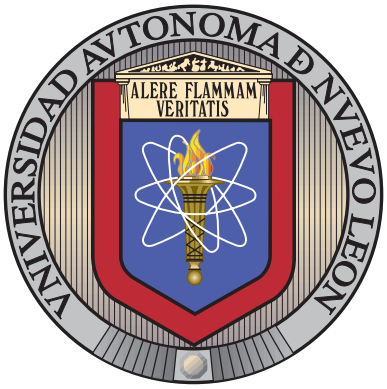
Universidad Autónoma de Nuevo León

Facultad de Ciencias Físico Matemáticas

División de estudios de posgrado



Aplicaciones de la programación matemática: Programación Binivel y Metaheurísticas

por

Carlos Eduardo Corpus Cardona

en opción al grado de

Doctor en Ciencias

con orientación en matemáticas.

San Nicolás de los Garza, Nuevo León — Septiembre 2024

Universidad Autónoma de Nuevo León

Facultad de Ciencias Físico Matemáticas

División de estudios de posgrado

Aplicaciones de la programación matemática: Programación Binivel y Metaheurísticas

por

Carlos Eduardo Corpus Cardona

en opción al grado de

Doctor en Ciencias

con orientación en matemáticas

San Nicolás de los Garza, Nuevo León Septiembre 2024

Universidad Autónoma de Nuevo León
Facultad de Ciencias Físico Matemáticas
División de estudios de posgrado

Los miembros del Comité de Tesis recomendamos que la Tesis "Aplicaciones de la programación matemática: Programación Binivel y Metaheurísticas", realizada por el alumno Carlos Eduardo Corpus Cardona, con número de matrícula 1585468, sea aceptada para su defensa como opción al grado de Doctor en Ciencias con Orientanción en Matemáticas.

El Comité de Tesis

Dr. José Fernando Camacho Vallejo
Director

Dr. Víctor Blanco Izquierdo
Revisor

Dr. Juan Guillermo Villegas
Revisor

Dr. Omar Jorge Ibarra Rojas
Revisor

Dr. Álvaro Eduardo Cordero Franco
Revisor

Vo. Bo.

Dr. Omar Jorge Ibarra Rojas
División de estudios de posgrado

San Nicolás de los Garza, Nuevo León, Septiembre 2024

Agradecimientos

La siguiente tesis no habría sido posible sin el apoyo no solo de mi coordinador de tesis, sino también de todas las personas que en algún momento me dieron palabras de aliento, consejos o se preocuparon por mí durante este viaje. Esta tesis también es de ustedes.

Quiero agradecer primero al Dr. Fernando Camacho, mi director de tesis. Por los consejos, no solo en lo académico, sino también en lo personal. Por brindarme su conocimiento, sus comentarios, las llamadas de atención, las críticas y los ánimos que me han ayudado a desarrollarme en muchos aspectos. Gracias, Doc.

También quiero agradecer a cada uno de esos mentores y amigos con los que tuve la oportunidad de trabajar. Al Dr. Juan Guillermo de la Universidad de Antioquía, en quien encontré no solo una persona trabajadora y sencilla, sino un amigo para toda la vida. Al Dr. Víctor Blanco de la Universidad de Granada, con quien tuve la oportunidad de reencontrarme, siempre con la intención de ayudar y con esa actitud de investigador rockstar. Gracias a ambos por su sabiduría.

Agradezco a mis amigos del posgrado: Karla, Yuli, con quienes siempre se podía echar un buen chisme después del trabajo. A Vale y Yuli por siempre andar con una sonrisa, contagiando felicidad. Al Dr. Omar y al Dr. Cordero por siempre buscar apoyarme con algún consejo o resolver alguna duda.

También a Sahori y Scarlett por ser dos grandes amigas, confidentes y madres de mis sobrinos favoritos. A mis compas, Toño, Marco, Bryan y Eder, con quienes siempre podré contar. A mis queridos amigos, Diana, Adrián y Jorge, gracias por preocuparse por mí en todo momento. Y a mis exalumnos, que reafirmaron mi deseo de dedicarme a esto por el resto de mi vida.

Agradezco a los amigos que conocí brevemente, pero que espero encontrar siempre haciendo lo que nos gusta... salir y tomar con la excusa de que estamos en un congreso. Pier, Francesca, Andreana, thank you for being my first friends during the course back in September, hope to see you soon. A Alberto, Paco y Gabriel por hacerme sentir bienvenido como si me hubieran conocido toda la vida. Y por último, pero no menos

importante, a Miguel y a María, quienes cuando llegué a Granada se encargaron de mí, permitiéndome conocer lugares que jamás pensé visitar,por su ayuda, por su preocupación, por su amistad, cariño y por abrirme las puertas de su piso, siempre tendrán un amigo en México. Se los agradezco siempre. Y a los que no pude mencionar porque se me acaba el espacio pero podría escribirles párrafos, sepan que siempre pienso en ustedes: Caro, Leslie, Gaby, Ana, Mateo, Alejo, Katherine, Nico, Branco, Juan, Carlos, muchas gracias por siempre apoyarme, agradezco mucho haberlos conocido.

Quiero agradecer a mi familia. Mamá, gracias por haberme apoyado toda la vida, sin ti no sabría dónde estaría en estos momentos. A mis hermanas: Vicky, por lo empalagosa que eres pero siempre preocupándote por mí, y Lucía, con quien creo que somos como dos gotas de agua pero al mismo tiempo muy diferentes. Las quiero a ambas. A mi papá, te fuiste muy pronto, pero espero que estés orgulloso dondequiera que estés. Cada uno de ustedes ha llenado y seguirá llenando mi vida.

Agradezco a la Universidad Autónoma de Nuevo León, en especial a la Facultad de Ciencias Físico Matemáticas y al Centro de Investigación en Ciencias Físico Matemáticas, por todo el apoyo brindado durante mi tiempo allí. También gracias al Consejo Nacional de Ciencia y Tecnología por sus apoyos económicos durante esta aventura.

Y finalmente, quiero agradecerme a mí mismo. No fue fácil, pero aquí estás, lo lograste. Hubo momentos en los que no podías más, noches de desvelo, incertidumbre y miedo, pero siempre con la certeza de lo que querías lograr. Felicidades, llegaste al final. Ahora queda lo más importante, terminar.

Índice general

Capítulo 1

Introducción

Muchas situaciones reales implican decisiones jerárquicas donde un nivel superior influye en las decisiones de un nivel inferior. Firmas con distinta posición en el mercado, gobierno y ciudadanos, redes de transporte y usuarios. Siempre que se tenga una situación donde se tenga un líder y un seguidor, la programación binivel proporciona un marco natural para modelar estas situaciones.

