejor individuo de cada subconjunto. Por otro lado, MOCell fomenta la explotación del espacio de soluciones aplicando las técnicas de ranking y distancia de aglomeración propuestas en NSGA-II (véase Sección 4.3.2) en cada uno de los vecindarios.

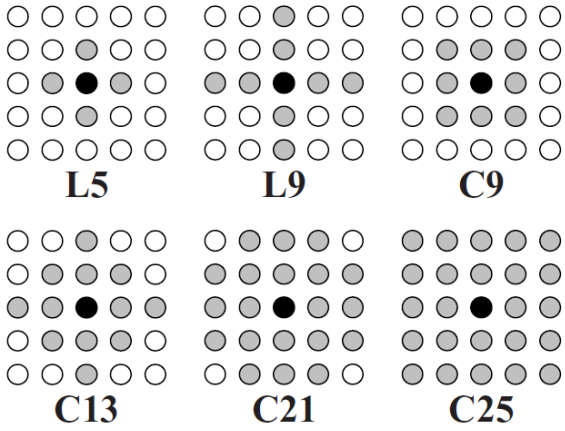


Figura 17. Tamaños de vecindario comúnmente utilizados en los algoritmos evolutivos celulares. Las etiquetas L_n y C_n representan el tipo de vecindario lineal (n vecinos más cercanos en dirección axial: norte, sur, oeste y este) y compacto ($n-1$ vecinos más cercanos en direcciones horizontal, vertical y diagonal), respectivamente. Los dos tipos más utilizados en la literatura son: (i) L5, también conocido como vecindario Von Neumann; y (ii) C9, conocido comúnmente como vecindario de Moore. Los vecindarios de mayor tamaño inducen un mayor nivel de migración implícita.

MOCell construye un conjunto ($\mathcal{P}_{ND}$) con n soluciones no-dominadas (la cardinalidad del conjunto implica el impacto del elitismo en el algoritmo) ordenadas por distancia de aglomeración, con las que posteriormente se retroalimenta la población. Una particularidad de este algoritmo es que no emplea el concepto de recambio generacional del mismo modo que el resto de los algoritmos evolutivos, ya que para cada generación es necesario revisar individuo por individuo, es decir, al evaluar cada elemento se ejecuta un torneo entre vecinos para seleccionar los padres o semillas que se utilizarán en el proceso de recombinación, para posteriormente aplicar el operador de mutación a la descendencia resultante. Una vez finalizado este proceso se hace un ranking entre hijos, padres y la solución a evaluar, con el propósito de definir quién de todos ellos ocupará dicho espacio en la población (usualmente el de mejor rendimiento) y así continuar con el siguiente. Por último, al finalizar la revisión de todas las soluciones, de manera aleatoria se seleccionan los n individuos que remplazarán a los ya alojados en el conjunto $\mathcal{P}_{ND}$ (véase **Figura 18**).

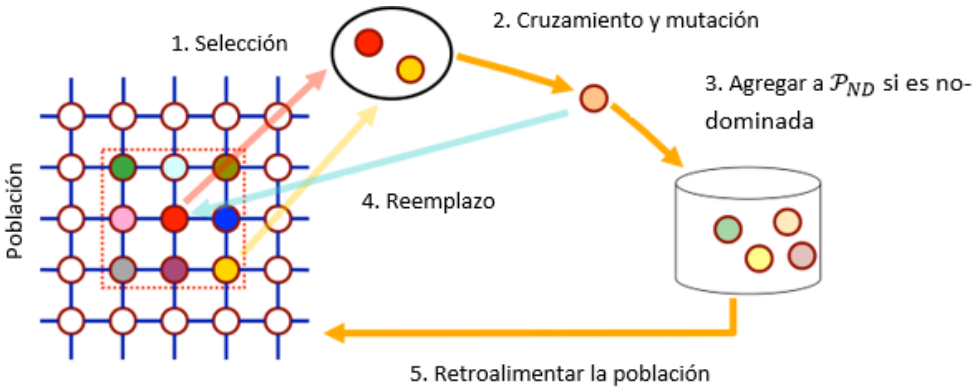


Figura 18. Ciclo de reproducción para cada individuo en un algoritmo genético celular con vecindario de dimensión nueve y un conjunto no-dominado de tamaño cuatro (Nebro et al., 2009).

Algoritmo 2. Pseudocódigo de algoritmo genético celular.

Inicio

Configurar parámetros de algoritmo y datos de entrada.

Generar conjunto vacío $\mathcal{P}_{\mathrm{ND}}$ para el posterior almacenamiento de soluciones no-dominadas.

Inicializar aleatoriamente la población ($\mathbb{X}$) con n soluciones candidatas (x). $x_1, x_2, \dots, x_n \in \mathbb{X}$.

Distribuir población en malla toroidal.

Evaluar el desempeño de cada individuo dentro de $\mathbb{X}$ con base a una función de aptitud. $x_i \rightarrow f_{aptitud}(x_i), i = \{1, \dots, n\}$.

Mientras (Criterio de parada ≠ verdadero) **hacer**

Para $i = 1$ **hasta** n **hacer**

Obtener vecindario $V = \{v_1, v_2, \dots, v_k\}$ para individuo i.

Seleccionar j padres de V. $\mathbb{P} = \{p_1, \dots, p_j\}$.

Recombinar pares de padres para generar m descendientes. $D = \{d_1, \dots, d_m\}$.

Mutar descendencia resultante.

Evaluar el desempeño de los nuevos candidatos. $d_l = f_{aptitud}(d_l), l = \{1, \dots, m\}$.

Reemplazar individuo $\mathbb{X}'(i)$ mediante un ranking entre descendientes, padres e individuo i.

Actualizar conjunto $\mathcal{P}_{\mathrm{ND}}$.

Fin

$\mathbb{X} \rightarrow \mathbb{X}'$.

Retroalimentar población $\mathbb{X}$ con $\mathcal{P}_{\mathrm{ND}}$.

Fin

4.4 Algoritmo propuesto

En esta sección se detalla el algoritmo genético celular de optimización multiobjetivo con penalización ambiental propuesto para la resolución del problema de enrutamiento de vehículos planteado en el Capítulo 2. Como primera instancia, se presenta el pseudocódigo del algoritmo implementado (véase **Algoritmo 2**), para posteriormente presentar la configuración efectuada en cada uno de los componentes clave del algoritmo evolutivo.

4.4.1 Codificación

Codificar una solución factible en un cromosoma no es más que asignar algún problema de optimización al dominio de un algoritmo evolutivo. No obstante, debe prestarse especial atención a dicha tarea, ya que la selección de una representación inapropiada puede conllevar a efectos perjudiciales que afecten directamente el rendimiento del proceso evolutivo.

Al aplicar este tipo de metaheurísticas al problema de enrutamiento de vehículos o en general al diseño de redes de transporte (topología y trayectos de las rutas) el esquema habitual de un EA cambia, esto debido a dificultades tales como: a) una ruta puede contener una cantidad variable de nodos, con una dimensión máxima de $n-1$ para un grafo compuesto por n nodos y por otra parte, b) una secuencia aleatoria de aristas usualmente no corresponderá a un camino valido en el grafo (Gen et al., 1997). Para dar frente a estas problemáticas, en este trabajo de investigación se adoptó el enfoque de codificación indirecto propuesto por Gen et al. (1997), también conocido como representación basada en prioridades, el cual consiste básicamente en codificar cierta información guía para la construcción de una ruta, en lugar de la ruta en sí. Como se mencionó anteriormente, un gen está compuesto por dos factores principales: el locus, es decir, la posición del gen dentro de la estructura del cromosoma, y el alelo, es decir, el valor que este toma. Siendo así, a través del uso de un genotipo expresado como permutación de enteros, esta codificación emplea la posición del gen (locus) para representar el identificador del nodo, mientras que el alelo indica la prioridad P_n asignada a este nodo (Gen et al., 2009) (véase **Figura 19**).

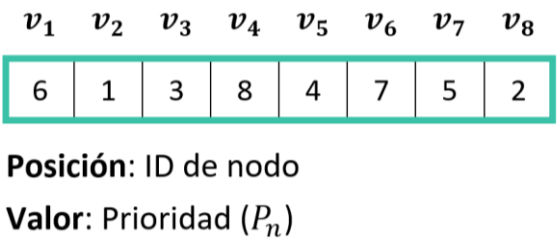
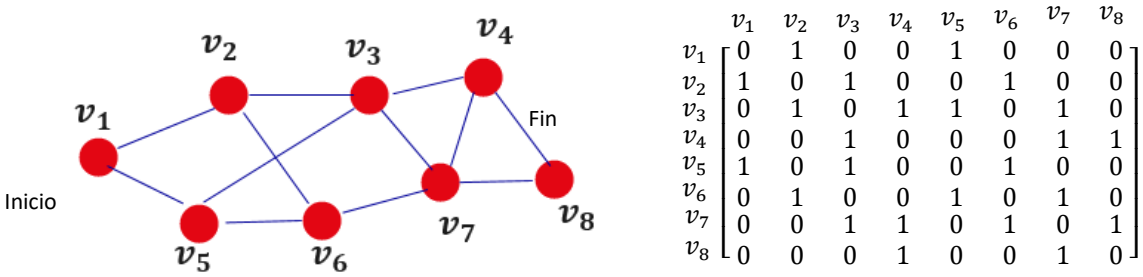


Figura 19. Codificación basada en prioridades para un cromosoma expresado como permutación de enteros de longitud $n = 8$.

La codificación basada en prioridades presenta una serie de ventajas con respecto a muchas otras representaciones propuestas en la literatura que la hacen factible particularmente para el problema de enrutamiento de vehículos, las principales son: 1) posibilita el mapeo de uno a uno entre codificaciones y rutas (**no redundancia**); 2) cualquier permutación en la codificación corresponde a una ruta valida (**legalidad**); 3) la gran mayoría de los operadores genéticos existentes en la literatura pueden aplicarse o adaptarse fácilmente a la codificación (**factibilidad**); 4) cualquier ruta posible tiene una codificación asociada (**completitud**), es decir, cualquier punto en el espacio objetivo es alcanzable en la búsqueda genética (Gen et al., 2006); dada la naturaleza del problema, es deseable que un generador de soluciones cuente con esta característica, ya que al no saber de antemano cómo es que se ve una *buena* solución, es conveniente que todas las posibles soluciones puedan ser representadas. Asimismo, otro factor a destacar de este enfoque indirecto es su propiedad **cuasi-Lamarkiana**, la cual hace referencia a la cuestión de si un cromosoma puede o no transmitir sus méritos a las próximas generaciones mediante operaciones genéticas comunes (recombinación, mutación y selección). De esta forma, el término *cuasi* se debe a que, al construir un camino *óptimo* por medio de un genotipo varias actividades compiten por una posición, lo cual significa que la interpretación de un gen con un mismo valor es parcialmente dependiente del contexto.



	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8
v_1	0	1	0	0	1	0	0	0
v_2	1	0	1	0	0	1	0	0
v_3	0	1	0	1	1	0	1	0
v_4	0	0	1	0	0	0	1	1
v_5	1	0	1	0	0	1	0	0
v_6	0	1	0	0	1	0	1	0
v_7	0	0	1	1	0	1	0	1
v_8	0	0	0	1	0	0	1	0

Figura 20. Grafo conexo no dirigido con 8 nodos y 12 aristas (izquierda) y su matriz de adyacencias correspondiente (derecha).

Siendo así, dado el grafo conexo no dirigido de la **Figura 20** que representa la topología subyacente de una red de carreteras, la **Figura 21** ejemplifica la decodificación de una ruta al ir del vértice v_1 al vértice v_4. Inicialmente, se verifica que v_2 y v_5 presentan adyacencias con el vértice inicial, mismos que tienen prioridades 1 y 4, respectivamente. Dado que el vértice v_5 cuenta con una prioridad mayor se coloca en la ruta resultante. Posteriormente, v_5 es adyacente a v_3 y v_6, de los cuales v_6 cuenta con la prioridad más alta, por lo que de igual manera se incluye en la ruta final. Luego se forma el conjunto de vértices adyacentes para v_6 y se selecciona v_7 de entre ellos. El proceso continúa hasta alcanzar el vértice destino v_4. En la práctica, el proceso de identificación de conexiones se logra por medio de una matriz de adyacencias, la cual representa la topología del grafo en sí. Se trata de una matriz cuadrada de n filas por n columnas, siendo n la cantidad de vértices en el grafo, donde para cada elemento $a_{ij} = 1$ si y solo si, existe una arista que una los vértices i y j; en caso contrario el elemento $a_{ij} = 0$ (véase **Figura 20**).

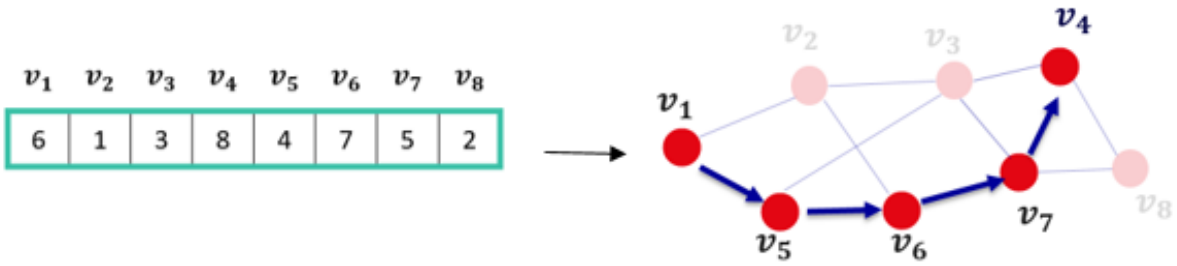


Figura 21. Codificación basada en prioridades para una ruta que va del vértice v_1 al v_4.

4.4.2 Función objetivo y evaluación de aptitud

El problema de optimización multiobjetivo se formula a través de un vector con tres funciones objetivo en conflicto $\mathbb{F} = [f_1, f_2, f_3]$, donde f_1 representa el tiempo para todos los vehículos al recorrer el conjunto de rutas, el cual se calcula a través del cociente de la distancia a recorrer entre la velocidad del vehículo en curso y, por ende, debe ser minimizado. De igual manera, se desea minimizar la función f_2, la cual evalúa el costo generado por cada vehículo al recorrer una ruta especifica. Dicho costo está dado por dos factores: el costo producido al recorrer la distancia de la ruta y la penalización ambiental generada, es decir, el costo implícito por kilómetro de trasladarse por un determinado segmento del mapa. Por otro lado, f_3 indica la suma de los contaminantes emitidos por cada uno de los vehículos al recorrer una determinada ruta, cuyo valor está dado por la potencia específica del vehículo (VSP, por sus siglas en inglés *"Vehicle Specific Power"*).

A grandes rasgos, las ecuaciones para el cálculo de f_1, f_2 y f_3 (véase Capítulo 2) consisten en la sumatoria de todos los tiempos, costos y emisiones por cada vehículo asignado a un viaje, respectivamente. Siendo así, se pretende encontrar un conjunto de rutas alternativas que eviten desplazarse por zonas altamente contaminadas y que, a su vez, minimicen el tiempo y la cantidad de emisiones al recorrer una determinada red de carreteras.

4.4.3 Operadores genéticos

Un factor clave en el desarrollo de cualquier algoritmo evolutivo es sin duda la implementación de diversos operadores genéticos, cuya analogía biológica yace en el proceso de herencia de genes para la generación de nuevos descendientes. En este sentido, dada su influencia en el rendimiento de estos enfoques evolutivos, se optó por realizar una sintonización de parámetros como parte del análisis experimental, permitiendo así identificar aquellos parámetros u operadores que permitan encontrar soluciones potenciales de mejor manera y a su vez, posibiliten una mejor exploración y explotación en el espacio de posibles soluciones. La Sección 5.3 de este documento presenta el procedimiento efectuado para la implementación de dicho estudio. A continuación, se describe brevemente la configuración resultante para cada uno de los componentes del EA.

Inicialización de la población: se generan aleatoriamente un conjunto de permutaciones con longitud n, donde n es el número de nodos en el grafo. Posteriormente, los individuos se distribuyen en una malla toroidal (véase **Figura 22**).