La programación binivel es un área de la programación matemática que desarrollado una importancia muy grande en los últimos años, modelado de situaciones complejas, desarrollo de nuevas técnicas para resolver estos problemas y el continuo desarrollo de la teoría detrás de esta, ha captado el interés de grandes investigadores.

Sin embargo, aún hay mucho que hacer, es por ello que esta tesis doctoral buscaremos seguir expandiendo el conocimiento detrás del área a través de distintos enfoques. Desde un análisis bibliométrico de literatura relacionada con metaheurísticas para problemas de programación binivel, desarrollo de algoritmos metaheurísticos en distintos contextos, como problemas de localización de hubs o de localización de plantas competitivas. Además de la importancia de los distintos enfoques al resolver problemas de programación binivel multi-objetivo.

1.1 Programación Binivel

En la vida real, existen muchas situaciones que involucran relaciones jerárquicas entre dos tomadores de decisiones. Por ejemplo, consideremos una situación en la que una autoridad establece los precios de peaje para una red de carreteras y los usuarios son libres de elegir las rutas más convenientes; o imaginemos una situación en la que una marca internacional abre nuevas tiendas y la marca local reacciona abriendo nuevas tiendas con el objetivo de mantener su cuota de mercado. En general, un tomador de decisiones de

nivel superior busca optimizar su propia función objetivo seleccionando una decisión que limite el espacio de decisiones de los tomadores de decisiones de nivel inferior, a partir del cual el tomador de decisiones de nivel inferior debe seleccionar su mejor decisión (en términos de su propia función objetivo). En esta relación jerarquizada, la decisión tomada a nivel inferior afecta el espacio de decisiones y/o la función objetivo del nivel superior [184], [96], [168].

Este tipo de problemas fueron descritos como programas matemáticos con un problema de optimización en sus restricciones [50]. Unos años más tarde, el término "programación binivel"fue introducido por [75]. Debido a su estrecha relación con los juegos líder-seguidor que comúnmente aparecen en un contexto económico [328], la terminología involucrada en los juegos de Stackelberg se utiliza en los problemas binivel. En particular, el nivel superior se asocia con un líder y el nivel inferior se asocia con un seguidor.

Una estructura general de un problema de programación binivel se presenta a continuación:

$$\max_{x,y} \quad F(x,y) \tag{1.1}$$

$$\text{s.t} \quad G(x,y) \leq b_1 \tag{1.2}$$

donde para un x fija, la variable y debe ser la solución optima del siguiente problema de optimización

$$\max_{y} \quad f(x,y) \tag{1.3}$$

$$\text{s.t.} \quad g(x,y) \leq b_2 \tag{1.4}$$

donde $x \in X$, $y \in Y, F : X \times Y \to \mathbb{R}$ y $f : X \times Y \to \mathbb{R}$ y además $b_1 \in \mathbb{R}^p, b_2 \in \mathbb{R}^q$

El problema definido por las ecuaciones (1.1 - 1.4) representa un caso general de un problema de programación binivel, el cual se sabe que incluso en su forma más simple (es decir donde el nivel superior e inferior son ambos lineales) es clasificado como NP-completo [161].

Recientemente, aplicaciones jerárquicas bajo una perspectiva de optimización binivel aparecen en muchos contextos diversos. Por ejemplo, en la logística marítima, un opera-

dor portuario maximiza su ganancia, y los transportistas minimizan sus propios costos de ruta [338]. En las cadenas de suministro sostenibles, el gobierno proporciona incentivos a las empresas para minimizar las emisiones de gases de efecto invernadero, mientras que las empresas buscan maximizar su tasa de producción [72]. En la logística humanitaria, una organización sin fines de lucro busca optimizar la equidad al entregar ayuda, y las autoridades en los refugios la distribuyen al costo mínimo [133]. En las cadenas de suministro agrícolas, un productor planifica su cosecha, y un mayorista decide la cantidad a adquirir del productor [13]. Como se puede observar, diferentes partes interesadas están involucradas en las cadenas de suministro, y existe una jerarquía entre ellas. Sin embargo, la optimización binivel también se puede aplicar para modelar problemas en contextos que no sean cadenas de suministro, como problemas de interdicción [201], congestión de tráfico [390], localización competitiva de instalaciones [28], aprendizaje automático [210], [223], entre otros.

Nótese que muchas de las aplicaciones de la vida real no se modelan como problemas binivel donde ambos niveles son lineales. Los problemas binivel pueden involucrar formulaciones multi-objetivo, enteras, no lineales, estocásticas, bi-lineales u otras en cada uno de los dos niveles de decisión jerarquizados. Por lo tanto, los enfoques genéricos de solución exacta son escasos y se limitan a las características específicas de cada problema binivel. Por ejemplo, el enfoque exacto más común al resolver un problema binivel consiste en reformularlo en un problema de un solo nivel equivalente. Esto se logra basándose en las condiciones necesarias y suficientes de Karush – Kuhn – Tucker (KKT) asociadas con el problema de nivel inferior [185], [51], [66], [61]. Sin embargo, esa reformulación solo es posible cuando el nivel inferior es convexo. Además, cuando se logra una reformulación de un solo nivel, surgen términos de gran-M, que deben seleccionarse cuidadosamente para evitar eliminar soluciones factibles del problema binivel. Esta no es una tarea fácil, como se muestra en [272] y [177]. Por lo tanto, la aplicabilidad de este enfoque de reformulación es limitada en muchas aplicaciones de la vida real.

Algunos enfoques exactos se basan en la enumeración de vértices [76], [40], [253] y en una estrategia de penalización aplicada al problema de nivel inferior [9], [32], [22], [26]. Otros enfoques exactos incluyen el enfoque de Kuhn-Tucker para resolver problemas de programación mixta entera [126], el pivoteo paramétrico de complementariedad [167] y la técnica de ramificación y acotamiento (branch and bound) [34]. Revisiones de métodos exactos para resolver problemas de programación binivel lineales se pueden encontrar en [290] y [178]. Sin embargo casi todos estos enfoques fallan al tratar con problemas de gran

tamaño.

Es importante mencionar que los software comerciales de optimización no contienen una función de propósito general para resolver problemas binivel. Sin embargo, existen dos solvers académicos disponibles para resolver de manera óptima problemas binivel mixtos enteros genéricos. Por ejemplo, los algoritmos para resolver programas lineales binivel mixtos enteros propuestos en [125] y [337].

Otro aspecto importante dentro de la programación binivel es que debería prestarse especial atención al problema de nivel inferior parametrizado en las variables fijas del nivel superior, ya que en muchos casos puede tener soluciones óptimas alternativas. Cuando esto ocurre, comúnmente se asumen los enfoques optimista o pesimista [116]. En el enfoque optimista, se considera la solución óptima del nivel inferior que es más conveniente para la función objetivo del nivel superior, mientras que en el enfoque pesimista, se selecciona la solución que es menos conveniente para la función objetivo del nivel superior en el nivel inferior.