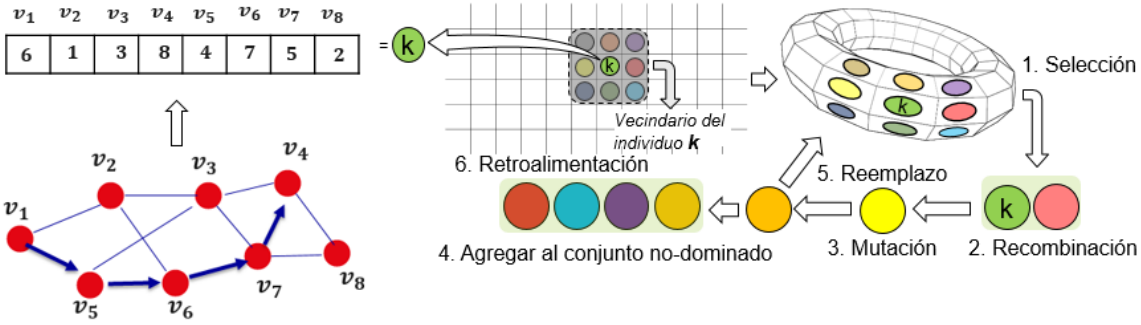


Figura 22. Representación de una solución (cromosoma) para el problema de enrutamiento de vehículos. Etapas de reproducción para un algoritmo genético celular.

Vecindario: se define un vecindario de dimensión nueve para cada individuo, conocido comúnmente como vecindario de Moore o C9, donde se permite a cada solución interactuar únicamente con sus ocho vecinos circundantes (véase **Figura 23**).

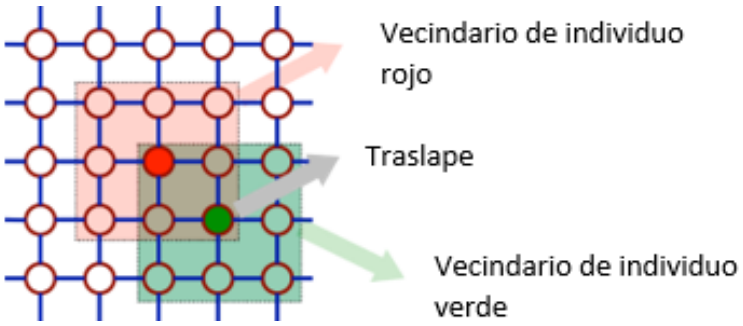


Figura 23. Vecindario con ocho vecinos circundantes para cada individuo (Nebro et al., 2009).

Selección de padres: se ejecuta una selección por torneo para cada uno de los individuos en la población y su correspondiente vecindario, obteniendo un par de individuos que serán utilizados para la generación del nuevo material genético. Dicho torneo implementa un ranking en frentes y la distancia de aglomeración para la elección de las dos mejores soluciones.

Mutación: para el caso puntual de cromosomas codificados como permutaciones de enteros existen dos posibles enfoques de mutación: mutaciones basadas en orden y basadas en adyacencias. Como bien su nombre lo indica, la primera está enfocada a problemas donde el orden en que ocurren los eventos es importante. Esto pueden suceder cuando los eventos utilizan recursos o tiempos limitados. Por otra parte, las mutaciones basadas en adyacencias buscan representar problemas donde las conexiones entre los eventos son más importantes que la secuencia en que estos suceden. De esta forma, dada la naturaleza del problema de enrutamiento de vehículos, se busca un operador de mutación basado en orden que permita hacer cambios puntuales en los cromosomas, de manera que la ruta resultante cambie únicamente la dirección de una de sus aristas. Por tal motivo, se consideraron cuatro distintos operadores de esta índole para la sintonización de parámetros efectuada: *"swap"*, *"insert"*, *"scramble"* e *"inverse"*. El operador de mutación *"swap"* propone la selección aleatoria de un par de genes en el cromosoma y el intercambio de valores entre ellos (véase **Figura 24**). Por otra parte, *"insert"* opera seleccionando dos alelos al azar y posicionando al segundo de ellos al lado del primero, afectando directamente a aquellos genes que se encuentren entre estas dos posiciones (véase **Figura 25**). *"Scramble"* consiste básicamente en seleccionar aleatoriamente un subconjunto de genes dentro del cromosoma e intercambiar sus valores,

manteniendo en todo momento las propiedades inherentes de una permutación (véase **Figura 26**). Por último, la mutación *"inverse"* selecciona aleatoriamente un subconjunto de genes dentro del cromosoma e invierte la secuencia en que sus valores aparecen (véase **Figura 27**). De acuerdo con dicho análisis estadístico se identificó al operador *"scramble"* como la mejor alternativa para la implementación del MOEA propuesto.

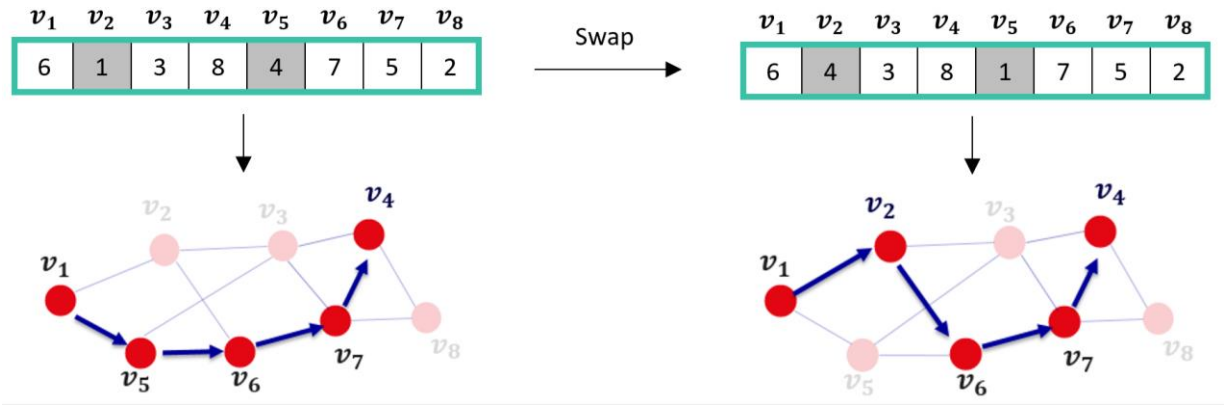


Figura 24. Operador de mutación *"swap"* implementado. Se seleccionan los vértices v_2 y v_5 como puntos de mutación y se realiza un intercambio de alelos entre ellos. Se ilustra la ruta resultante al decodificar el cromosoma antes y después de aplicar el operador.

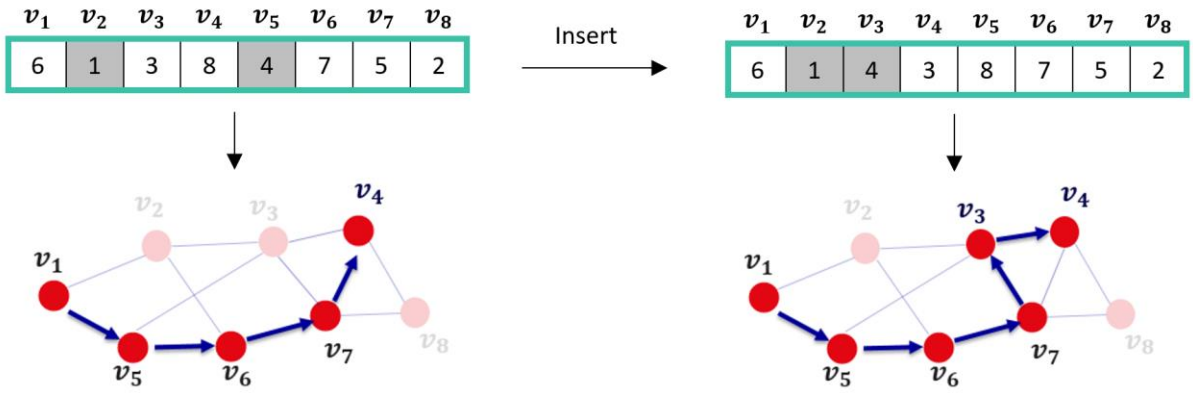


Figura 25. Operador de mutación *"insert"* implementado. Se seleccionan los vértices v_2 y v_5 como puntos de mutación y se posiciona el valor de v_5 al lado de v_2, es decir, en v_3; posteriormente, los alelos de aquellos genes posicionados entre v_2 y v_5 se recorren una posición. Se ilustra la ruta resultante al decodificar el cromosoma antes y después de aplicar el operador.

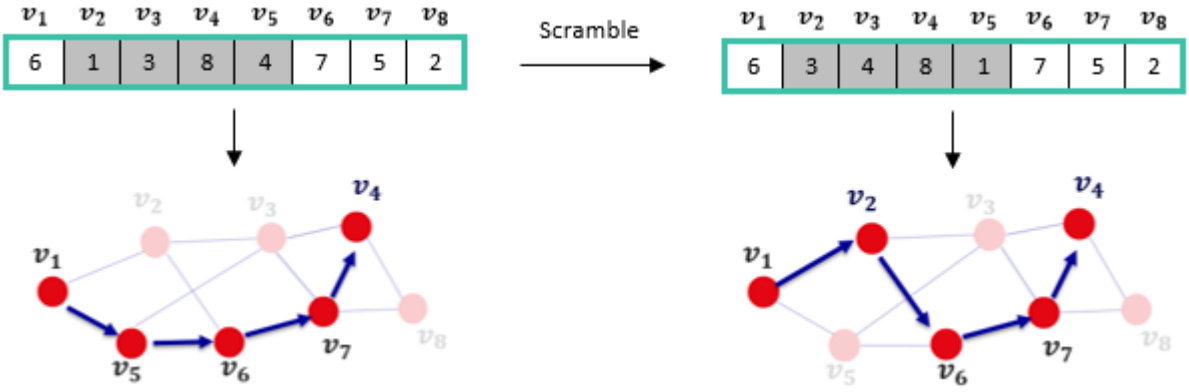


Figura 26. Operador de mutación *"scramble"* implementado. Se seleccionan los vértices v_2 y v_5 como puntos de mutación y se realiza una mezcla de los alelos entre ellos. Se ilustra la ruta resultante al decodificar el cromosoma antes y después de aplicar el operador.

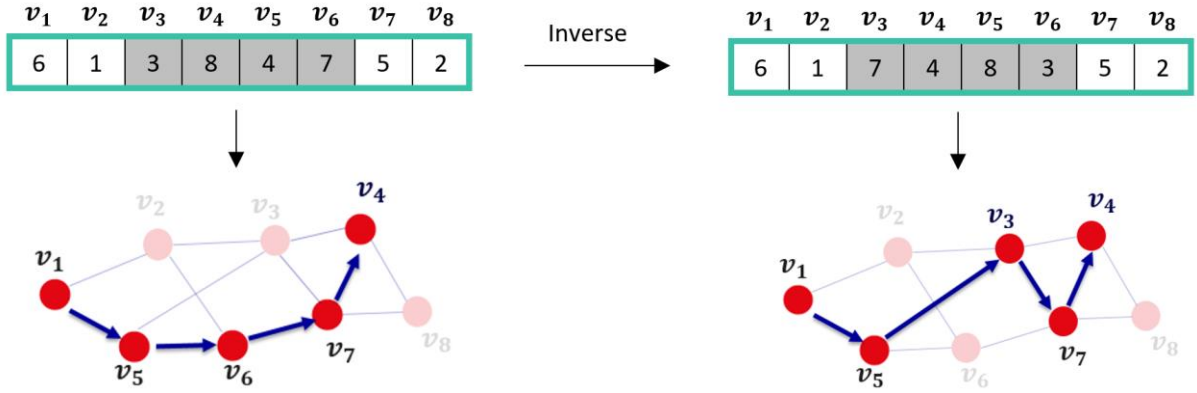


Figura 27. Operador de mutación *"inverse"* implementado. Se seleccionan los vértices v_3 y v_6 como puntos de mutación y se invierte la secuencia de los alelos entre ellos. Se ilustra la ruta resultante al decodificar el cromosoma antes y después de aplicar el operador.

Recombinación: al hacer uso de una codificación indirecta o basada en prioridades, es posible aplicar cualquier operador de recombinación diseñado para permutaciones de enteros; esto sin afectar en lo absoluto la validez de las rutas resultantes. En este sentido, se consideraron cuatro distintos operadores de esta índole para la sintonización de parámetros efectuada: OX, PMX, CX y EX. El operador de recombinación *"Order Crossover"* (OX) permite transmitir de padres a descendientes información sobre el orden relativo en que aparecen los genes (véase **Figura 28**). Por otro lado, *"Partially Mapped Crossover"* (PMX) pretende garantizar que cualquier información llevada en ambos padres esté presente en la descendencia (véase **Figura 29**).

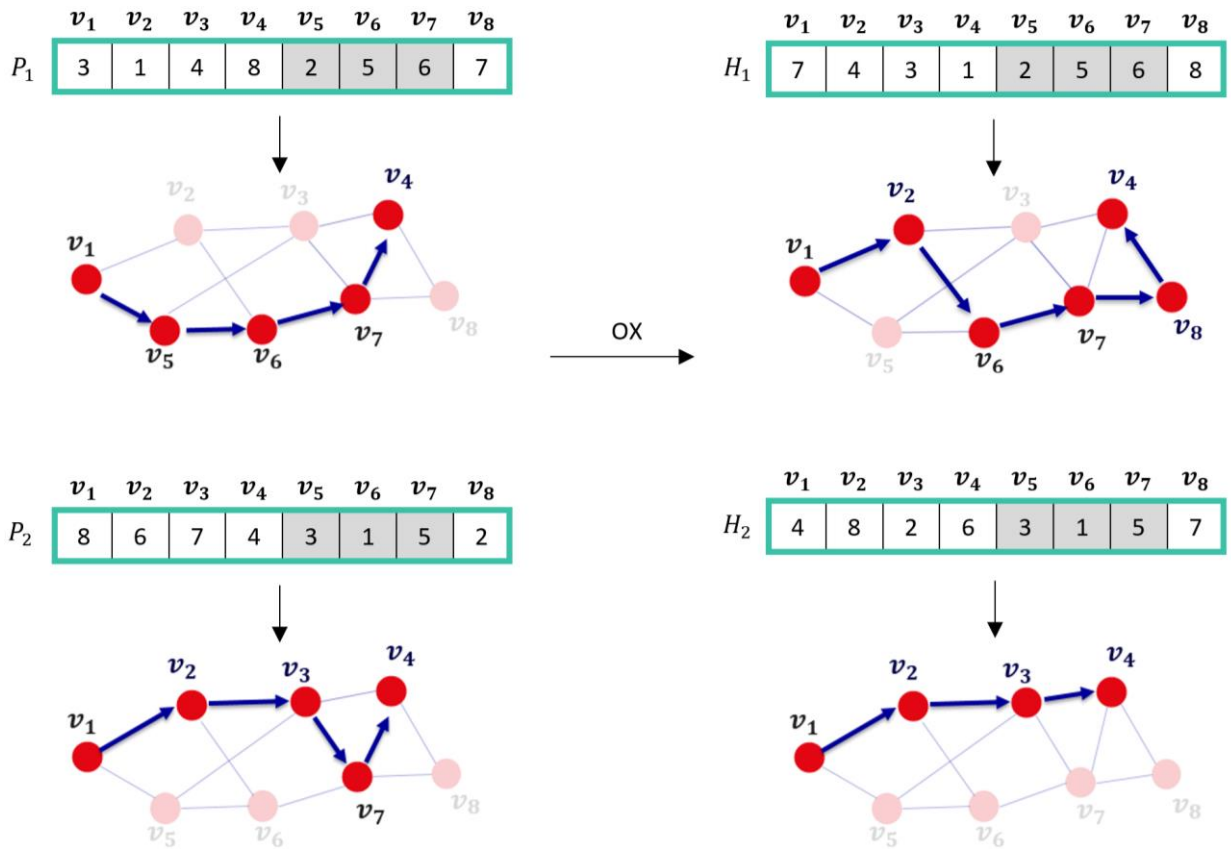


Figura 28. Operador de recombinación OX implementado. Dados dos padres P_1 y P_2 se da lugar a dos hijos H_1 y H_2 mediante el siguiente procedimiento: se seleccionan los vértices v_5 y v_7 como puntos de cruce y se copia el segmento entre ellos desde P_1 a la primera descendencia H_1. Posteriormente, a partir del segundo punto de cruce en P_2, se copian los elementos restantes no utilizados ya en H_1 en el orden en que aparecen en P_2. H_2 es generado de manera análoga con los roles invertidos para P_1 y P_2. Se ilustra la ruta resultante al decodificar el cromosoma de los padres y de la descendencia generada al aplicar el operador de recombinación OX.

El operador *"Cycle Crossover"* (CX) propone la segmentación del cromosoma mediante la identificación de ciclos. De esta forma, la descendencia es generada al alternar ciclos de cada uno de los padres, permitiendo preservar gran cantidad de información sobre la posición absoluta de los elementos (véase **Figura 30**). *"Edge Crossover"* (EX) es un operador de recombinación basado en adyacencias, es decir, actúa bajo la premisa de que la descendencia debe generarse únicamente a través de las conexiones presentes en los padres. Para ello, EX construye una lista donde para cada uno de los genes se enumeran los elementos vinculados a él en ambos padres (véase **Tabla 2** y **Figura 31**). De acuerdo con dicha sintonización de parámetros se encontró que un cruzamiento basado en orden (OX) es la mejor alternativa para la problemática en curso.

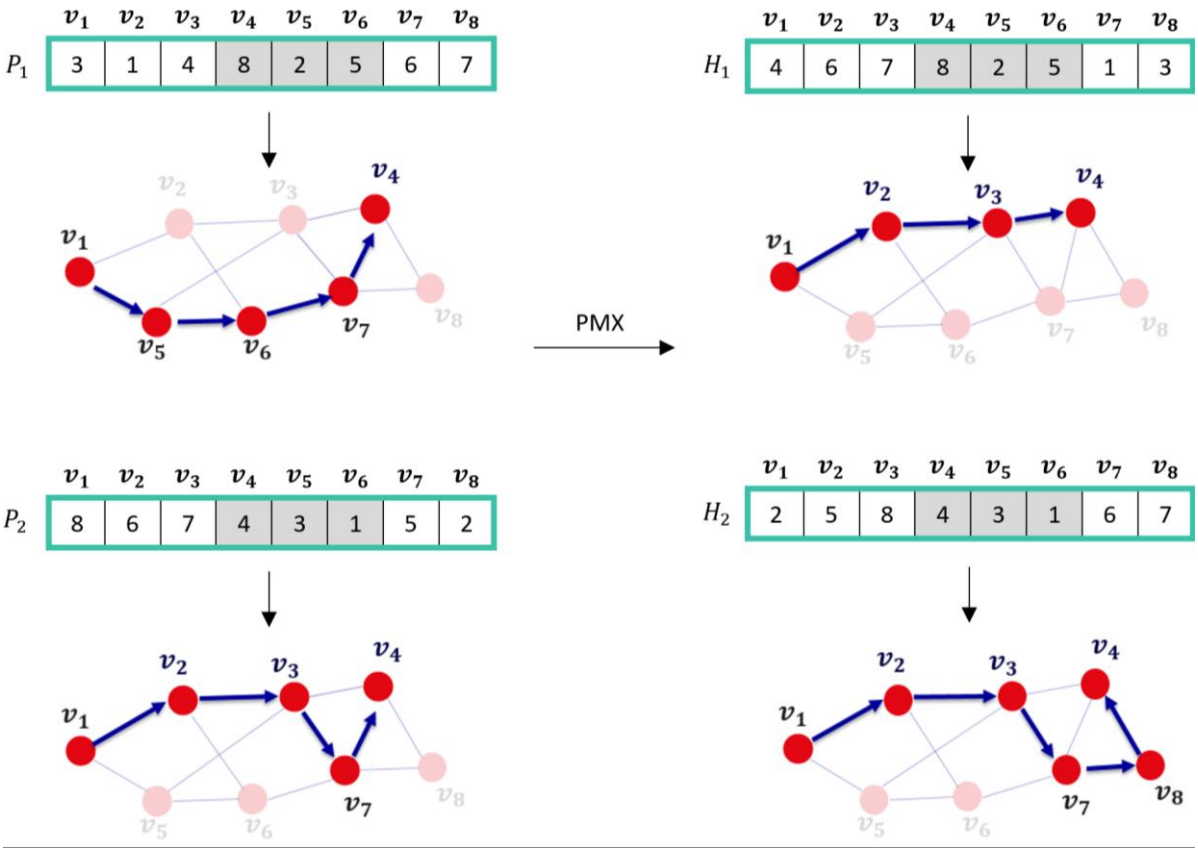


Figura 29. Operador de recombinación PMX implementado. Dados dos padres P_1 y P_2 se da lugar a dos hijos H_1 y H_2 mediante el siguiente procedimiento: se seleccionan los vértices v_4 y v_6 como puntos de cruce y se copia el segmento entre ellos desde P_1 a la primera descendencia H_1. Posteriormente, a partir del primer punto de cruce, se busca en P_2 elementos i que no hayan sido copiados ya en H_1 y que a su vez, estén incluidos en el segmento inicial; para cada elemento i se identifica en H_1 un elemento j posicionado en su lugar. De esta manera, se coloca i en la posición ocupada por j en P_2. Si el lugar ocupado por j en P_2 ya ha sido llenado en la descendencia por un elemento k, se coloca i en la posición ocupada por k en P_2. Las posiciones restantes no contenidas en el segmento inicial se copian a H_1 desde P_2. H_2 es generado de manera análoga con los roles invertidos para P_1 y P_2. Se ilustra la ruta resultante al decodificar el cromosoma de los padres y de la descendencia generada al aplicar el operador de recombinación PMX.

Reemplazo: para cada celda de la malla toroidal se ejecuta un torneo entre padres, hijos y el individuo posicionado en dicha celda, donde aquel que presente el mejor rendimiento será el que ocupará este lugar en la población. Después, esta solución podrá competir por un lugar en el conjunto de soluciones no-dominadas $\mathcal{P}_{ND}$. Finalmente, con el fin de dar lugar a nueva generación, se procede a realizar una selección aleatoria de $|\mathcal{P}_{ND}|$ individuos que serán eliminados y reemplazados de la población actual por los almacenados en el conjunto $\mathcal{P}_{ND}$ (véase **Figura 22**).

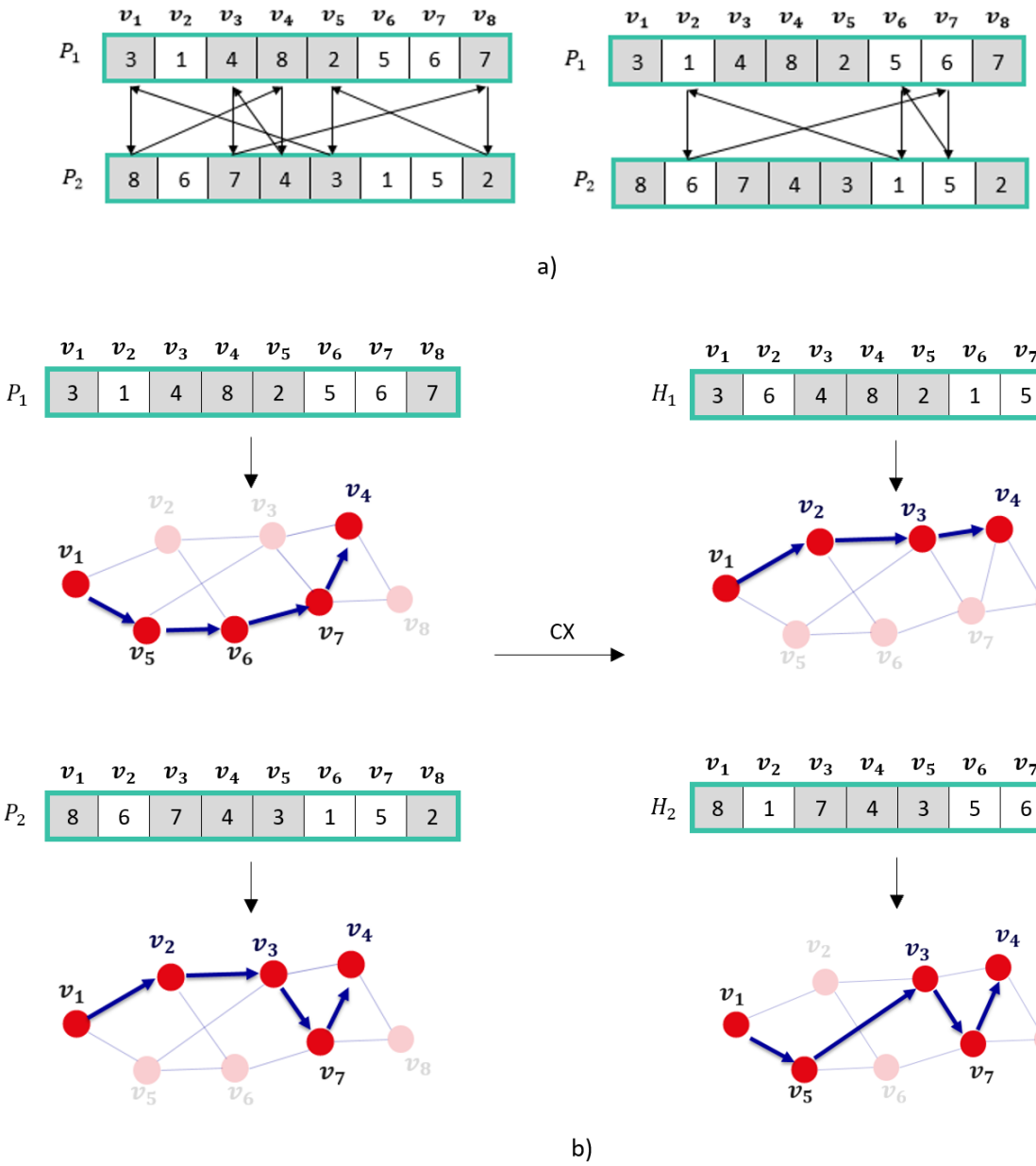


Figura 30. Operador de recombinación CX implementado. a) Identificación de ciclos y, b) construcción de descendencia al alternar ciclos resultantes. Dados dos padres P_1 y P_2 se da lugar a dos hijos H_1 y H_2 mediante el siguiente procedimiento: se busca en P_1 el primer elemento i no asignado a un ciclo y se identifica en P_2 un elemento j posicionado en su lugar. Posteriormente, se incluye al ciclo a aquel elemento con el mismo alelo que j en P_1. Dicho proceso itera hasta alcanzar nuevamente el elemento inicial i, es decir, hasta generar un ciclo. H_2 es generado de manera análoga con los roles invertidos para P_1 y P_2. Se ilustra la ruta resultante al decodificar el cromosoma de los padres y de la descendencia generada al aplicar el operador de recombinación CX.

Tabla 2. Lista de adyacencias para operador de recombinación EX implementado. Un símbolo "+" en la tabla indica que dicha conexión está presente en ambos padres.

Nodo	Adyacencias
1	3+, 4, 5
2	5+, 8+
3	1+, 4, 7
4	1, 3, 7, 8
5	1, 2+, 6
6	5, 7+, 8
7	3, 4, 6+
8	2+, 4, 6

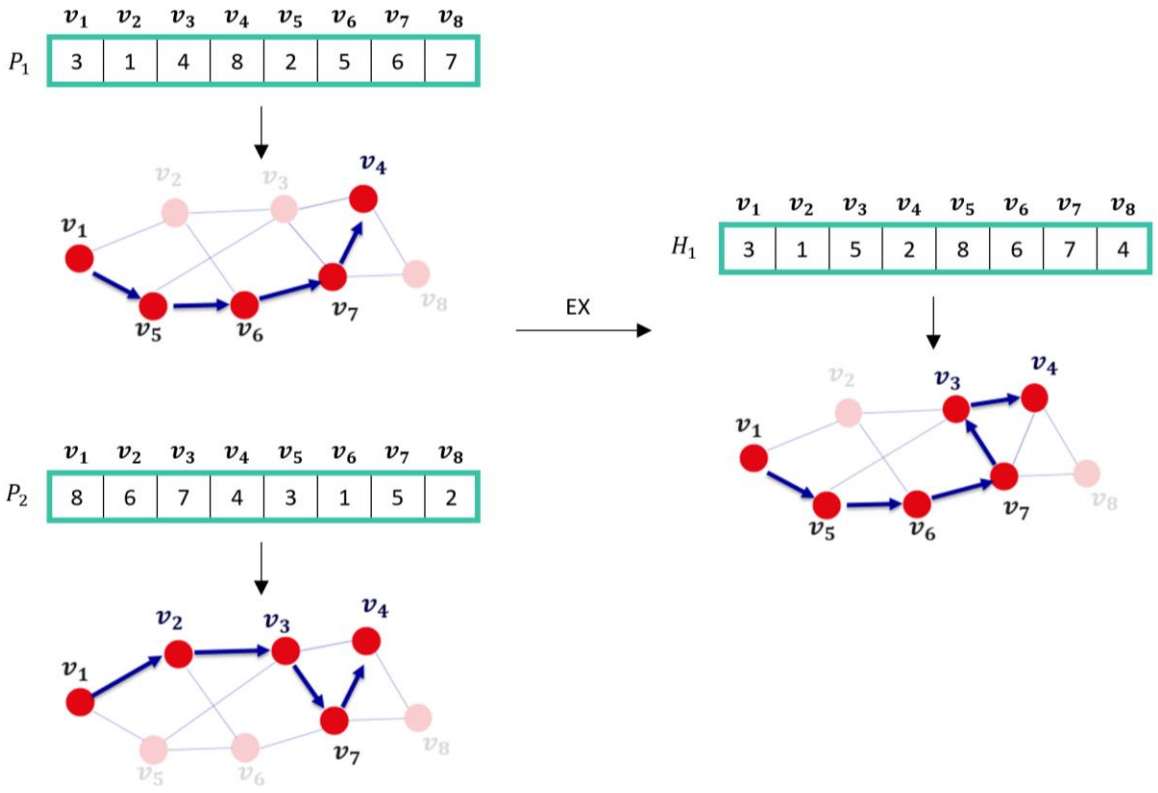


Figura 31. Operador de recombinación EX implementado. Dados dos padres P_1 y P_2 es posible dar lugar a un hijo H_1 mediante el siguiente procedimiento: dada una lista de adyacencias, donde se enumeran los elementos vinculados a cada uno de los genes (véase Tabla 2), se selecciona aleatoriamente un elemento i y se eliminan todas las referencias efectuada a él en dicha lista. Posteriormente, a partir de la lista de adyacencias de i, se selecciona al siguiente nodo considerando tres posibles criterios (ordenados jerárquicamente): a) elegir aquel elemento con adyacencias en común para ambos padres, b) seleccionar el elemento con la lista de adyacencias de menor dimensión o, c) seleccionar aleatoriamente un elemento. Los criterios deben ser considerados en el orden indicado. Dicho proceso itera hasta generar completamente el cromosoma de H_1. Se ilustra la ruta resultante al decodificar el cromosoma de los padres y de la descendencia generada al aplicar el operador de recombinación EX.

Capítulo 5. Resultados experimentales

Este capítulo detalla la evaluación experimental de los distintos MOEA implementados para la resolución del problema de enrutamiento de vehículos con penalización ambiental, objeto de estudio del presente trabajo de investigación. La Sección 5.1 describe los indicadores de calidad (métricas de rendimiento) empleados para cuantificar la calidad de las soluciones encontradas tanto por el algoritmo genético celular propuesto como por NSGA-II y SPEA2; dos técnicas de optimización multiobjetivo representativas del estado del arte. La Sección 5.2 describe el diseño de los experimentos implementados para la evaluación de los MOEA. La Sección 5.3 presenta el análisis estadístico ejecutado para la parametrización de los algoritmos. Por último, las Secciones 5.4 y 5.5 presentan los resultados experimentales y la discusión en torno al análisis de los mismos.

5.1 Indicadores de calidad

A diferencia de problemas donde únicamente se busca satisfacer un simple objetivo y la optimalidad está dada por una única solución que lo maximiza o minimiza, en los MOOP se busca encontrar aquel conjunto de soluciones no-dominadas que satisfaga las restricciones inherentes de la problemática y a su vez, optimice dos o más funciones simultáneamente. A raíz de ello, surge la necesidad de diseñar mecanismos que permitan comparar los resultados entregados por dos distintos algoritmos de optimización y determinar cuál de ellos presentó un mejor rendimiento. En el contexto del cómputo evolutivo, dicha tarea no resulta trivial dado que los MOEA no producen un numero escalar como indicador de aptitud, el cual pueda ser tratado con diversos métodos estadísticos. En contraparte, estos entregan un conjunto de soluciones no dominadas (puntos en el espacio objetivo) como resultado. Por tanto, numerosas métricas han sido propuestas en la literatura para evaluar el desempeño de algoritmos evolutivos multiobjetivo, considerando tanto la convergencia hacia el frente de Pareto como la dispersión del conjunto de soluciones encontrado (Coello et al., 2007).