En los últimos años se ha hecho mucho hincapié en muchas aplicaciones complejas de la vida real que pueden ser formuladas como modelos binivel pero que no pueden ser resueltas mediante los enfoques exactos existentes. Por lo tanto, los algoritmos aproximados son buenas alternativas. Ejemplos de algoritmos aproximados incluyen: (i) heurísticas [303]: "procedimientos para resolver problemas matemáticos bien definidos mediante un enfoque intuitivo en el cual la estructura del problema puede ser interpretada y explotada de manera inteligente para obtener una solución razonable"; (ii) metaheurísticas [255]: "estrategias de resolución de problemas de alto nivel para coordinar la cooperación entre otros métodos de búsqueda, incluyendo heurísticas y/o técnicas de búsqueda tradicionales, proporcionando una guía para diseñar estrategias generales independientes del problema, que luego los investigadores deben adaptar para resolver problemas difíciles específicos", (iii) matheurísticas [222]: "(meta)heurísticas que se basan en conceptos de la literatura tradicional de programación matemática. Estas heurísticas pueden tener subrutinas que involucran, por ejemplo, programación lineal, programación entera, programación dinámica, relajación lagrangiana o descomposición de Benders"; (iv) hiperheurísticas [54]: "enfoques de alto nivel que, dada una instancia particular del problema y un número de heurísticas de bajo nivel, pueden seleccionar y aplicar una heurística de bajo nivel adecuada en cada punto de decisión", y (v) simheurísticas [166]: "algoritmos que integran la simulación (en cualquiera de sus variantes) en un marco impulsado por

metaheurísticas para resolver problemas de optimización estocásticos complejos".

1.2 Contribuciones

El objetivo de la tesis es seguir contribuyendo a la literatura y el desarrollo de la programación binivel. Ante la falta de un review exhaustivo de metaheurísticas con programación binivel, buscamos que próximos investigadores tengan una fuente completa al buscar métodos aproximados de solución de problemas binivel. El desarrollo de algoritmos evolutivos anidados para contextos novedosos de la localización de hubs. Así como un nuevo algoritmo co-evolutivo para un problema de programación binivel de hubs que mejora las soluciones del problema de nivel superior, mostrando la efectividad de este algoritmo ante la falta de solvers comerciales capaz de resolverlo y por último, un nuevo problema de plantas competitivas binivel multi-objetivo donde el líder tiene un problema de optimización mono-objetivo y el seguidor tiene un problema bi-objetivo, la importancia de seleccionar un enfoque cuando se tienen múltiples soluciones en el nivel inferior así como el desarrollo de dos nuevos enfoques y las diferencias computacionales que conlleva resolver el problema de manera exacta contra las aproximaciones.

1.3 Estructura

La estructura de la tesis se encuentra a continuación: En el Capitulo 2, se mostrará una bibliometría completa de metaheuristicas para problemas de programación binivel haciendo énfasis en las características de las metaheuristicas y características del problema binivel, mostrando datos, gráficas y señalando practicas buenas y no tan buenas dentro de la literatura. En el Capitulo 3, se seguirá fomentando el uso de metaheuristicas con programación binivel a través de una aplicación en problemas de redes donde se desarrollará un modelo y un método de solución a través de un algoritmo evolutivo. Por otro lado en el capitulo 4 se mostrará el desarrollo de un nuevo algoritmo metaheurístico co-evolutivo enfocado al líder. En el capitulo 5 se mostrará un nuevo problema de localización de plantas competitivas utilizando programación binivel multi-objetivo y nuevos enfoques para resolverlo. Finalmente en el capitulo 6 se hablarán de las conclusiones y áreas de oportunidad.

Capítulo 2

Una bibliometría de programación binivel y metaheuristicas

Un modelo de programación binivel representa la relación en un proceso de toma de decisiones específico que involucra decisiones dentro de una estructura jerárquica de dos niveles. El problema de nivel superior está asociado con el tomador de decisiones de mayor jerarquía, y el problema anidado está asociado con un nivel inferior. Este enfoque de modelado se ha aplicado a diversas situaciones de la vida real. Sin embargo, los métodos de solución exacta son limitados debido a la complejidad inherente de los modelos de optimización binivel. Por lo tanto, se necesitan métodos de solución alternativos, como las metaheurísticas. El diseño e implementación de algoritmos metaheurísticos efectivos que produzcan soluciones de alta calidad en un tiempo computacional aceptable ha sido un área de investigación activa. Recientemente, ha habido un aumento en esta área de investigación. Este capítulo tiene como objetivo revisar todos los artículos publicados dedicados a la implementación de metaheurísticas para resolver problemas binivel hasta la fecha. Se realiza un análisis bibliométrico para rastrear la evolución de este tema. Además, se identifican las revistas y autores con más contribuciones. Los componentes específicos de las metaheurísticas propuestas se describen en detalle, independientemente de la inspiración detrás de las metaheurísticas. Se incluye un análisis detallado de la combinación de componentes para determinar los más comunes. Además, se detalla la clasificación de la forma en que se manejan los aspectos cruciales del problema binivel en las metaheurísticas, como el tipo de enfoque considerado para diseñar la metaheurística, lo que puede estar anidado, reformulación de un solo nivel, co-evolución y transformación bi-objetivo. También se señalan las metaheurísticas que asumen una única reacción del seguidor, los enfoques optimistas o pesimistas. Se incluye una discusión sobre los hallazgos interesantes, en la que se realiza una crítica de las prácticas laxas. Se enumeran algunas áreas de oportunidad para la investigación y enfoques recientes prometedores.

2.1 Antecedentes: Metaheurísticas

De acuerdo con Sörensen and Glover [325], "una *metaheurística es un marco algorítmico de alto nivel e independiente del problema que proporciona un conjunto de directrices o estrategias para desarrollar algoritmos de optimización heurística. El término también se refiere a una implementación específica del problema de un algoritmo de optimización heurística de acuerdo con las directrices expresadas en dicho marco*". Muchas metaheurísticas clásicas se inspiraron en metáforas biológicas, físicas u otras (algoritmos genéticos, recocido simulado, optimización de colonia de hormigas, etc.). Sin embargo, en esta revisión, adherimos a las ideas expresadas en [324], [73] y [27], que destacan que el uso excesivo de metáforas para introducir supuestas nuevas metaheurísticas es perjudicial para el desarrollo de este campo de la optimización. En cambio, los autores en [135], [208], [251], [329] y [106], resaltan que las metaheurísticas pueden diseñarse a partir de un conjunto relativamente pequeño de componentes algorítmicos. Por lo tanto, analizaremos las metaheurísticas para la optimización binivel desde la perspectiva de sus componentes. El diseño de las metaheurísticas busca equilibrar dos objetivos principales: la diversificación y la intensificación de la búsqueda. Mientras que la diversificación tiene como objetivo cubrir ampliamente el espacio de búsqueda, la intensificación concentra la búsqueda cerca de soluciones de alta calidad encontradas durante la búsqueda [255]. Las metaheurísticas ofrecen diferentes compromisos entre estos dos objetivos según los componentes específicos que tengan.

En esta revisión, adoptamos la visión unificadora de [135], que define las metaheurísticas como marcos que consisten en un número relativamente pequeño de componentes algorítmicos. Si bien nuestra investigación no tiene como objetivo proponer un nuevo marco, ampliamos su concepto de dos maneras (consulte la Figura 2.1). En primer lugar, incorporamos componentes adicionales que han surgido recientemente en la literatura o que creemos que faltaban en su propuesta. En segundo lugar, categorizamos estos componentes en dos grupos: operadores de solución, que son responsables de crear, modificar, mejorar o combinar soluciones, y estrategias de control de búsqueda, que guían el proceso de búsqueda.

Para ilustrar el concepto, se muestran tres metaheurísticas evolutivas bien conocidas en la Figura 2.1. Estas incluyen la estrategia evolutiva para la optimización binivel [235], un algoritmo genético [69] y una búsqueda dispersa [270]. La parte izquierda de la figura muestra el flujo de la metaheurística entre los operadores de solución, con el color indicando las conexiones. Además, pequeñas cajas en el lado derecho representan la presencia de estrategias específicas de control de búsqueda. Para perspectivas alternativas libres de metáforas sobre las metaheurísticas, sugerimos consultar [135], [208], [251], [329]

y [106]. Estas referencias ofrecen diferentes puntos de vista sobre estos procedimientos, centrándose en operadores de solución y estrategias de control de búsqueda comunes.

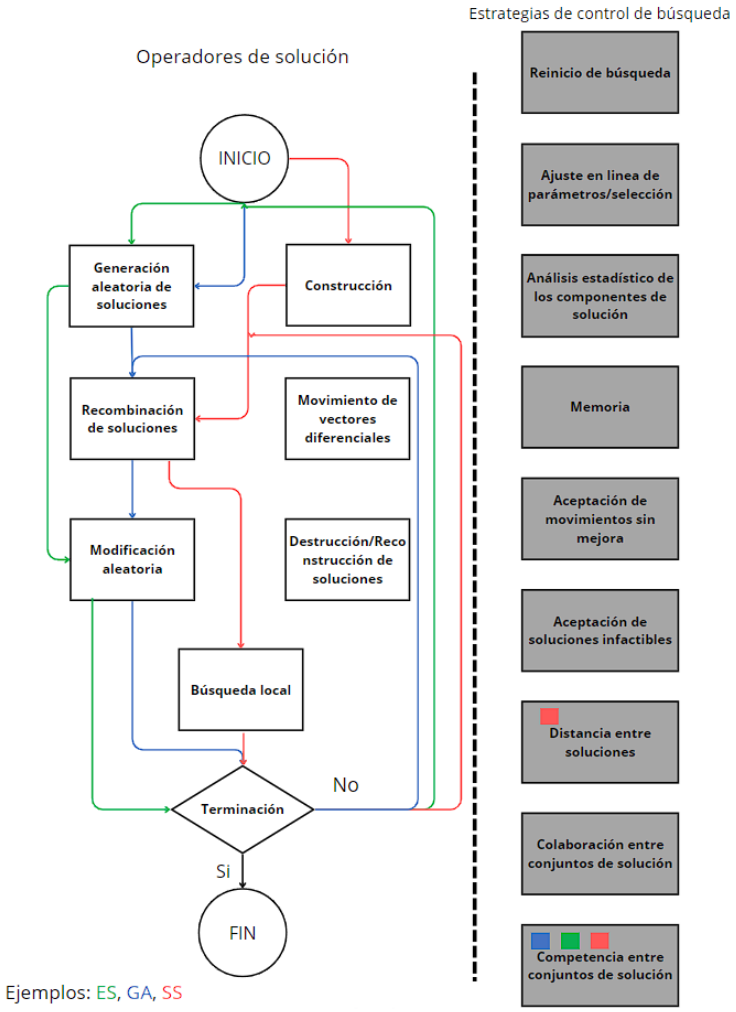


Figura 2.1: Metaheurísticas como conjunto de componentes algoritmicos. Operadores de solución (izquierda) y estrategias de control de búsqueda (derecha).

Por lo tanto, la Tabla 2.1 define los componentes algorítmicos que utilizaremos para clasificar las metaheurísticas en esta revisión de literatura, junto con una breve explicación y ejemplos de metaheurísticas clásicas que los utilizan. En la mayoría de los casos, existen descripciones claras y actualizadas de las metaheurísticas o componentes en [232] o [274], y tutoriales introductorios en [53]. También incluimos referencias que discuten cada uno de los componentes o métodos en las descripciones de la Tabla 2.1. Los acrónimos utilizados en algunas metaheurísticas se muestran en el Apéndice A.

La combinación de estos componentes ha dado lugar a lo que se conoce como metaheurísticas híbridas [277]. Aunque incluirlas en una de las analogías clásicas o marcos conceptuales es difícil, en muchos casos han permitido el desarrollo de los mejores méto-