Siendo así, dado un MOOP con k funciones objetivo expresado como $\mathbb{F} = [f_1(\mathbb{x}), \dots, f_k(\mathbb{x})]$ donde se busca minimizar $\mathbb{F}$**,** para cada función objetivo k se asigna a cada solución $\mathbb{x}$ en el espacio objetivo Ω un valor real $z_i = f_i(x)$ que denota el mérito alcanzado de acuerdo con el i – ésimo criterio de optimización. Por tanto, cada solución $\mathbb{x} \in \Omega$ es mapeada a un vector $\mathbb{z} = [z_1, \dots, z_k] \in \Omega$. De este modo, una métrica de rendimiento de orden h puede definirse formalmente como una función I_i: $\Omega^h \rightarrow \mathbb{R}$, la cual asigna un

valor real $I_i(\mathcal{P}_1, \dots, \mathcal{P}_h)$ a cada conjunto $\mathbb{P} = [\mathcal{P}_1, \dots, \mathcal{P}_h]$ de h conjuntos aproximados. Es posible clasificar estas métricas en tres principales grupos (Riquelme et al., 2015): métricas de cardinalidad, exactitud y diversidad. Para el primer conjunto de métricas, la cardinalidad de un conjunto de soluciones está dada por la cantidad de elementos presentes en $\mathcal{P}$. Por otra parte, las métricas de exactitud evalúan el nivel de convergencia de un $\mathcal{P}$, es decir, indica la distancia que separa $\mathcal{P}$ de $\mathcal{P}_{\mathrm{F}}$ en caso de que este último sea conocido, de lo contrario se emplea un frente de referencia $\mathcal{P}_{\mathrm{R}}$ (regularmente calculado de forma experimental). Por último, las métricas de diversidad evalúan la distribución y extensión de las soluciones en $\mathcal{P}$, donde la distribución representa la distancia relativa entre las soluciones no-dominadas, mientras que la extensión hace referencia al espacio cubierto por cada solución en $\mathcal{P}$.

Estos indicadores pueden ser unarios o binarios, en función de cuantos conjuntos de soluciones pueden evaluar simultáneamente, ya sea en relación a convergencia, diversidad o cardinalidad del conjunto encontrado con respecto al frente de Pareto. No obstante, de acuerdo con Riquelme et al. (2015) entre los años 2005 y 2013, un total de 54 métricas diferentes fueron citadas en la literatura; por lo que surge naturalmente la pregunta ¿Cuántos o cuáles indicadores de calidad se deben utilizar? (Peña y Tchernykh, 2017).

En este trabajo de investigación, se toma como base el estudio implementado por Riquelme et al. (2015), donde se examinan los indicadores más utilizados, permitiéndonos seleccionar un conjunto relativamente pequeño de métricas que posibiliten evaluar al menos dos de los tres criterios de rendimiento previamente mencionados. En este sentido, se eligieron los indicadores Hipervolumen (I_{HV}) y Spread (I_{Δ}) con el propósito de cuantificar convergencia y dispersión de los frentes aproximados otorgados por cada una de las implementaciones (MOCell, NSGA-II y SPEA2) y así, comparar su desempeño para el problema de enrutamiento de vehículos con penalización ambiental.

No obstante, el frente de Pareto—que es desconocido para la instancia del problema estudiada— es aproximado por el conjunto de soluciones no-dominadas encontradas en el total de ejecuciones realizadas para cada uno de los algoritmos implementados. Dicho frente aproximado de Pareto es objeto de referencia en las múltiples métricas de calidad utilizadas en la evaluación experimental de este documento.

5.1.1 Hipervolumen (I_{HV})

El indicador de calidad I_{HV}, también conocido como métrica S o medida de Lebesgue, calcula el volumen cubierto por los miembros de un conjunto de soluciones no-dominadas $\mathcal{P}$ para problemas donde todos los objetivos deben ser minimizados, en otras palabras, mide el tamaño de la porción del espacio objetivo dominada por dichas soluciones, colectivamente. Una propiedad teórica interesante de esta métrica es su relación estrictamente monotónica con respecto a la dominancia de Pareto, la cual garantiza que el frente de Pareto ($\mathcal{P}_F$) logra el máximo hipervolumen posible, mientras que cualquier otro conjunto recibirá un valor menor para este indicador. Fleischer et al. (2003) demostraron que, dado un espacio de búsqueda finito y un punto de referencia W, la maximización del hipervolumen es directamente proporcional a la obtención del conjunto óptimo. Por lo que, un conjunto acotado que obtenga el valor máximo ($I_{HV} = 1$) consta únicamente de soluciones que pertenecen a $\mathcal{P}_F$. Además, estudios empíricos han demostrado que para una cierta cantidad de puntos previamente determinados, la maximización de dicha métrica produce subconjuntos del frente de Pareto bien distribuidos (Coello, 2018). De tal forma, comparar el rendimiento de las soluciones arrojadas por distintos MOEA con respecto a este indicador permitirá posteriormente cuantificar tanto la convergencia como su diversidad a lo largo del frente óptimo.

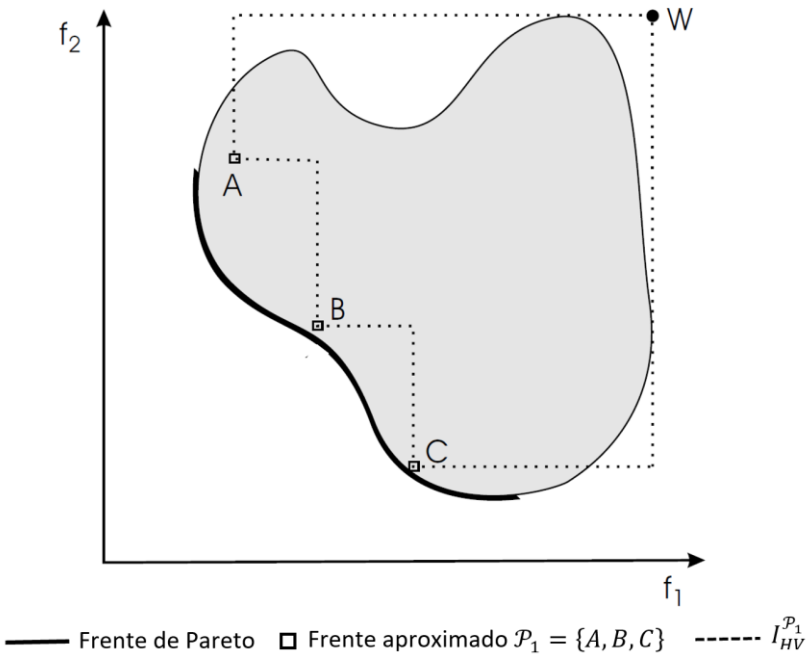


Figura 32. El hipervolumen encerrado por las soluciones que pertenecen al conjunto de soluciones no-dominadas $\mathcal{P}_1 = \{A, B, C\}$ y el punto de referencia W.

Matemáticamente, para cada elemento $i \in \mathcal{P}$, se construye un hipercubo v_i utilizando como esquinas diagonales un punto de referencia W y la solución i. El punto de referencia no debe ser rebasado por ninguna solución factible, por lo que usualmente se define como un vector con los peores valores posibles de las funciones objetivo (Nebro et al., 2009). A partir de ello, se realiza la unión de todos estos hipercubos y se procede a calcular el hipervolumen (véase **Ecuación 28**).

$$I_{HV} = volumen\left(\bigcup_{i=1}^{|\mathcal{P}|} v_i\right). \quad (28)$$

No obstante, dado que este indicador no está exento de una escala de objetivos arbitraria, se recomienda emplearla en espacios objetivo normalizados. La región encerrada por la línea discontinua en la **Figura 32** representa el hipervolumen del frente aproximado $\mathcal{P}_1$, conjunto compuesto por los puntos no-dominados A, B y C. Al contar únicamente con dos funciones objetivo f_1 y f_2, basta con calcular el área de cada rectángulo generado al utilizar como diagonal la línea entre cada solución y el punto de referencia W.

5.1.2 Spread (I_Δ)

I_Δ es un indicador de diversidad unario que cuantifica la dispersión de las soluciones no-dominadas obtenidas mediante un MOEA en un conjunto $\mathcal{P}$, contemplando la distancia hacia los extremos del frente de Pareto (Nebro et al., 2009) (véase **Ecuación 29**).

$$I_\Delta = \frac{d_f + d_l + \sum_{i=1}^{|\mathcal{P}|-1}\left|d_i - \bar{d}\right|}{d_f + d_l + (|\mathcal{P}| - 1)\bar{d}}, \quad (29)$$

donde d_i indica la distancia Euclidiana entre soluciones consecutivas, $\bar{d}$ es la media de estas distancias, d_f y d_l son las distancias Euclidianas a los extremos del frente de Pareto en el espacio objetivo (véase **Figura 33**). Esta medida toma un valor de cero $I_\Delta = 0$ para una distribución ideal que incluye los extremos del frente óptimo en el conjunto $\mathcal{P}$, valor asociado a una distribución perfectamente equiespaciada de las

soluciones no-dominadas. De igual manera, se recomienda emplear dicho indicador en espacios objetivo previamente normalizados.

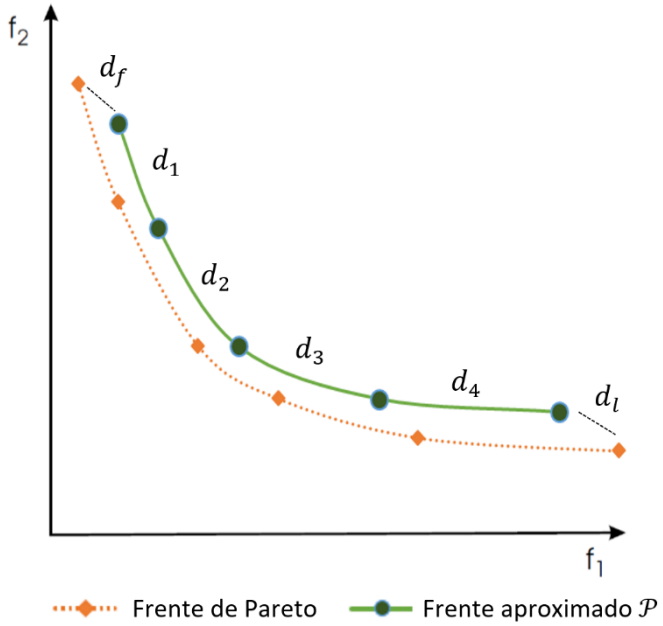


Figura 33. Distancias entre soluciones consecutivas del frente aproximado $\mathcal{P}$, denotadas como d_i donde $i = 1, \dots, 4$. De igual forma, se ilustran las distancias Euclidianas a los extremos del frente de Pareto d_f y d_l a partir de las soluciones más cercanas en $\mathcal{P}$.

5.2 Diseño de experimentos

Los distintos algoritmos evolutivos multiobjetivo utilizados en este trabajo de investigación (MOCell, NSGA-II y SPEA2) fueron implementados en el lenguaje de programación orientado a objetos Java, utilizando el framework de desarrollo JMetal 5 (Durillo y Nebro, 2011), una biblioteca de código abierto dirigida al desarrollo, experimentación y estudio de metaheurísticas para la resolución de problemas de optimización multiobjetivo. JMetal incluye una serie de algoritmos representativos del estado del arte, un amplio conjunto de problemas de referencia o "*benchmarks*", e indicadores de calidad bien conocidos para la evaluación del rendimiento de estos enfoques. Asimismo, esta herramienta permite la generación automática de información estadística de los resultados obtenidos y su vez, facilita el uso de procesadores multinúcleo para la aceleración del tiempo de ejecución en los experimentos. De tal forma, al ser una plataforma completamente abierta, altamente portable (dado el uso de Java), flexible y extensible, JMetal ha pasado a ser una valiosa herramienta en el área de la optimización multiobjetivo.

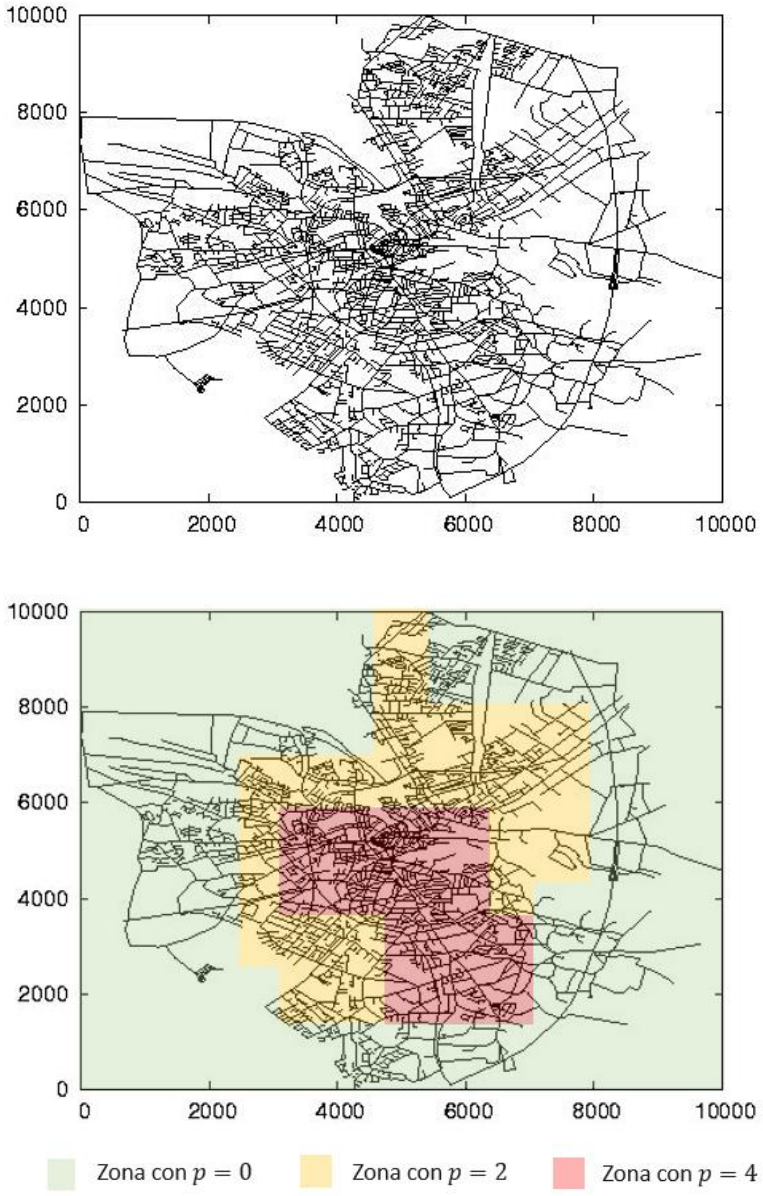


Figura 34. Red de carreteras de la ciudad de Oldemburgo (OL) (Brinkhoff, 2002) (arriba) y red de carreteras OL segmentada con base en la penalización ambiental (p) por zona (abajo). Una arista posicionada en una zona de color verde tiene asociada una penalización nula ($p = 0$), mientras que una zona amarilla $p = 2$ y una zona roja $p = 4$.

Como instancia del problema, se utilizó un grafo conexo dirigido compuesto por 6104 nodos, el cual representa la topología subyacente de la red de carreteras en la ciudad de Oldemburgo (Brinkhoff, 2002) (véase **Figura 34**). Dicho grafo se segmentó en tres diferentes tipos de zona, las cuales simbolizan áreas con distinta penalización ambiental, es decir, el costo ambiental implícito de trasladarse por un determinado segmento del mapa. Una arista posicionada en una zona de color verde tiene asociada una

penalización nula ($p = 0$), mientras que una zona amarilla $p = 2$ y una zona roja $p = 4$. A grandes rasgos, se pretende encontrar un conjunto de rutas alternativas que eviten desplazarse por zonas altamente contaminadas (o bien, con alta penalización ambiental) y que, a su vez, permitan minimizar el tiempo y la cantidad de emisiones generadas por múltiples vehículos al recorrer una determinada red de rutas.

Por otro lado, una valoración realista del desempeño de un MOEA requiere de una evaluación experimental mediante la realización de numerosas ejecuciones, y la aplicación de diversos análisis estadísticos que permitan interpretar adecuadamente los resultados arrojados. Por ello, tanto para el algoritmo genético celular propuesto como para los algoritmos NSGA-II y SPEA2, dos técnicas representativas del estado del arte, se realizaron 30 ejecuciones independientes con 25,000 evaluaciones de calidad como criterio de parada en cada una de ellas. Los conjuntos aproximados fueron normalizados considerando los valores máximos posibles para cada función objetivo.

Al igual que en muchos otros algoritmos evolutivos enfocados a la resolución de problemas de optimización, en el algoritmo genético celular propuesto se ejecuta un proceso de corrección de soluciones previo a la evaluación de calidad, garantizando que todos los cromosomas cumplan con las restricciones inherentes a la codificación de individuos. Específicamente, se implementó un mecanismo que evita la generación de ciclos dentro de las rutas decodificadas, es decir, evita que un nodo sea incluido en más de una ocasión dentro de una ruta. De tal forma, dicha función ofrece a toda permutación de enteros la posibilidad de convertirse en una solución factible para el problema de enrutamiento de vehículos.

Una vez definida la instancia del problema a tratar, la estructura general del algoritmo evolutivo implementado y las métricas de calidad a utilizar para evaluar convergencia y diversidad del conjunto de soluciones no-dominadas resultante, se procede a realizar una sintonización de parámetros a través de un análisis de varianzas multifactorial; esto con el propósito de identificar aquellos parámetros que permitan encontrar soluciones potenciales de manera eficiente y a su vez, posibiliten una mejor exploración y explotación en el espacio de posibles soluciones (véase **Sección 5.3**). Todo esto para posteriormente implementar un análisis comparativo entre el MOEA propuesto y dos enfoques comúnmente utilizados en la literatura: NSGA-II y SPEA2.

5.3 Sintonización de parámetros

Como es bien sabido, los algoritmos evolutivos pertenecen al campo de las metaheurísticas y, por ende, son técnicas altamente no-determinísticas. Esto implica principalmente que dos ejecuciones del mismo algoritmo para un mismo problema bajo las mismas condiciones, pueden llegar a arrojar diferentes soluciones. Esta particularidad de las metaheurísticas representa un importante problema al momento de evaluar resultados o bien, comparar implementaciones.

Dada dicha naturaleza no-determinística, realizar comparativas de resultados a partir de una simple ejecución no es factible, por lo que estas deben realizarse a través de un gran conjunto de resultados obtenidos después de múltiples ejecuciones independientes. En la literatura, funciones estadísticas comúnmente utilizadas suelen ser el promedio, la mediana o las mejores y peores soluciones encontradas por las técnicas estudiadas. Sin embargo, estos no son valores concluyentes y pueden conllevar a interpretaciones erróneas. Por lo tanto, se ha optado por utilizar estudios que garanticen una alta significancia estadística de los resultados y las comparativas presentadas. El análisis de varianza (ANOVA) es un ejemplo de ello, el cual permite determinar si los efectos observados en los resultados obtenidos son estadísticamente significativos o, por el contrario, se deben a errores en las muestras realizadas (Alba y Dorronsoro, 2008). Básicamente, en un estudio de esta índole se verifica en qué grado una cierta variable continua Y, llamada variable respuesta, y ciertas variables categóricas $F_1, F_2, \dots, F_n$ llamadas factores, están o no relacionadas, cuantificando la diferencia entre las medias poblacionales obtenidas por los distintos factores. Cabe destacar que las muestras deben contar con una distribución normal.

El desempeño de la planificación táctica en el enrutamiento de vehículos depende directamente de las características propias de la instancia del problema, los métodos de solución, así como de la parametrización de los algoritmos seleccionados. En este sentido, es necesario la implementación de análisis estadísticos que permitan interpretar los resultados o bien, validar una hipótesis dada. Como bien se mencionó en el Capítulo 4, los algoritmos evolutivos constan de secciones que operan con diferentes probabilidades para su ejecución (recombinación, mutación o selección), además de elementos de tamaño variable (población, torneo de selección o conjunto de individuos no-dominados para retroalimentación), mismos que afectan directamente el desempeño y comportamiento del algoritmo.

En este contexto, con el propósito de identificar aquellos parámetros que permitan encontrar soluciones potenciales de manera eficiente y a su vez, posibiliten una mejor exploración y explotación en el espacio

de posibles soluciones, se optó por realizar una sintonización de parámetros a través de un análisis de varianzas multifactorial.

Para ello, en este trabajo se adaptó el método experimental propuesto por Ruíz y Maroto (2006), el cual consta de los siguientes pasos: a) probar todas las instancias producidas con todas las posibles combinaciones de parámetros; b) cuantificar el desempeño de cada instancia (Heu_{sol}); c) aplicar un Análisis de Varianza Multifactorial (ANOVA) con un nivel de confianza del 95% para identificar los parámetros con mayor significancia estadística; d) configurar los parámetros del algoritmo con base en los valores arrojados por el estudio; e) calcular la distancia relativa sobre la mejor solución obtenida durante la prueba de todas las instancias.

Siendo así, se definieron diversos niveles para cada uno de los parámetros a sintonizar (véase **Tabla 3**), los cuales son: operadores de recombinación y mutación, probabilidad de recombinación (P_r), probabilidad de mutación (P_m) y tamaño de la población. Por lo tanto, un total de $3 \cdot 4 \cdot 5 \cdot 4 \cdot 5 = 1200$ diferentes configuraciones fueron contempladas.

Tabla 3. Niveles por factor utilizados en análisis estadístico ANOVA Multifactorial.

Tamaño de la población	Recombinación	Probabilidad de recombinación	Mutación	Probabilidad de mutación
100	PMX	0.5	Swap	0.05
144	OX	0.6	Insert	0.1
225	CX	0.7	Scramble	0.2
	EX	0.8	Inverse	0.3
		0.9		0.4

Tomando como base el trabajo realizado por Sadeghi et al. (2014), se consideraron cuatro métricas diferentes para cuantificar el desempeño de las configuraciones, cada una representando un aspecto importante en la calidad de una solución obtenida mediante un MOEA. Estas son: 1) hipervolumen (I_{HV}), 2) *spread* (I_{Δ}), 3) distancia generacional (I_{GD}) y 4) la mejor solución no-dominada obtenida ($best_{NDP}$) (Deb y Sundar, 2006). Nótese que para obtener el valor $best_{NDP}$ de cada instancia, se realizó una suma ponderada con pesos idénticos de los tres objetivos previamente normalizados. De esta forma, el punto no-dominado con la suma de menor valor fue seleccionado como el mejor.

De esta manera, el desempeño de cada instancia corresponde a la suma ponderada de las cuatro métricas previamente normalizadas, donde $\omega_1, \omega_2, \omega_3$ y ω_4 son pesos de igual valor. Asimismo, a diferencia de $best_{NDP}$, I_Δ y GD, las cuales son métricas donde un valor bajo implica una mejor solución, en el indicador I_{HV} se requiere hacer un ajuste para transformarlo en una métrica a minimizar (ver **Ecuación 30**).

$$Heu_{sol} = \omega_1 \cdot (1 - I_{HV}) + \omega_2 \cdot I_\Delta + \omega_3 \cdot I_{GD} + \omega_4 \cdot best_{NDP}. \quad (30)$$

Una vez obtenidos los resultados de las 1200 ejecuciones (una por configuración), se procede a la aplicación de la prueba ANOVA, la cual permite observar el efecto de diferentes parámetros en la calidad de las soluciones y así, identificar aquellos factores más relevantes. En este estudio se parte de una hipótesis nula H_0 (véase **Ecuación 31**), la cual asume que las medias poblaciones son idénticas, es decir, no son estadísticamente significativas. Asimismo, se cuenta con una hipótesis alternativa H_1 (véase **Ecuación 32**), la cual supone que al menos una media poblacional difiere del resto.

$$H_0: \mu_1 = \mu_2 = \cdots = \mu_k = \mu. \quad (31)$$

$$H_1: \exists \mu_j \neq \mu, j = \{1, 2, \ldots, K\}. \quad (32)$$

Un punto importante en este estudio es el nivel de significancia, también denotado como valor p o α, el cual representa la probabilidad de rechazar la hipótesis nula cuando es verdadera. En este trabajo se consideró un nivel de confianza del 95% (valor $p < 0.05$). A grandes rasgos, esto significa que es posible garantizar que las diferencias de los algoritmos comparados son estadísticamente significativas o no con una probabilidad del 95% o, dicho de otra manera, la probabilidad de que las diferencias hayan ocurrido por casualidad es de apenas el 5%.

Los experimentos fueron realizados con un Intel Core i5 @ 2.3Ghz, 8GB RAM 2133 MHz LPDDR3 con sistema operativo 64 bit macOS Mojave versión 10.14.4.

Las **Figuras 35-39** muestran las medias poblacionales de los factores con mayor significancia estadística para cada uno de los parámetros a sintonizar, donde se muestran las distancias relativas (véase **Ecuación 33**) desde las soluciones arrojadas (Heu_{sol}) a la mejor solución obtenida durante la prueba de todas las posibles combinaciones de parámetros ($Best_{sol}$).

$$\frac{Heu_{sol} - Best_{sol}}{Best_{sol}} \cdot 100\,. \qquad (33)$$

La **Figura 35** muestra los resultados obtenidos para el tamaño de la población, donde se observa que una población de dimensión 225 o bien, una malla cuadrada de 15x15 individuos es estadísticamente más significativa. La **Figura 36** presenta las gráficas para los operadores de recombinación, donde se muestra que el operador de cruce OX es el mejor entre los cuatro probados. Esto se debe principalmente a su capacidad para transmitir información en común del material genético de padres a descendientes. La **Figura 37**, presenta los resultados de la sintonización para los operadores de mutación, en la cual se observa que el operador de mutación "*scramble*" es el mejor, esto debido a su capacidad para generar cambios puntuales en el genotipo del individuo.

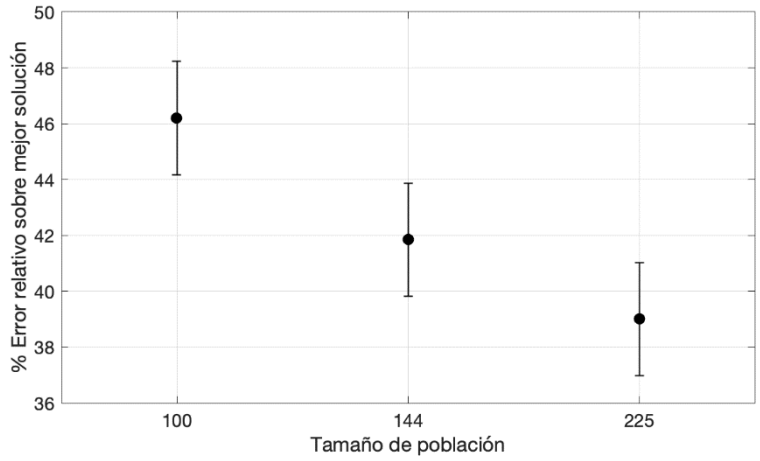


Figura 35. Medias poblaciones para los distintos tamaños de población, con un nivel de significancia o $p - value = 0.0002$.

La **Figura 38** presenta las gráficas para la probabilidad de mutación (P_m), donde la mejor probabilidad de que un individuo presente o no mutaciones estocásticas es del 30%. Asimismo, la **Figura 39** muestra los resultados para la probabilidad de cruzamiento (P_r), donde se observa que la probabilidad de que dos posibles padres o semillas puedan o no crear nuevo material genético es del 60%. Sin embargo, como se observa en la misma gráfica, las medias poblacionales arrojadas por los diversos factores no difieren, razón por la cual el análisis de varianzas arrojó un valor de $p = 0.3712 > 0.05$. Por tanto, al ser mayor que 5%, no es posible rechazar la hipótesis nula para este parámetro en particular.

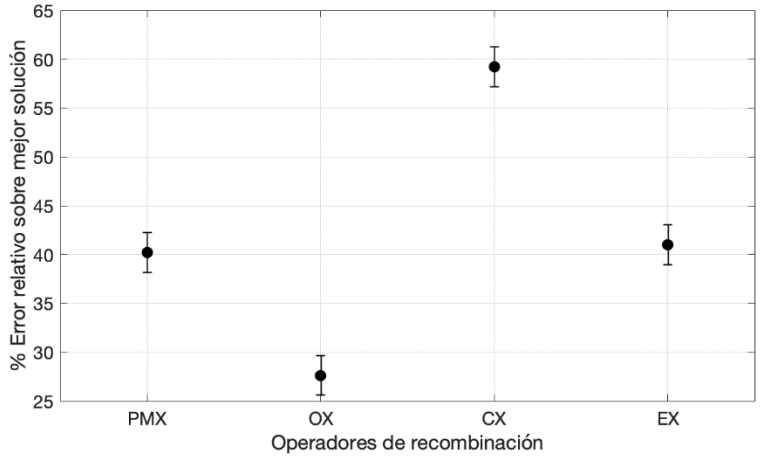


Figura 36. Medias poblaciones para los distintos operadores genéticos de recombinación implementados, con un nivel de significancia o $p - value = 0$.

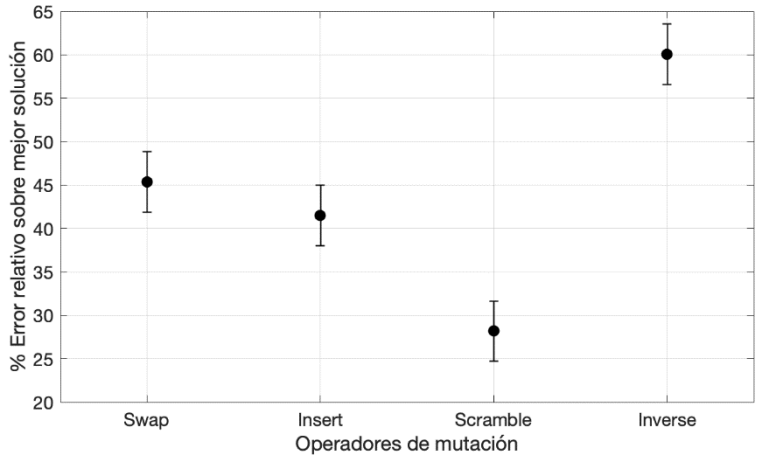


Figura 37. Medias poblaciones para los distintos operadores genéticos de mutación implementados, con un nivel de significancia o $p - value = 0$.

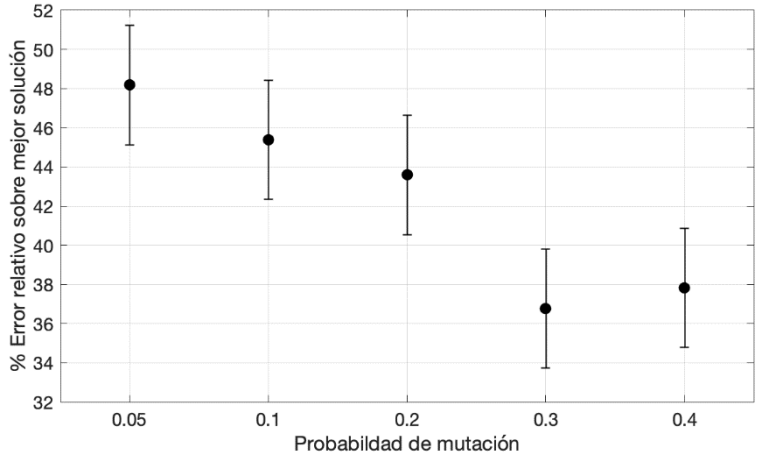


Figura 38. Medias poblaciones para diferentes probabilidades de mutación (P_m), con un nivel de significancia o $p - value = 0$.

De esta forma, tomando como referencia los resultados arrojados por el análisis estadístico previamente detallado, la **Tabla 4** resume la configuración de parámetros utilizada para el algoritmo genético celular. Se definió una población inicial de dimensión 225 (malla cuadrada de $15x15$ individuos), un torneo binario para la selección de padres, la evaluación de 25,000 funciones *"fitness"* como criterio de parada, el uso de OX como operador de recombinación con una probabilidad de cruzamiento de 90% y, un operador de mutación *"scramble"* con una probabilidad de 30%.

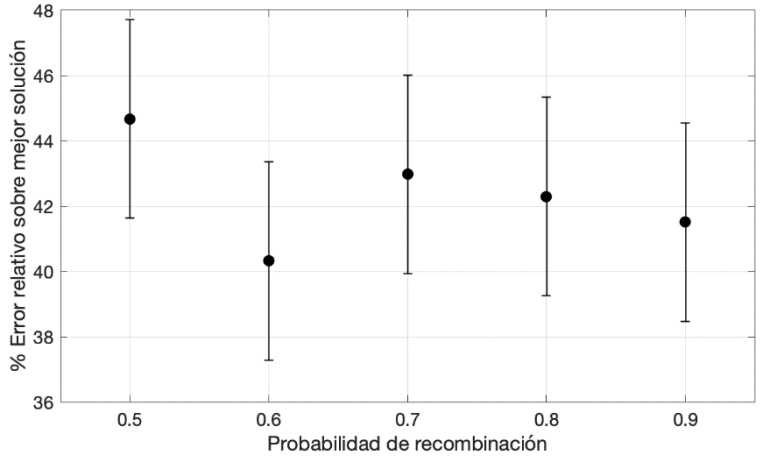


Figura 39. Medias poblaciones para diferentes probabilidades de recombinación (P_r), con un nivel de significancia o $p - value = 0.3712$.

Tabla 4. Parámetros de configuración de algoritmo genético celular (AGC).