Tabla 2.1: Componentes de las metaheurísticas analizados en la revisión

Componente	**Descripción**	**Ejemplos de metaheurísticas**
	Operadores de solución	
Generación aleatoria de soluciones	Se utilizan números aleatorios para generar soluciones (iniciales) sin información específica del problema [298]	GA (población inicial), ES (soluciones iniciales), SA (solución inicial)
Modificación aleatoria	Modificar la(s) solución(es) actual(es) mediante perturbaciones aleatorias [135]	ILS (perturbación), VNS (shake), GA (mutación), ES (mutación)
Recombinación de soluciones	Seleccionar y combinar soluciones utilizando un operador de cruce o método de combinación para generar nuevas soluciones [251]	GA (cruce), SS (votación o método de combinación)
Búsqueda local	Seguir una secuencia de mejoras locales para alcanzar una solución óptima local [208]	TS (vecindario), VNS (múltiples vecindarios), ILS (búsqueda local)
Movimiento de vectores diferenciales	Se utiliza un vector/movimiento diferencial para cambiar la(s) solución(es) actual(es) hacia nuevas soluciones candidatas [251]	PSO, DE
Construcción	Utilizar los datos del problema para construir una solución, paso a paso, con alguna aleatorización en muchos casos.[280]	GRASP (construcción), ACO (construcción de hormigas)
Destrucción y reconstrucción de soluciones	Destrucción parcial iterativa y posterior reconstrucción de soluciones utilizando una sola [330] o varias heurísticas [224]	ALNS, IGA
	Estrategias de control de búsqueda	
Competencia entre conjuntos de soluciones (a.k.a Evolución)	La actualización/reemplazo de las soluciones en un conjunto está guiada por diferentes criterios de búsqueda, como la calidad y/o diversidad de las soluciones [36]	GA, ES, SS, DE, MA
Colaboración entre un conjunto de soluciones/agentes (a.k.a. Estigmergia)	Comunicación indirecta y coordinación entre soluciones o agentes utilizados para crear nuevas soluciones o intercambiar información recopilada durante el proceso de búsqueda [251]	ACO, PSO, COEAs
Aceptación de movimientos no mejora	Permitir movimientos a soluciones peores mediante un mecanismo probabilístico o determinista [208]	SA (regla de aceptación), TS (criterio de aspiración)
Aceptación de soluciones infactibles	Aceptar soluciones infactibles durante la búsqueda, tal vez penalizando la violación de las restricciones o utilizando otros mecanismos para recuperar la factibilidad [139] [108]	TS (oscilación estratégica), GA penalización de soluciones infactibles
Distancia entre soluciones	Se utiliza una medida de distancia entre soluciones para controlar la diversidad de las soluciones durante la búsqueda [281] [326]	SS (método de actualización del conjunto de referencia), MAPM (gestión de población)
Memoria	Utilizar la memoria de la experiencia de búsqueda pasada para guiar la búsqueda futura [208]	TS
Reinicio de búsqueda	Reiniciar el proceso de búsqueda en una región diferente una vez que ha convergido en un óptimo local [208]	Métodos de multi-arranque [230]
Ajuste/selección de parámetros en línea	Aprendizaje en línea para ajustar parámetros simultáneamente durante la búsqueda de la metaheurística [135]	TS reactivo, GRASP reactivo
Análisis estadístico de componentes de solución	Se muestrean nuevas soluciones a partir de una distribución de probabilidad, que se estima a partir de soluciones de las iteraciones anteriores [107] [187]	CE

dos para resolver diferentes problemas de optimización. Del mismo modo, la hibridación de las metaheurísticas con métodos exactos es común en la actualidad, dando lugar a algoritmos matheurísticos [222]. Sin embargo, los problemas de tipo binivel (en general) implican encontrar la solución óptima del problema del nivel inferior, lo que hace que el término "matheurístico"no sea apropiado en este contexto, ya que la optimalidad de las decisiones del seguidor se alcanza comúnmente mediante métodos exactos de programación matemática. Por lo tanto, no vamos a utilizar el término "matheurístico". No obstante, en nuestro análisis indicaremos los enfoques (óptimos o aproximados) utilizados para resolver el nivel inferior al abordar problemas binivel con metaheurísticas.

A esta lista de componentes, añadimos una distinción bastante común en las metaheurísticas, basada en si, en cada iteración, la metaheurística opera sobre una sola solución o sobre un conjunto de soluciones [135] [329]. También hemos examinado el uso de la computación paralela para reducir el tiempo de cálculo de las metaheurísticas al resolver problemas complejos [101].

En esta revisión bibliometrica, nos enfocamos en las metaheurísticas dedicadas a resolver problemas de programación binivel (BPPs, por sus siglas en inglés). Existe una clara necesidad de una revisión exhaustiva sobre este tema debido a la creciente actividad en esta área en los últimos años. Hay dos publicaciones relacionadas que vale la pena señalar. El libro editado por Talbi [339], en el cual se presenta un compendio de algunas metaheurísticas dedicadas a resolver problemas binivel específicos; y la revisión de Sinha et al. [317], en la cual se analizan los algoritmos evolutivos para resolver problemas binivel. En esa investigación, los autores se enfocaron únicamente en los algoritmos evolutivos, pero no incluyeron todos los documentos existentes en ese tema, por lo que no fue una revisión exhaustiva. Además, dado que las metaheurísticas no se limitan únicamente a los algoritmos evolutivos, en nuestra revisión incluimos todo tipo de metaheurísticas. Realizamos tres clasificaciones diferentes: (i) según el tipo de problema de programación matemática considerado en cada nivel de decisión, (ii) según los componentes incluidos en las metaheurísticas, y (iii) según el enfoque binivel específico asumido.

2.2 Enfoques al utilizar metaheurísticas para resolver problemas de programación binivel

Aplicar metaheurísticas a los problemas de programación binivel no es una tarea sencilla debido a la estructura inherente del problema. Es decir, las variables de decisión se dividen en dos conjuntos disjuntos. Además, se necesita la reacción óptima del seguidor ante una decisión dada del líder. Como consecuencia, la reacción del seguidor afecta

al espacio de decisión del líder y/o a su función objetivo. Como es de suponer, existen diferentes esquemas para manejar las variables de decisión involucradas en un problema binivel.

Los enfoques más comunes utilizados al diseñar metaheurísticas para resolver problemas binivel son el enfoque anidado y el esquema co-evolutivo. Enfoques menos comunes que, no obstante, se han utilizado son la consideración de una reformulación de un solo nivel y la transformación en un problema biobjetivo.

El enfoque anidado es el más popular. En este enfoque, la metaheurística primero se ocupa de las soluciones del líder, y para cada una de ellas se resuelve el problema del seguidor. Por lo tanto, la calidad general de la solución binivel se evalúa considerando tanto las variables de decisión del nivel superior como las del nivel inferior. Como su nombre indica, existe un enfoque anidado en el que primero el líder construye soluciones y luego el seguidor reacciona a la decisión del líder. Sin embargo, este enfoque puede ser costoso (en términos computacionales), ya que el problema del seguidor se resuelve para cada decisión del líder. Por ejemplo, cuando se realiza una búsqueda local, cada movimiento genera una nueva solución del líder y el problema del seguidor debe resolverse para cada movimiento.

El segundo enfoque más utilizado es reformular el problema binivel en un problema de un solo nivel. Como se mencionó antes, se pueden derivar reformulaciones de un solo nivel equivalentes para BPPs particulares. Por lo general, estas reformulaciones son no lineales debido a algunas de las restricciones. Las metaheurísticas pueden desarrollarse para evitar lidiar con esas restricciones complicadas y las formulaciones resultantes. Sin embargo, estas no resuelven adecuadamente un problema binivel, sino una reformulación de un solo nivel equivalente. En otras palabras, las metaheurísticas clásicas se pueden aplicar de manera directa para resolver las reformulaciones de un solo nivel. Es importante destacar que las reformulaciones de un solo nivel pueden ocurrir cuando existe convexidad y condiciones de regularidad suficientes en el problema del seguidor.