Condición de parada	25,000 evaluaciones de función de aptitud
Tamaño de la población	225 individuos (malla cuadrada de 15x15 soluciones)
Tamaño del vecindario	8 vecinos circundantes
Operador de selección	Torneo binario (9,2)
Operador de recombinación	OX
Probabilidad de cruce	0.90
Operador de mutación	*Scramble*
Probabilidad de muta	0.30
Reemplazo	Reemplazo por ranking de dominancia y/o distancia de aglomeración
Estimador de densidad	Distancia de aglomeración o "*Crowding distance*"
Retroalimentación	20 individuos

No obstante, con el fin de realizar un análisis comparativo justo entre el algoritmo genético celular propuesto y los dos algoritmos representativos del estado del arte: NSGA-II y SPEA2, se realizó una sintonización de parámetros siguiendo la misma temática que en el estudio previamente detallado para estos últimos dos, definiendo tanto los operadores genéticos adecuados como la parametrización de los algoritmos en sí (véase **Tabla 5**), permitiendo explotar las capacidades de cada uno ellos para la problemática abordada en este trabajo de investigación.

Tabla 5. Configuración de parámetros utilizada en algoritmos NSGA-II y SPEA2.

	NSGA-II	**SPEA2**
Condición de parada	25,000 evaluaciones de función de aptitud	25,000 evaluaciones de función de aptitud
Tamaño de la población	100 individuos	100 individuos
Operador de selección	Torneo binario	Torneo binario
Operador de recombinación	OX	OX
Probabilidad de cruce	0.90	0.90
Operador de mutación	*Swap*	*Swap*
Probabilidad de muta	0.20	0.30
Reemplazo	Reemplazo por ranking de dominancia y/o distancia de aglomeración	Reemplazo por ranking de dominancia
Estimador de densidad	Distancia de aglomeración o "*Crowding distance*"	Mecanismo de truncamiento llamado "*método de enlace promedio*"
Retroalimentación		20 individuos

5.4 Resultados experimentales

Esta sección presenta los resultados obtenidos en los experimentos de evaluación de los algoritmos estudiados, analizados desde el punto de vista de la calidad de sus resultados, es decir, se cuantifica el rendimiento de los conjuntos aproximados arrojados por cada uno de ellos con respecto a los indicadores de desempeño hipervolumen (I_{HV}) y "*spread*" (I_{Δ}). Los experimentos de evaluación fueron realizados sobre la instancia del problema presentada en secciones precedentes. Se realizaron al menos treinta ejecuciones independientes (utilizando diferentes semillas para el generador de números aleatorios) de cada implementación. Los resultados se resumen y comentan a continuación.

En primera instancia, con el propósito de evidenciar de manera gráfica el conflicto entre los objetivos a optimizar. La **Figura 40** muestra la población inicial y final para una ejecución del algoritmo genético celular en la instancia del problema previamente mencionada (véase **Sección 5.2**); a su vez, se observa a grandes rasgos la magnitud del espacio de posibles soluciones, es decir, las posibles rutas para que un vehículo se desplace de un punto a otro. Del mismo modo, esta figura muestra gráficamente cómo el algoritmo evoluciona la población a medida que aumentan las generaciones, esto a través de cambios puntuales en cada uno de los individuos que representan una mejora en la calidad de cada una de las soluciones factibles para cada función objetivo.

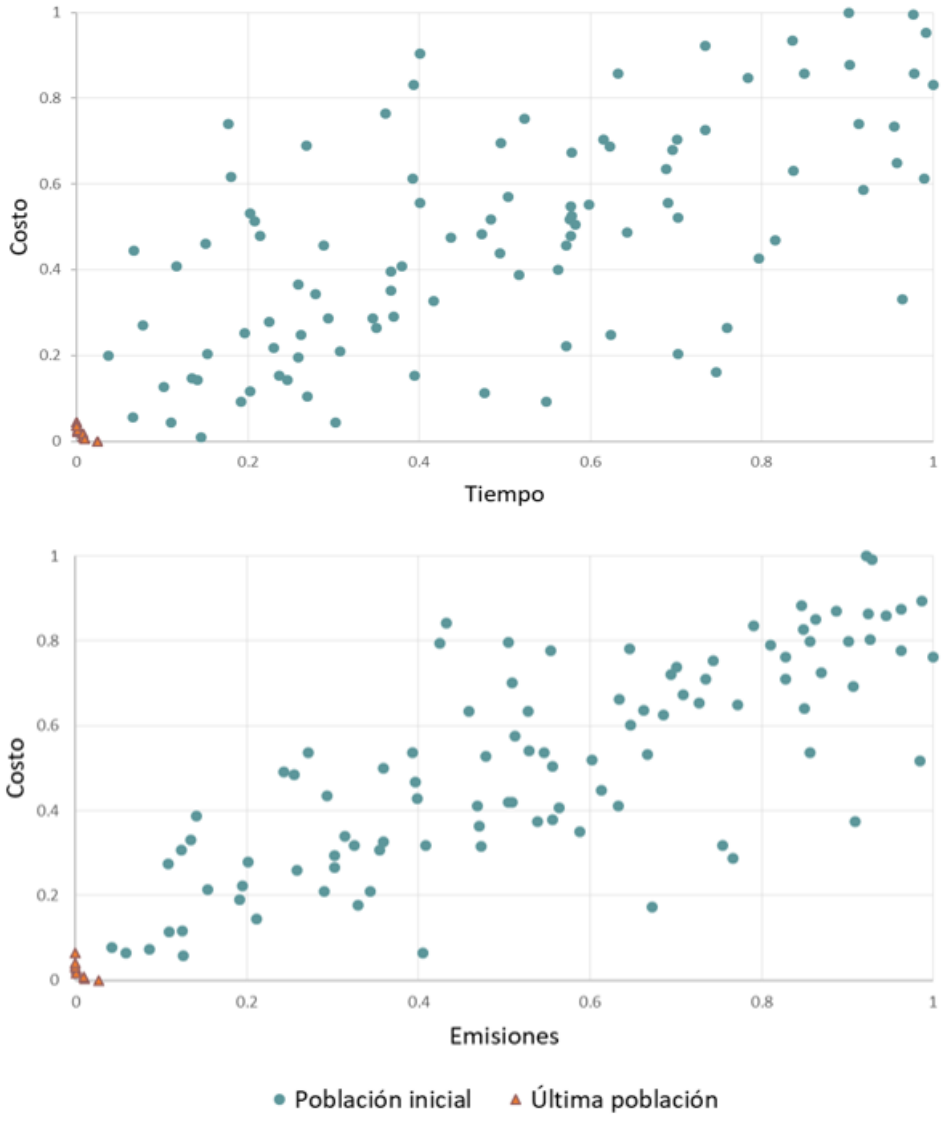


Figura 40. Conjunto de soluciones no-dominadas obtenidas por el algoritmo genético celular multiobjetivo MOCell para criterios de optimización en conflicto. Frente aproximado con I_{HV} = 0.518 en comparación a la población inicial para una ejecución con P_r = 0.90 y P_m = 0.30.

Una vez sintonizados los parámetros para el algoritmo genético celular propuesto, se estudia su comportamiento en el espacio objetivo. La **Figura 41** muestra el rendimiento de dicho algoritmo en términos del indicador de calidad I_{HV} para treinta ejecuciones independientes, se busca evidenciar un comportamiento que incremente el valor de I_{HV} en cada iteración, dado que se ha demostrado teóricamente que la maximización del hipervolumen es directamente proporcional a la obtención del frente óptimo.

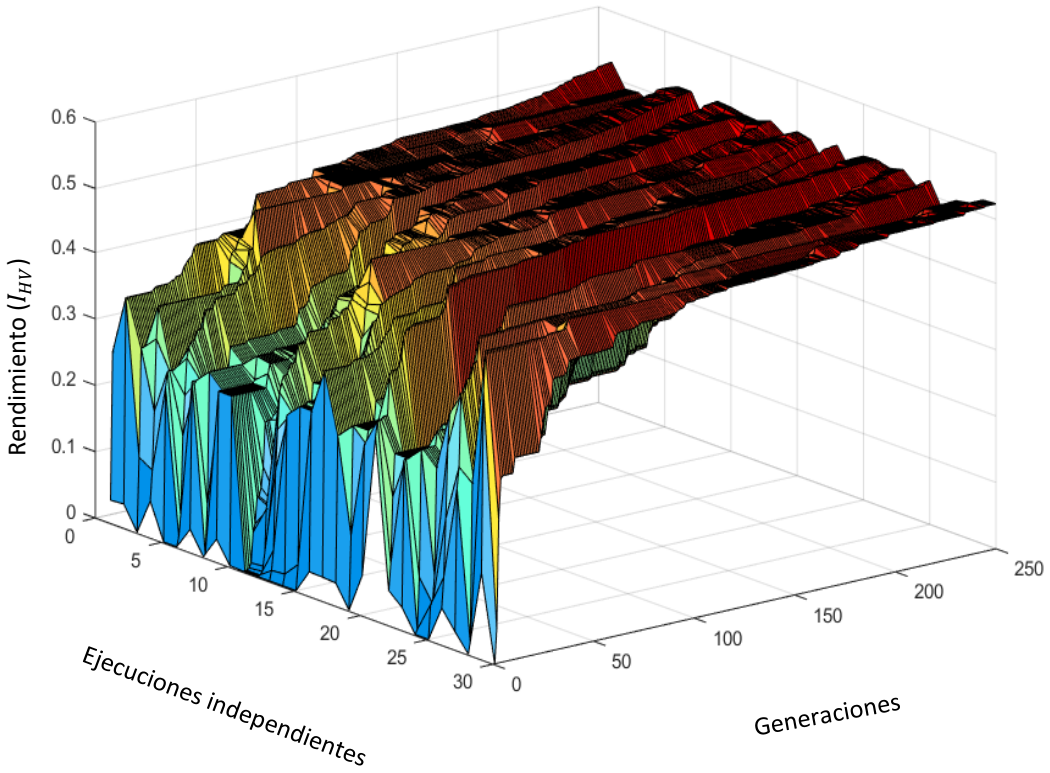


Figura 41. Rendimiento expresado en términos del indicador de calidad I_{HV} en 30 ejecuciones independientes del algoritmo genético celular a lo largo de 250 generaciones.

La **Figura 42** muestra el frente aproximado con mejor rendimiento generado por cada uno de los algoritmos implementados, cuyas parametrizaciones se detallan en la **Sección 5.3**. Además, se muestra el frente artificial de Pareto, el cual es construido a través de la combinación de todos los conjuntos de soluciones no-dominadas encontrados en el total de ejecuciones realizadas para cada uno de los algoritmos estudiados. Dicho frente aproximado de Pareto es objeto de referencia en las múltiples métricas de calidad utilizadas a lo largo de la evaluación experimental, permitiendo así comparar los resultados obtenidos por el MOEA propuesto con respecto a los arrojados por NSGA-II y SPEA2, esto en

términos de convergencia y dispersión de los frentes aproximados encontrados para el problema de enrutamiento de vehículos.

La **Tabla 6** muestra los resultados obtenidos por cada algoritmo en las métricas de calidad multiobjetivo definidas en la **Sección 5.1**. Para cada métrica se reporta el mejor valor, la media y la desviación estándar.

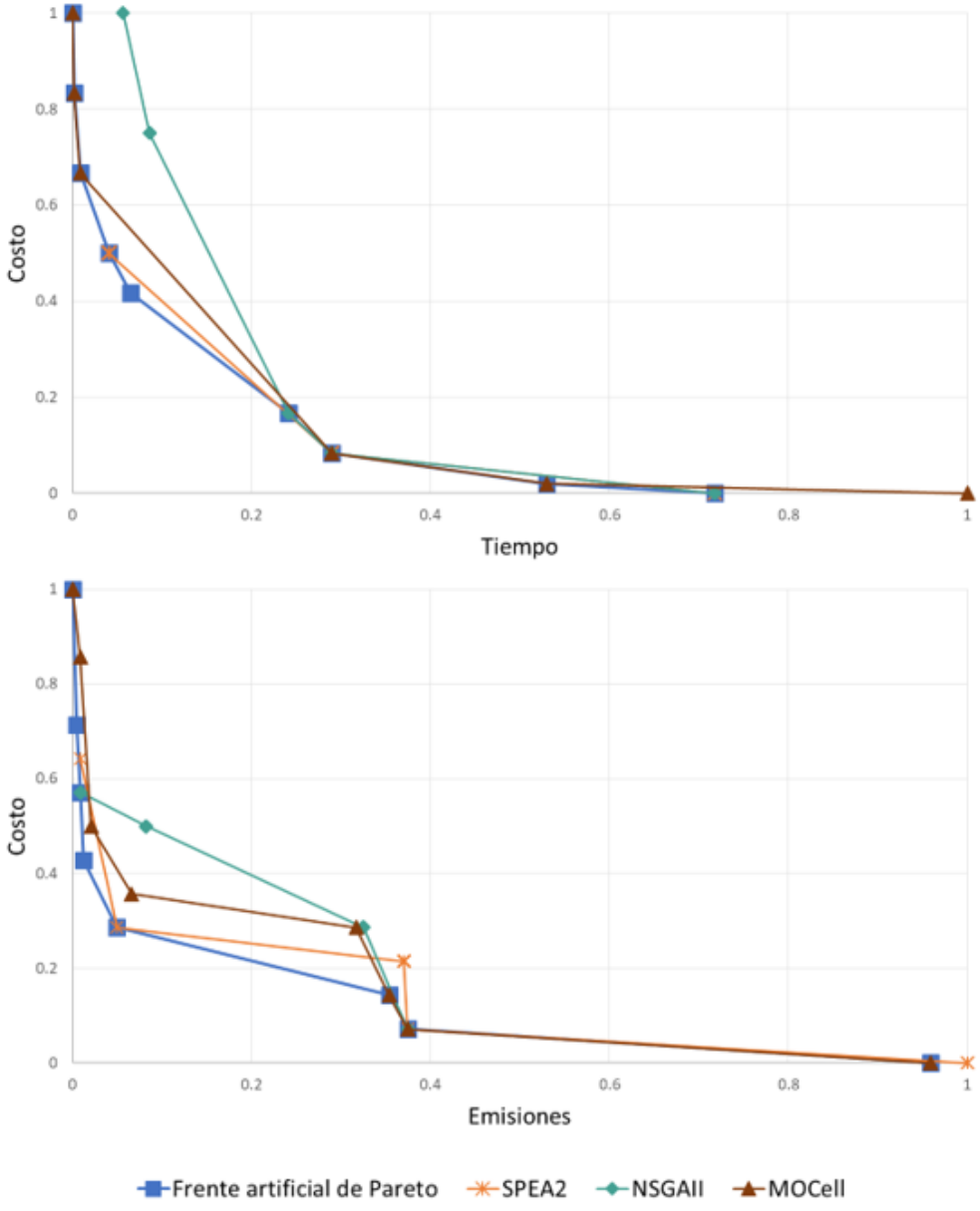


Figura 42. Mejor aproximación al frente artificial de Pareto obtenida por cada uno de los algoritmos con respecto al indicador de calidad Spread (I_Δ).

Tabla 6. Indicadores de calidad calculados para MOCell, NSGA-II y SPEA2.

Métrica		MOCell	NSGA-II	SPEA2
Hypervolume (I_{HV})	max	0.521	0.520	0.512
	media	**0.518**	0.515	0.420
	σ	0.0130	0.0145	0.0448
Spread (I_{Δ})	max	0.628	0.982	1.360
	media	**0.718**	1.760	1.790
	σ	0.0631	0.4490	0.2393

Por último, la **Figura 43** muestra un importante aspecto en los algoritmos evolutivos: la evolución del rendimiento sobre las generaciones. En esta figura se presenta la evolución en la calidad de las soluciones sobre las generaciones expresado en términos del indicador I_{HV}, observadas para cada uno de los algoritmos estudiados durante ejecuciones representativas sobre la instancia del problema presentada en secciones precedentes.