Uno de los enfoques menos comunes es el enfoque co-evolutivo, en el que se consideran dos poblaciones de soluciones parciales, una para el líder y otra para el seguidor. Existe un mecanismo co-evolutivo que intercambia información entre las poblaciones, por ejemplo, enviando soluciones de élite del líder a la población del seguidor. En este caso, se puede obtener la reacción del seguidor y evaluar la solución binivel completa. En contraste, cuando se envían soluciones del seguidor a la población del líder, se deben asumir criterios más especializados. A menudo se considera la cooperación simétrica y asimétrica

entre las poblaciones. Dado que la jerarquía predefinida líder-seguidor en un BPP debe mantenerse, los enfoques co-evolutivos para resolver BPPs son escasos. En particular, el problema principal es la manera en que se manejará la información enviada por el seguidor al líder.

Finalmente, otro enfoque menos común consiste en transformar el BPP en un problema multiobjetivo. La principal desventaja de este enfoque es que la solución óptima de un problema binivel no está incluida (en general) en el frente de Pareto de la versión biobjetivo del problema [95], [301], [361]. Ha habido diferentes estudios intentando establecer las condiciones de equivalencia para ambos problemas, pero no han tenido éxito [347], [148].

2.2.1 Obteniendo la reacción del seguidor

Considerando el siguiente problema binivel general lineal-lineal y utilizando las definiciones establecidas en [33],

$$\begin{aligned} \min_{x \in X} \quad & F(x,y) = c_1 x + d_1 y && (2.1)\\ s.t. \quad & A_1 x + B_1 y \leq b_1 && (2.2)\\ & \min_{y \in Y} f(x,y) = c_2 x + d_2 y && (2.3)\\ & s.t. \quad A_2 x + B_2 y \leq b_2 && (2.4) \end{aligned}$$

donde $x \in X \subset \mathbb{R}^n$, $y \in Y \subset \mathbb{R}^m, F : X \times Y \to \mathbb{R}$ y $f : X \times Y \to \mathbb{R}$ además $c_1, c_2 \in \mathbb{R}^n, d_1, d_2 \in \mathbb{R}^m, b_1 \in \mathbb{R}^p, b_2 \in \mathbb{R}^q, A_1 \in \mathbb{R}^{p \times n}, B_1 \in \mathbb{R}^{p \times m}, A_2 \in \mathbb{R}^{q \times n}, B_2 \in \mathbb{R}^{q \times m}$.

se describen algunos conceptos relevantes para obtener soluciones factibles de BPPs. Primero, definamos la región de restricciones del BPP considerado:

$$S = (x,y) : x \in X, y \in Y, A_1 x + B_1 y \leq b_1, A_2 x + B_2 y \leq b_2 \quad (2.5)$$

Recordemos que una vez que el líder ha tomado una decisión, el seguidor debe responder con el objetivo de optimizar su propia función objetivo, pero teniendo en cuenta el impacto de la decisión del líder en sus restricciones. En otras palabras, el seguidor debe resolver un problema parametrizado en la decisión fija del líder. Formalizando esta idea, el conjunto de soluciones factibles del seguidor para un $x \in X$ dado por el líder se denota

de la siguiente manera:

$$S(x) = y \in Y : B_2 y \leq b_2 - A_2 x \tag{2.6}$$

En la Ecuación (2.6), se considera la decisión del líder en el problema del seguidor. Además, el seguidor debe reaccionar de manera *racional*, es decir, de acuerdo con su propia función objetivo y posibilidades. Para un x fijo en X, el conjunto de reacciones *racionales* del seguidor se define de la siguiente manera:

$$P(x) = y \in Y : y \in \text{argmin}[f(x, \hat{y}) : \hat{y} \in S(x)] \tag{2.7}$$

Finalmente, la región factible para un BPP se llama región inducible (IR), que se define de la siguiente manera:

$$IR = (x, y) : (x, y) \in S, y \in P(x) \tag{2.8}$$

Los conjuntos definidos en las Ecuaciones (2.7) y (2.8) indican que el problema del seguidor parametrizado en la decisión del líder debe resolverse de manera óptima para tener una solución factible de BPP. Sin embargo, el término racional.[en] la definición implica que el seguidor actúa a favor de sus propios objetivos y no hay un comportamiento inesperado. Además, existe un problema importante cuando resolver el problema del seguidor resulta en una tarea muy compleja, por ejemplo, cuando un problema NP-difícil está involucrado en el problema del seguidor. En ese caso, el seguidor debe actuar de manera *racional*, lo que implica resolver su propio problema de manera conveniente.

Considerando un par $(x, y) \in S$, pero sin asegurar que $y \in P(X)$ conduce a soluciones de BPP infactibles, en un sentido estricto. Sin embargo, para resolver problemas complejos de la vida real, se han considerado soluciones semi-factibles [14], alcanzables [79], y pseudo-factibles [239]. Estos tipos de soluciones relajan las condiciones del conjunto de reacciones *racionales* al aceptar soluciones de buena calidad o casi óptimas al resolver el problema del seguidor. Es importante mencionar que las soluciones semi-factibles de BPP conducen a un límite válido, pero clasificar esto como un límite superior o inferior depende de la estructura específica del BPP en estudio. Se han dedicado esfuerzos a analizar el impacto de tener sub-optimalidad al resolver el problema del seguidor [23], [79], [239].

En el contexto de las metaheurísticas diseñadas para resolver BPPs, es común obtener soluciones de buena calidad para el problema del seguidor en lugar de resolverlo hasta la optimalidad. En estos casos, se utilizan dos enfoques: (i) resolver el problema del segui-

dor utilizando un algoritmo de aproximación (heurístico, metaheurístico, matheurístico, etc.), y (ii) estimar la reacción del seguidor mediante una función de aproximación (modelos estadísticos, proyecciones, etc.). El segundo enfoque se conoce como modelo de sustitución-aproximación y tiene como objetivo reducir el número de veces que se resuelve el problema del seguidor. Dado que se está estimando la reacción del seguidor, este enfoque claramente impacta significativamente en el esfuerzo computacional requerido [316], [238].

También vale la pena mencionar que se deben hacer algunas suposiciones para garantizar que el BPP definido en las Ecuaciones (2.1)-(2.4) esté bien planteado. La primera asegura que el seguidor tenga opciones para reaccionar a una decisión fija del líder, es decir, $P(x) \neq \emptyset$. La segunda se refiere al caso en que existen múltiples soluciones óptimas para el problema del seguidor parametrizado. En este caso, diferentes reacciones óptimas del seguidor pueden afectar la función objetivo del líder de manera diferente. Al final, el seguidor debe seleccionar solo una de sus múltiples reacciones óptimas. Para lograr esto, se asume el esquema optimista o pesimista. El esquema optimista asume un comportamiento cooperativo, es decir, el seguidor tomará su decisión favoreciendo la función objetivo del líder [116]. Por otro lado, el esquema pesimista es no cooperativo y se asume en un contexto de aversión al riesgo cuando la decisión del seguidor no favorece la función objetivo del líder [346]. Recordemos que, para una decisión fija del líder, ambos esquemas producen el mismo valor de función objetivo para el seguidor. Por lo general, el esquema optimista se selecciona preferentemente sobre el pesimista. Además, en general, el enfoque pesimista es más complejo de resolver. Además, la mayoría de los algoritmos exactos o aproximados para resolver un BPP optimista no se aplican a su versión pesimista [367]. Sin embargo, las metaheurísticas para resolver BPPs pueden adaptarse fácilmente para tratar cualquiera de los dos esquemas, optimista o pesimista. Si se incluye un procedimiento para obtener todas las reacciones óptimas del seguidor, entonces se evalúan todas las soluciones óptimas y se elige la adecuada.

2.3 Metodología de la revisión y análisis descriptivo

Para llevar a cabo una revisión de literatura correcta y válida, adaptamos los principales pasos seguidos en [31] y [282]. En detalle, los pasos seguidos son los siguientes: definición de las preguntas de investigación, recolección de material, análisis descriptivo, selección de categorías y evaluación del material.

2.3.1 Definición de las preguntas de investigación

El primer paso en esta revisión de literatura es definir las preguntas de investigación, que guían la investigación y aclaran las principales ideas que estamos persiguiendo al clasificar el estado del arte en lo que respecta a las metaheurísticas aplicadas a la resolución de problemas de programación binivel (BPPs). Se definen diez preguntas de investigación:

Q1 : ¿Cuándo fueron propuestas estas metaheurísticas?

Q2 : ¿Dónde se han publicado estas metaheurísticas?

Q3 : ¿Quiénes han desarrollado estas metaheurísticas?

Q4 : ¿Dónde se encuentran ubicados los investigadores que han estado proponiendo el uso de estas metaheurísticas?

Q5 : ¿Cuáles son las metaheurísticas más utilizadas para resolver BPPs?

Q6 : ¿Cuáles son los principales componentes que suelen aparecer en las metaheurísticas dedicadas a resolver BPPs?

Q7 : ¿Es el enfoque anidado el más común al diseñar estas metaheurísticas?

Q8 : ¿Se han aplicado metaheurísticas para resolver reformulaciones de un solo nivel de los BPPs?

Q9 : ¿Cómo se resuelve comúnmente el problema de nivel inferior al diseñar metaheurísticas que tratan con soluciones de nivel superior?

Q10 : ¿Se asumen los enfoques optimistas o pesimistas o simplemente se ignoran en la mayoría de las metaheurísticas diseñadas para resolver BPPs?

2.3.2 Recolección del material

Con el objetivo de obtener artículos relacionados, se realizó una búsqueda en las principales bases de datos académicas: Scopus, Web of Science y Google Scholar, utilizando tres conjuntos jerarquizados de palabras clave. Las palabras clave consideradas en el conjunto de nivel más alto son *"optimización binivel", "programación binivel", "problemas líder-seguidor", "problemas de Stackelberg", "optimización jerarquizada"*. Luego, las palabras clave consideradas en el conjunto de nivel intermedio son *""metaheurísticas", "algoritmos evolutivos", "inteligencia de enjambre", "basado en población", "búsqueda local", "basado en trayectoria", "método de solución única"*. Finalmente, las palabras clave

incluidas en el conjunto de nivel inferior son *"cadena de suministro", "diseño de redes", "precios", "logística verde", "logística humanitaria", "logística marítima", "ubicación de instalaciones", "preferencias de los clientes", "ubicación de instalaciones competitivas", "problemas de ataque-defensa", "problemas de interdicción"*. Luego, se buscaron palabras clave en los conjuntos de nivel más alto, intermedio e inferior. Además, también se buscaron combinaciones de palabras clave de los tres conjuntos.

Con el fin de tener un conjunto más completo de artículos, también realizamos una búsqueda en cascada hacia atrás (referencias citadas) y hacia adelante (artículos citantes) basada en los resultados iniciales obtenidos en estas bases de datos. Como se resalta en [368], las estrategias híbridas que combinan la búsqueda en bases de datos y la cascada hacia atrás y hacia adelante mejoran la cantidad y la cobertura temporal de los artículos incluidos en las revisiones sistemáticas de literatura.

Los artículos recopilados se limitan a aquellos escritos en inglés y publicados en revistas revisadas por pares, actas de conferencias con referato y capítulos de libros hasta el 31 de diciembre de 2022. Después de revisar los resultados y descartar los duplicados, se obtuvo una colección de 263 artículos. Para garantizar la relevancia en esta revisión actual, se consideran los siguientes criterios de inclusión y exclusión: se debe estudiar un problema de optimización binivel; se excluyen los problemas de dos niveles, dos etapas, dos fases y secuenciales; los artículos deben incluir una metaheurística para resolver un problema binivel. En los casos en que un artículo proponga una metaheurística denominada "binivel", pero no incluya un problema binivel, se excluyó de esta revisión.

Por otro lado, para garantizar que se consideren todas las metaheurísticas utilizadas para resolver problemas de tipo binivel en esta investigación, también se incluyeron los artículos que proponen un método exacto, heurístico, matheurístico u otro para resolver un problema binivel, pero utilizan una metaheurística con fines de comparación en la experimentación computacional.

Cabe mencionar que existe un artículo muy influyente de Sinha et al. [307] con más de 90 citas, pero está disponible en el sitio de preimpresión arXiv. Este artículo no ha sido revisado por pares, por lo que, para ser coherentes con nuestros criterios, no se considera en esta revisión. De manera similar, la referencia [247] se eliminó de la base de datos debido a que no fue posible acceder al texto completo del artículo.

2.3.3 Analisis Descriptivo

Las primeras cuatro preguntas proporcionan un marco general en el que se han propuesto metaheurísticas para resolver problemas binivel. Además, ayudan a estructurar

nuestros análisis al centrarnos en enfoques específicos o componentes involucrados en cada metaheurística.