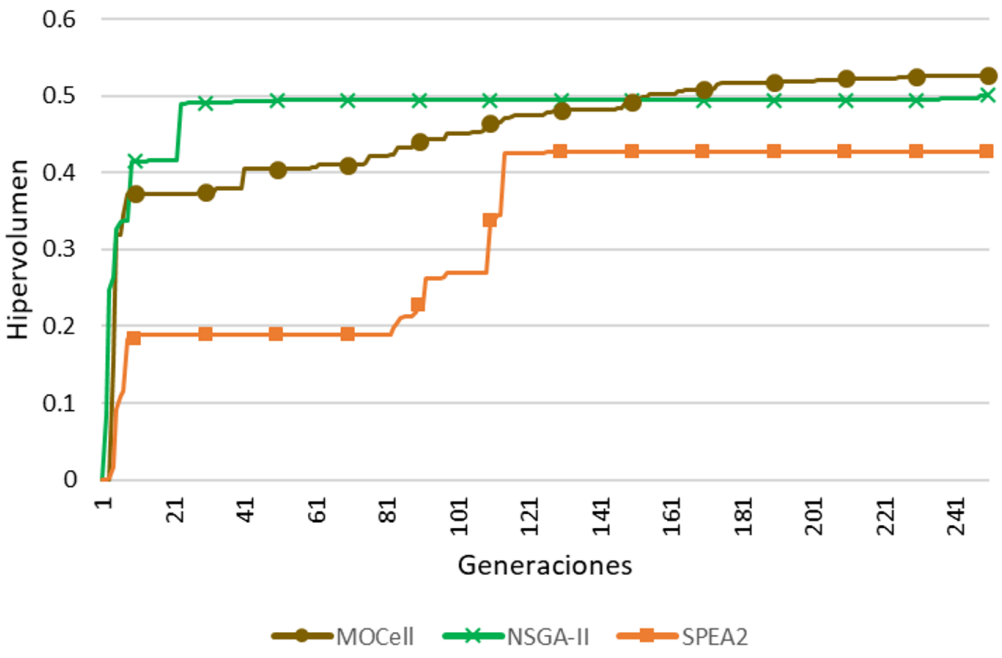


Figura 43. Evolución del rendimiento sobre las generaciones, expresado en términos del indicador de calidad I_{HV}.

5.5 Análisis y discusión de resultados

En términos generales, un buen desempeño por parte de un algoritmo evolutivo multiobjetivo consiste en generar soluciones no-dominadas que permitan converger hacia el frente de Pareto y que, a su vez, estén propagadas a lo largo del mismo. Siendo así, en esta sección se analizan los resultados arrojados en los experimentos de evaluación de los algoritmos estudiados con respecto a estos dos criterios, permitiendo identificar patrones de comportamiento característicos en cada uno de estos enfoques.

Se observa en las **Tablas 4** y **5** que cada uno de los algoritmos implementados opera con la configuración de parámetros que posibilita su mejor rendimiento para la problemática abordada. A través de esta sintonización de parámetros es posible percatarse de lo relevante que es el tamaño de la población para un algoritmo genético celular, a diferencia de NSGA-II y SPEA2 donde una población más grande no representa una mejora considerable en su desempeño. Dicha particularidad se atribuye principalmente al uso de una población de tipo celular, la cual además de permitir la explotación y exploración del espacio de posibles soluciones de una mejor manera, posibilita una lenta difusión del material genético inmerso y, por ende, una alta diversificación dentro de la población.

Un aspecto prevalente en estos tres MOEA es el hecho de que su implementación está basada en bloques implícitos, y a su vez, dependen en gran medida de sus mecanismos para la estimación de densidad. A pesar de dichas similitudes, los experimentos indicaron que el algoritmo genético celular MOCell obtiene mejores resultados en términos de convergencia, y claramente supera estadísticamente a los otros dos algoritmos multiobjetivo con respecto a la dispersión del conjunto de soluciones a lo largo del frente óptimo.

En la **Figura 40** se puede apreciar como el algoritmo genético celular tiene la capacidad de mover gran parte de la población inicial en dirección al frente aproximado de Pareto, a expensas de una notoria reducción en la cantidad de individuos diferentes en la población.

En la **Figura 42** se observa que para los objetivos en conflicto costo y cantidad de emisiones, MOCell fue el único algoritmo que encontró tanto soluciones óptimas para minimizar cada criterio por separado, como rutas que contemplan ambos objetivos, es decir, identificó puntos no-dominados posicionados en los extremos y a lo largo del frente aproximado de Pareto. Lo cual, desde una perspectiva práctica, permite a tomadores de decisiones seleccionar caminos que permitan optimizar cada criterio de manera

independiente o bien, optar por alternativas que generen buenos compromisos, reduciendo los costos asociados de trasladarse de un lugar a otro sin dejar de lado aspectos ecológicos.

Para discutir lo anterior, la **Tabla 6** presenta los mejores valores, el promedio y la desviación estándar obtenidos en las treinta ejecuciones independientes para cada uno de los algoritmos estudiados, con respecto a diversas métricas comúnmente utilizadas en la literatura, tales como hipervolumen y "*spread*". En dicha tabla es sencillo observar que los algoritmos NSGA-II y SPEA2 presentan dificultades a la hora de preservar la diversidad de sus soluciones. La comparativa muestra que el algoritmo implementado MOCell supera al resto de los MOEA en términos de dispersión, logrando un valor en I_{Δ} 145% mejor en el caso promedio que NSGA-II y 149% mejor que SPEA2. Por otro lado, en hipervolumen, los algoritmos NSGA-II y MOCell presentaron resultados similares, indicando que los conjuntos de soluciones no-dominadas encontradas por el algoritmo genético celular si bien presentan una mejor extensión a lo largo del frente aproximado, en términos de convergencia pueden llegar a ser semejantes.

Al analizar los resultados de la **Tabla 6**, es posible apreciar que el algoritmo evolutivo multiobjetivo SPEA2 muestra los peores resultados, notoriamente inferiores a los del resto. SPEA2 parece incapaz de manejar la complejidad intrínseca del problema de enrutamiento de vehículos, al menos para la instancia propuesta. Comportamiento hasta cierto punto inesperado, dado su mecanismo para la preservación de soluciones de frontera, y el uso de un archivo externo para el almacenamiento de soluciones globales no-dominadas. Asimismo, es posible destacar el buen desempeño del algoritmo genético de tipo celular para las condiciones y características de la problemática planteada. Predominando un rendimiento superior tanto en convergencia como en diversidad sobre NSGA-II y SPEA2, dos algoritmos bien conocidos en la literatura.

Para validar la información anterior, en la **Figura 43** es posible apreciar como MOCell y NSGA-II son capaces de encontrar individuos altamente adaptados en un número moderado de generaciones, alcanzando altos niveles de I_{HV} en un poco más de 100 generaciones. El algoritmo SPEA2 presenta el peor comportamiento, siguiendo un patrón más letárgico de evolución, estancándose debido a problemas de convergencia prematura. En los algoritmos NSGA-II y SPEA2 fue posible identificar una pérdida significativa de diversidad como el principal motivo por el cual detienen la evolución de su desempeño al avanzar las generaciones. De esta forma, considerando que el espacio de soluciones factibles del problema VRP tiene una dimensión reducida en comparación con el número de soluciones no-factibles, y que a medida que los algoritmos evolucionan moverse de un punto factible a otro factible se hace cada vez menos probable, la importancia

de un mecanismo para preservar la dispersión de las soluciones y permitir la exploración del espacio objetivo es evidente.

En este contexto, si bien MOCell evoluciona más lentamente, requiriendo más generaciones antes de encontrar resultados de calidad comparable, se debe principalmente a su mecanismo para la preservación de diversidad en las soluciones. Siendo así, de este análisis se puede concluir que la segmentación de la población en pequeños vecindarios constituye una mejor estrategia que utilizar una única población panmíctica sin restricción intrínseca para sus individuos al interactuar entre sí. Ya que al estudiar la evolución de los valores de I_{HV} sobre las generaciones, se detectó que NSGA-II si bien permite obtener individuos altamente adaptados rápidamente, converge prematuramente a óptimos locales. Por otro lado, la lenta difusión del material genético inmersa en MOCell permite introducir una notable fuente de diversidad que ayuda al proceso evolutivo del propio algoritmo.

Los resultados anteriores ponen de manifiesto claramente la superioridad del algoritmo genético celular como solución al problema propuesto, la pobre precisión alcanzada por el algoritmo multiobjetivo SPEA2, y la mejora poco significativa en la rapidez para identificar individuos adaptados de NSGA-II. Es posible suponer entonces que la implementación de AGC puede ser una herramienta precisa y eficiente para la planificación táctica en el enrutamiento de vehículos con penalizaciones ambientales. En términos generales, el enfoque propuesto converge al frente aproximado de Pareto y los conjuntos de soluciones no-dominadas mantienen une buena propagación a lo largo del mismo.

Capítulo 6. Conclusiones y trabajo futuro

Este capítulo resume los principales hallazgos y conclusiones resultantes del trabajo de investigación presentado en los capítulos precedentes. Asimismo, se presentan aquellas líneas de investigación que han dejado interesantes aspectos abiertos que merecen una profundización como continuación lógica en el futuro.

6.1 Conclusiones

Este trabajo de investigación presentó el diseño e implementación de un método de optimización basado en el algoritmo multiobjetivo genético celular MOCell para la resolución del problema de enrutamiento de vehículos con penalización ambiental, permitiendo disminuir el impacto ambiental en áreas altamente concurridas, al tiempo que se provee de rutas alternativas capaces de minimizar los costos asociados de trasladarse de un lugar a otro, esto bajo la premisa de que no siempre el camino más corto representa la mejor solución.

El algoritmo presentado minimiza simultáneamente tres importantes objetivos: tiempo, cantidad de emisiones y una penalización ambiental, donde esta última representa el costo implícito de desplazarse por un determinado segmento del mapa. Es decir, se obtiene un conjunto aproximado de soluciones no-dominadas, las cuales representan rutas alternativas que evitan moverse por zonas con un alto grado de contaminación y que, a su vez, minimizan el tiempo y la cantidad de emisiones al recorrer una determinada red de rutas. Como instancia del problema, se utilizó un grafo conexo dirigido compuesto por 6104 nodos, el cual representa la topología subyacente de la red de carreteras en la ciudad de Oldemburgo. Dicho grafo se segmentó en tres diferentes tipos de zonas, las cuales simbolizan áreas con distinta penalización ambiental.

Asimismo, se realizó una sintonización de parámetros a través de un análisis de varianzas multifactorial como parte del estudio experimental, donde se ejecutaron alrededor de 1200 corridas independientes del algoritmo para distintas configuraciones, permitiendo identificar aquella combinación de parámetros que posibilita encontrar soluciones potenciales de manera eficiente y a su vez, ofrece una mejor exploración y explotación en el espacio de posibles soluciones. Corroborando así que una elección adecuada de valores u operadores genéticos es directamente proporcional a la calidad de las manipulaciones estocásticas en

los genotipos, lo cual implica mejores aproximaciones al frente óptimo tanto en convergencia como en la dispersión de las soluciones no-dominadas.

Se realizó un exhaustivo análisis comparativo de los resultados obtenidos con respecto a diversos indicadores de calidad, tales como hipervolumen y "*spread*". El análisis exhibió un rendimiento competitivo por parte del algoritmo genético de tipo celular para las condiciones y características de la problemática presentada en términos de convergencia y diversidad, esto con respecto a NSGA-II y SPEA2: dos algoritmos evolutivos multiobjetivo bien conocidos en la literatura; logrando un valor en I_Δ 145% mejor en el caso promedio que NSGA-II y 149% mejor que SPEA2. Esto nos permite afirmar que los algoritmos genéticos de tipo celular pueden representar una herramienta útil y precisa para la planificación táctica en el enrutamiento de vehículos. De igual manera, posibilitó la identificación de numerosos patrones de comportamiento característicos de los diferentes algoritmos evolutivos considerados, que agregan conocimiento importante a utilizar al momento de decidir una continuación del trabajo.

El trabajo reportado en esta tesis resultó en un artículo de conferencia (Pulido et al., 2019), el cual aborda el tema incluido en este manuscrito. Además, durante su desarrollo fue posible establecer una importante colaboración con otros grupos de trabajo e investigación de la Universidad de la República de Uruguay y la Universidad Veracruzana, posibilitando un valioso intercambio de opiniones e ideas sobre la aplicabilidad de estas técnicas evolutivas para la resolución de problemas de optimización en el diseño de redes de transporte. El resultado del proceso de análisis de datos reportado en este documento fue empleado para abordar problemáticas en tópicos relacionados, lo que llevo a una serie de artículos de coautoría sobre el tema (Salas et al., 2019a, 2019b).

6.2 Trabajo futuro

En el marco del trabajo de investigación se estudiaron en profundidad los fundamentos de las técnicas de cómputo evolutivo, particularmente sobre el paradigma de los algoritmos genéticos, adquiriendo valiosos conocimientos sobre esta familia de metaheurísticas que han permitido una formación adecuada a un servidor. Esta tarea complementaria al diseño, implementación y análisis de algoritmos bio-inspirados de optimización multiobjetivo para la resolución del problema de enrutamiento de vehículos con penalización ambiental, permite establecer una base sólida para futuros trabajos desarrollados en el área.

Las principales líneas de trabajo futuro incluyen el abordar un estudio teórico y experimental para el uso de técnicas de segmentación de grafos que permitan limitar la búsqueda de rutas únicamente en áreas donde potencialmente se encuentren las mejores soluciones y así, posibilitar la ejecución del enfoque propuesto para instancias o mapas de mayor dimensión.

Otra línea interesante de trabajo futuro basada en la presente tesis, está encaminada a la inclusión de información vial en tiempo real para la caracterización de zonas, es decir, estimar la concurrencia en determinadas áreas del mapa y con base en ello, definir las penalizaciones ambientales para cada una de ellas. La integración del framework de desarrollo jMetal 5 con plataformas de datos en tiempo real como "*Spark*" (Cordero et al., 2016) facilita en gran medida este tipo de implementaciones. Si bien esta integración elevaría considerablemente el costo computacional, los algoritmos genéticos celulares poseen una propiedad capaz de mitigar dicho inconveniente, esta es la sencillez con la cual pueden ser paralelizados debido a la segmentación de su población en múltiples vecindarios. Aunado a la posibilidad de introducir fuentes de datos en *streaming,* esto permitiría identificar comportamientos que mejoren el desempeño no solo desde el punto de vista de la calidad de los resultados obtenidos, sino también en términos de mejora en la eficiencia computacional alcanzada.

Considerando que se identificó a la pérdida de diversidad como el principal motivo por el cual la mayoría de los algoritmos estudiados no fueron capaces de alcanzar mejores resultados, el estudio a futuro de los diferentes mecanismos de introducción de diversidad se manifiesta como una tarea lógica para mejorar los resultados obtenidos. Asimismo, el diseño e implementación de otro tipo de estrategias u operadores genéticos especializados que permitan explotar y explorar el espacio objetivo de mejor manera a partir de la experiencia obtenida en el presente trabajo, constituyen líneas claras de investigación a futuro.

Literatura citada

Alba, E., & Dorronsoro, B. (2008). *Cellular Genetic Algorithms*. *Operations Research/ Computer Science Interfaces Series* (Vol. 42). Boston, MA: Springer US. https://doi.org/10.1007/978-0-387-77610-1

Albino, V., Berardi, U., & Dangelico, R. M. (2014). Smart Cities: Definitions, Dimensions, Performance, and Initiatives. *Journal of Urban Technology*, *0*(1), 3–21. https://doi.org/https://doi.org/10.1080/10630732.2014.942092

Barrionuevo, J. M., Berrone, P., & Ricart Costa, J. E. (2015). Smart Cities, Sustainable Progress: Opportunities for Urban Development. *IESE Insight*. https://doi.org/10.15581/002.art-2152

Barth, M. J., Wu, G., & Boriboonsomsin, K. (2015). Intelligent Transportation Systems and Greenhouse Gas Reductions. *Current Sustainable/Renewable Energy Reports*. https://doi.org/10.1007/s40518-015-0032-y

Bell, J. E., & McMullen, P. R. (2004). Ant colony optimization techniques for the vehicle routing problem. *Advanced Engineering Informatics*. https://doi.org/10.1016/j.aei.2004.07.001

Bocquier, P. (2005). World urbanization prospects: An alternative to the UN model of projection compatible with the mobility transition theory. *Demographic Research*. https://doi.org/10.4054/DemRes.2005.12.9

Brinkhoff, T. (2002). A Framework for Generating Network-Based Moving Objects. *GeoInformatica*, *6*(2), 153–180. https://doi.org/10.1023/A:1015231126594

Chakraborty, B., Maeda, T., & Chakraborty, G. (2005). Multiobjective Route Selection for Car Navigation System using Genetic Algorithm. https://doi.org/10.1109/SMCIA.2005.1466971

Charnes, A., & Cooper, W. W. (1957). Management Models and Industrial Applications of Linear Programming. *Management Science*, *4*(1), 38–91. https://doi.org/10.1287/mnsc.4.1.38

Chourabi, H., Nam, T., Walker, S., Gil-Garcia, J. R., Mellouli, S., Nahon, K., … Scholl, H. J. (2012). Understanding smart cities: An integrative framework. *Proceedings of the Annual Hawaii International Conference on System Sciences*, 2289–2297. https://doi.org/10.1109/HICSS.2012.615

Clarke, G., & Wright, J. W. (1964). Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. *Operations Research*, *12*(4), 568–581. https://doi.org/10.1287/opre.12.4.568

Coello Coello, C. A. (2018). Multi-objective Optimization. In *Martí R., Panos P., Resende M. (eds) Handbook of Heuristics* (pp. 177–204). Springer, Cham. https://doi.org/10.1007/978-3-319-07124-4_17

Coello Coello, C. A., Lamont, G. B., & Van Veldhuizen, D. A. (2007). *Evolutionary Algorithms for Solving Multi-Objective Problems*. *Genetic and Evolutionary Computation* (Second). Springer US. https://doi.org/10.1007/978-0-387-36797-2

Cohon, J. L., & Marks, D. H. (1975). A review and evaluation of multiobjective programing techniques. *Water Resources Research*, *11*(2), 208–220. https://doi.org/10.1029/WR011i002p00208

Cordero, J. A., Nebro, A. J., Barba-González, C., Durillo, J. J., García-Nieto, J., Navas-Delgado, I., & Aldana-Montes, J. F. (2016). Dynamic Multi-Objective Optimization with jMetal and Spark: A Case Study. *Pardalos P., Conca P., Giuffrida G., Nicosia G. (Eds) Machine Learning, Optimization, and Big Data. MOD 2016. Lecture Notes in Computer Science*, *10122*, 106–117. https://doi.org/10.1007/978-3-319-51469-7_9

Dantzig, G. B., & Ramser, J. H. (2008). The Truck Dispatching Problem. *Management Science*, *6*(1), 80–91. https://doi.org/10.1287/mnsc.6.1.80

De Coensel, B., Can, A., Degraeuwe, B., De Vlieger, I., & Botteldooren, D. (2012). Effects of traffic signal coordination on noise and air pollutant emissions. *Environmental Modelling & Software*, *35*, 74–83. https://doi.org/10.1016/j.envsoft.2012.02.009

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182–197. https://doi.org/10.1109/4235.996017

Deb, Kalyanmoy, & Sundar, J. (2006). Reference point based multi-objective optimization using evolutionary algorithms. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation - GECCO '06* (p. 635). New York, New York, USA: ACM Press. https://doi.org/10.1145/1143997.1144112

Deb, Kalyanmoy, Thiele, L., Laumanns, M., & Zitzler, E. (2001). *Scalable Test Problems for Evolutionary Multiobjective Optimization*. *Evolutionary Multiobjective Optimization*. https://doi.org/10.1007/1-84628-137-7_6

Durillo, J. J., & Nebro, A. J. (2011). jMetal: A Java framework for multi-objective optimization. *Advances in Engineering Software*, *42*(10), 760–771. https://doi.org/10.1016/j.advengsoft.2011.05.014

Edgeworth, F. Y. (1881). Mathematical psychics: An essay on the application of mathematics to the moral sciences, *10*.

Eiben, A. E., & Smith, J. E. (2015). *Introduction to Evolutionary Computing*. *New York* (Vol. 2). https://doi.org/10.1162/evco.2004.12.2.269

Fagúndez, G., Massobrio, R., & Nesmachnow, S. (2014). Online taxi sharing optimization using evolutionary algorithms. In *Proceedings of the 2014 Latin American Computing Conference, CLEI 2014*. https://doi.org/10.1109/CLEI.2014.6965163

Fishburn, P. C. (1974). Exceptional Paper—Lexicographic Orders, Utilities and Decision Rules: A Survey. *Management Science*, *20*(11), 1442–1471. https://doi.org/10.1287/mnsc.20.11.1442

Fleischer, M. (2003). The Measure of Pareto Optima Applications to Multi-objective Metaheuristics. *Fonseca C.M., Fleming P.J., Zitzler E., Thiele L., Deb K. (Eds) Evolutionary Multi-Criterion Optimization. EMO 2003*, *2632*, 519–533. https://doi.org/10.1007/3-540-36970-8_37

Floyd, R. W. (1962). Algorithm 97: Shortest path. *Communications of the ACM*, *5*(6), 345. https://doi.org/10.1145/367766.368168

Gen, M., Altiparmak, F., & Lin, L. (2006). A genetic algorithm for two-stage transportation problem using priority-based encoding. *OR Spectrum*, *28*(3), 337–354. https://doi.org/10.1007/s00291-005-0029-9

Gen, M., Cheng, R., & Dingwei Wang. (2009). Genetic algorithms for solving shortest path problems. In *Proceedings of 1997 IEEE International Conference on Evolutionary Computation (ICEC '97)* (Vol. 187, pp. 401–406). Berlin, Heidelberg: IEEE. https://doi.org/10.1109/ICEC.1997.592343

Gen, M., Cheng, R., & Lin, L. (2008). *Network Models and Optimization*. London: Springer London. https://doi.org/10.1007/978-1-84800-181-7

Gen, M., Runwei Cheng, & Dingwei Wang. (1997). Genetic algorithms for solving shortest path problems (pp. 401–406). https://doi.org/10.1109/icec.1997.592343

Girianna, M., & Benekohal, R. (2002). Dynamic Signal Coordination for Networks with Oversaturated Intersections. *Transportation Research Record: Journal of the Transportation Research Board*, *1811*, 122–130. https://doi.org/10.3141/1811-15

Glover, F. (1989). Tabu Search. *ORSA Journal on Computing*, *1*(3), 190–206. https://doi.org/10.1287/ijoc.1.3.190

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning*. (I. Addison-Wesley Longman Publishing Co., Ed.) (1st ed.). Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc. https://doi.org/10.5860/CHOICE.27-0936

Gordon, R. (2016). *Intelligent Transportation Systems*. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-14768-0

Graham, R. L., Lawler, E. L., Lenstra, J. K., & Kan, A. H. G. R. (1979). Optimization and Approximation in Deterministic Sequencing and Scheduling: a Survey. *Annals of Discrete Mathematics*, *5*, 287–326. https://doi.org/10.1016/S0167-5060(08)70356-X

Guzek, M., Pecero, J. E., Dorronsoro, B., & Bouvry, P. (2014). Multi-objective evolutionary algorithms for energy-aware scheduling on distributed computing systems. *Applied Soft Computing*, *24*, 432–446. https://doi.org/10.1016/J.ASOC.2014.07.010

Haimes, Y. V., Lasdon, L. S., & Wismer, D. A. (1971). On a Bicriterion Formulation of the Problems of Integrated System Identification and System Optimization. *IEEE Transactions on Systems, Man, and Cybernetics*, *SMC-1*(3), 296–297. https://doi.org/10.1109/TSMC.1971.4308298

Ijiri., Y. (1965). Management Goals and Accounting for Control. North-Holland, Amsterdan.

Kanoh, H. (2007). Dynamic route planning for car navigation systems using virus genetic algorithms. *International Journal of Knowledge-Based and Intelligent Engineering Systems*, *11*, 65–78. https://doi.org/10.3233/KES-2007-11105

Kim, G., Ong, Y. S., Heng, C. K., Tan, P. S., & Zhang, N. A. (2015). City Vehicle Routing Problem (City VRP): A Review. *IEEE Transactions on Intelligent Transportation Systems*. https://doi.org/10.1109/TITS.2015.2395536

Lawrence, D. (1991). Handbook of genetic algorithms. *Van Nostrand Reinhold, New York*.

Lee, W.-H., Tseng, S.-S., & Tsai, S.-H. (2009). A knowledge based real-time travel time prediction system for urban network. *Expert Systems with Applications*, *36*(3), 4239–4247. https://doi.org/10.1016/J.ESWA.2008.03.018

Lenstra, J. K., & Kan, A. H. G. R. (1981). Complexity of vehicle routing and scheduling problems. *Networks*, *11*(2), 221–227. https://doi.org/10.1002/net.3230110211

Liao, T.-Y., & Hu, T.-Y. (2011). An object-oriented evaluation framework for dynamic vehicle routing problems under real-time information. *Expert Systems with Applications*, *38*(10), 12548–12558. https://doi.org/10.1016/J.ESWA.2011.04.041

Massobrio, R., Fagúndez, G., & Nesmachnow, S. (2014). A parallel micro evolutionary algorithm for taxi sharing optimization. *VII ALIO/EURO Workshop on Applied Combinatorial Optimization*, 1–6. https://doi.org/10.13140/RG.2.1.3047.7925

Mengyin Fu, Jie Li, & Zhihong Deng. (2004). A practical route planning algorithm for vehicle navigation system. In *Fifth World Congress on Intelligent Control and Automation (IEEE Cat. No.04EX788)* (Vol. 6, pp. 5326–5329). IEEE. https://doi.org/10.1109/WCICA.2004.1343742

Messac, A., Ismail-Yahaya, A., & Mattson, C. A. (2003). The normalized normal constraint method for generating the Pareto frontier. *Structural and Multidisciplinary Optimization*, *25*(2), 86–98. https://doi.org/10.1007/s00158-002-0276-1

Miettinen, K. (1998). *Nonlinear Multiobjective Optimization* (Vol. 12). Boston, MA: Springer US. https://doi.org/10.1007/978-1-4615-5563-6

Minister of Infrastructure and the Environment. (2016). Smart Mobility Building. Retrieved July 7, 2018, from www.rijksoverheid.nl/ienm

Müller-Hannemann, M., & Schnee, M. (2007). Finding All Attractive Train Connections by Multi-criteria Pareto Search. In *Algorithmic Methods for Railway Optimization* (pp. 246–263). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-74247-0_13

Nambiar, R., Shroff, R., & Handy, S. (2018). Smart cities: Challenges and opportunities. In *2018 10th International Conference on Communication Systems & Networks (COMSNETS)* (pp. 243–250). IEEE. https://doi.org/10.1109/COMSNETS.2018.8328204

Nebro, A. J., Durillo, J. J., Luna, F., Dorronsoro, B., & Alba, E. (2009). MOCell: A cellular genetic algorithm for multiobjective optimization. *International Journal of Intelligent Systems*, *24*(7), 726–746. https://doi.org/10.1002/int.20358

Nha, V. T. N., Djahel, S., & Murphy, J. (2012). A comparative study of vehicles routing algorithms for route planning in smart cities. In *2012 First International Workshop on Vehicular Traffic Management for Smart Cities (VTM)* (pp. 1–6). IEEE. https://doi.org/10.1109/VTM.2012.6398701

Nunes, J., Matos, L., & Trigo, A. (2011). Taxi Pick-Ups Route Optimization Using Genetic Algorithms. In *LNCS* (Vol. 6593, pp. 410–419). https://doi.org/10.1007/978-3-642-20282-7_42

Olague, G. (2016). *Evolutionary Computer Vision*. Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-43693-6

Osyczka, A. (1978). An approach to multicriterion optimization problems for engineering design. *Computer Methods in Applied Mechanics and Engineering*, *15*(3), 309–333. https://doi.org/10.1016/0045-7825(78)90046-4

Osyczka, A. (1984). *Multicriterion optimization in engineering with FORTRAN programs*. E. Horwood.

Pallottino, S., & Scutellà, M. G. (1997). Shortest Path Algorithms in Transportation models: classical and innovative aspects Shortest Path Algorithms in Transportation models: classical and innovative aspects.

Pareto, V. (1896). *Cours d'économie politique* (Vol. 1). Librairie Droz.

Peña, D., & Tchernykh, A. (2017). *Optimización multiobjetivo para la planificación de transporte público aplicando técnicas metaheurísticas*. Centro de Investigación Científica y de Educación Superior de Ensenada, Baja California.

Peña, D., Tchernykh, A., Nesmachnow, S., Massobrio, R., Feoktistov, A., Bychkov, I., … Garichev, S. N. (2018). Operating cost and quality of service optimization for multi-vehicle-type timetabling for urban bus systems. *Journal of Parallel and Distributed Computing*. https://doi.org/10.1016/j.jpdc.2018.01.009

Peña, D., Tchernykh, A., Radchenko, G., Nesmachnow, S., Ley-Flores, J., & Nazariega, R. (2017). Multiobjective Optimization of Greenhouse Gas Emissions Enhancing the Quality of Service for Urban Public Transport Timetabling. In *2017 IVth International Conference on Engineering and Telecommunication (EnT)* (Vol. 70, pp. 114–118). Moscow: IEEE. https://doi.org/10.1109/ICEnT.2017.31

Possel, B., Wismans, L. J. J., Van Berkum, E. C., & Bliemer, M. C. J. (2018). The multi-objective network design problem using minimizing externalities as objectives: comparison of a genetic algorithm and simulated annealing framework. *Transportation*. https://doi.org/10.1007/s11116-016-9738-y

Pulido, L. B., Tchernykh, A., Nesmachnow, S., Salas, A., Avetisyan, A., Castro, H. E., & Barrios, C. J. (2019). Multi-Objective Optimization of Vehicle Routing with Environmental Penalty. In *10th International Supercomputing Conference in Mexico*.

Rao, S. S. (2009). *Engineering optimization: theory and practice* (4th ed).

Riquelme, N., Von Lucken, C., & Baran, B. (2015). Performance metrics in multi-objective optimization. In *2015 Latin American Computing Conference (CLEI)* (pp. 1–11). IEEE. https://doi.org/10.1109/CLEI.2015.7360024

Rosenberg, R. S. (1960). *Simulation of genetic populations with biochemical properties*. University of Michigan, Ann Arbor, Michigan, USA.

Ruiz, R., & Maroto, C. (2006). A genetic algorithm for hybrid flowshops with sequence dependent setup times and machine eligibility. *European Journal of Operational Research*, *169*(3), 781–800. https://doi.org/10.1016/j.ejor.2004.06.038

Sadeghi, J., Sadeghi, S., & Niaki, S. T. A. (2014). A hybrid vendor managed inventory and redundancy allocation optimization problem in supply chain management: An NSGA-II with tuned parameters. *Computers & Operations Research*, *41*, 53–64. https://doi.org/10.1016/j.cor.2013.07.024

Salas, A., Tchernykh, A., Nesmachnow, S., Pulido, L. B., Cuenca, J. M., Vicente, B., & Maldonado, C. S. (2019a). Transportation System Real-Time Re-Organization Due to Civil Protection Events. In *10th International Supercomputing Conference in Mexico*.

Salas, A., Tchernykh, A., Nesmachnow, S., Pulido, L. B., Vicente, B., Cuenca, J. M., & Maldonado, C. S. (2019b). Multi-Objective Evolutive Multi-Thread Path-Planning Algorithm considering Real-Time Extreme Weather Events. In *10th International Supercomputing Conference in Mexico*.

Schaffer, J. D. (1985). *Multiple Objective Optimization with Vector Evaluated Genetic Algorithms*. Vanderbilt University, Nashville, Tennessee.

Schaffer, J. D. (1986). *Some experiments in machine learning using vector evaluated genetic algorithms*. Vanderbilt University.

Serafini, P. (1994). Simulated Annealing for Multi Objective Optimization Problems. In *Multiple Criteria Decision Making* (pp. 283–292). New York, NY: Springer New York. https://doi.org/10.1007/978-1-4612-2666-6_29

Sharma, S., & Mathew, T. V. (2011). Multiobjective network design for emission and travel-time trade-off for a sustainable large urban transportation network. *Environment and Planning B: Planning and Design*. https://doi.org/10.1068/b37018

Solich, R. (1969). Linear Programming Problem with Several Objective Functions. In *Przeglad Statystyczny* (pp. 16:24–30).

Tatomir, B., Rothkrantz, L. J. M., & Suson, A. C. (2009). Travel time prediction for dynamic routing using Ant Based Control. In *Proceedings of the 2009 Winter Simulation Conference (WSC)* (pp. 1069–1078). IEEE. https://doi.org/10.1109/WSC.2009.5429648

Washburn, D., & Sindhu, U. (2010). Making Leaders Successful Every Day Helping CIOs Understand "Smart City" Initiatives. Retrieved from www.forrester.com.

Wierzbicki, A. P. (1980). The Use of Reference Objectives in Multiobjective Optimization (pp. 468–486). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-48782-8_32

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, *1*(1), 67–82. https://doi.org/10.1109/4235.585893

Zadeh, L. (1963). Optimality and non-scalar-valued performance criteria. *IEEE Transactions on Automatic Control*, *8*(1), 59–60. https://doi.org/10.1109/TAC.1963.1105511

Zavala, G., Nebro, A. J., Luna, F., & Coello Coello, C. A. (2016). Structural design using multi-objective metaheuristics. Comparative study and application to a real-world problem. *Structural and Multidisciplinary Optimization*, *53*(3), 545–566. https://doi.org/10.1007/s00158-015-1291-3

Zitzler, E., Deb, K., & Thiele, L. (2000). Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, *8*(2), 173–195. https://doi.org/10.1162/106365600568202

Zitzler, E., Laumanns, M., & Thiele, L. (2001). *SPEA2: Improving the Strength Pareto Evolutionary Algorithm*.

Zitzler, E., & Thiele, L. (1998). An evolutionary algorithm for multiobjective optimization: the strength Pareto approach. *TIK-Report*, *43*. https://doi.org/https://doi.org/10.3929/ethz-a-004288833