Para responder a la pregunta Q1, los artículos recopilados se han clasificado por año de publicación. La Figura 2.2 muestra el número de artículos que se han publicado cada año sobre este tema. La primera metaheurística dedicada a resolver un problema binivel se publicó en 1994. El número de artículos fue relativamente bajo hasta 2006, cuando el número aumentó a más de cinco artículos por año. Es notable que hay un aumento significativo en los últimos dos años, con más de 20 y 30 artículos por año, respectivamente.

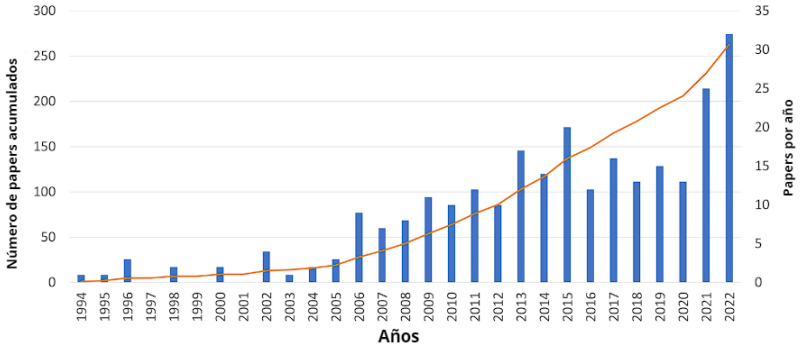


Figura 2.2: Evolución en el tiempo de articulos de metaheuristicas para optimización binivel

La pregunta Q2 se aborda para identificar las revistas donde se han publicado las metaheurísticas propuestas para resolver problemas binivel. La Figura 2.3 enumera las 10 principales revistas que han publicado los artículos revisados. Las metaheurísticas revisadas se publicaron en 194 artículos de revistas, 58 en actas de conferencias y 11 en capítulos de libros. A partir de la Figura 2.3, se puede observar que los artículos revisados están dispersos en muchas revistas diferentes. “Computers & Operations Research”tiene la mayoría de los artículos publicados, seguida de cerca por “European Journal of Operational Researchz “Applied Soft Computing”. Además, vale la pena mencionar que las actas de la Conferencia sobre Computación Evolutiva (CEC) contienen 19 artículos cortos sobre este tema, lo que representa alrededor del 33 % de los artículos en las actas.

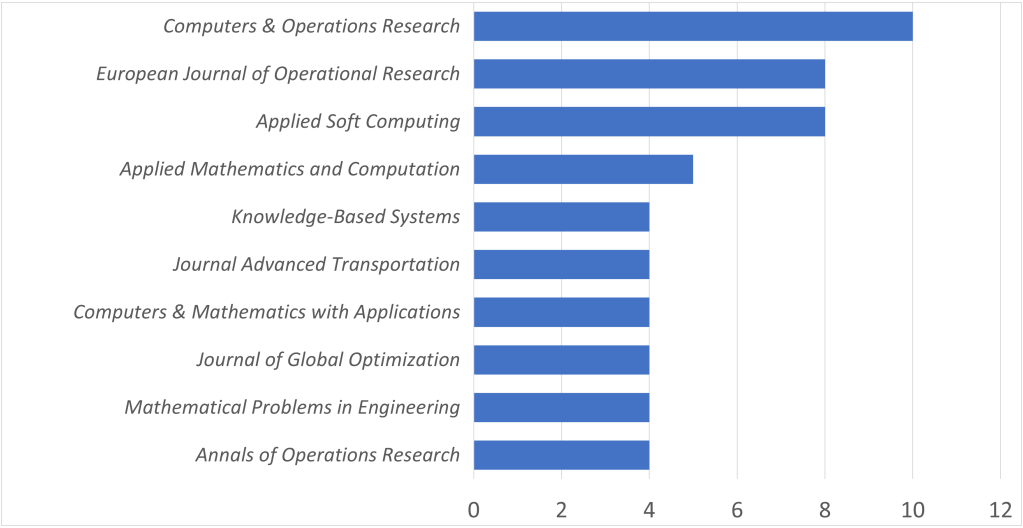


Figura 2.3: Top 10 de revistas con artículos publicados sobre el uso de metaheurísticas para resolver problemas binivel

Para categorizar los diferentes grupos de investigación que han propuesto metaheurísticas para resolver problemas binivel, se registran el nombre y el país de afiliación del autor correspondiente. Es decir, se registran el nombre y el país de su afiliación. En los casos en que no se indique el autor correspondiente, se utiliza el primer autor para llevar a cabo este análisis. Además, si el autor registrado tiene más de una afiliación, se utiliza la primera que se indique. Al hacer esto, se responden las preguntas de investigación Q3 y Q4. La Figura 2.4 muestra los nombres de los 10 principales autores en el campo. Además, también se indica el acrónimo de cada revista o actas en las que han publicado los artículos revisados, en orden cronológico de abajo hacia arriba. La lista de los nombres completos se muestra en B.

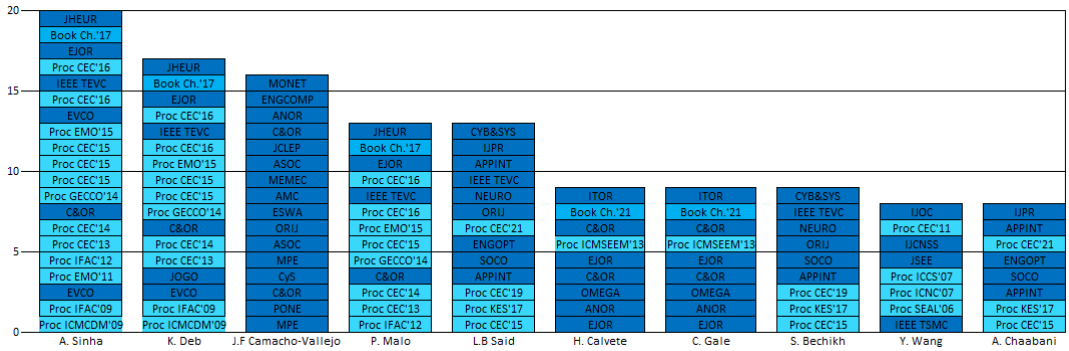


Figura 2.4: Top 10 de contribuyentes en el área bajo revisión

De la Figura 2.4 se puede observar que el autor más prolífico en este tema es el indio A. Sinha, pero la mayoría de sus contribuciones se publican en actas de conferencias. El

autor con más artículos publicados en revistas es el mexicano J.F. Camacho-Vallejo. A partir de este análisis, se identifican tres grupos de investigación principales en esta área: el grupo de A. Sinha, K. Deb y P. Malo, el grupo español de H. Calvete y C. Galé y, más recientemente, el grupo tunecino de L.B. Said, A. Chaabani y S. Bechikh han demostrado una fuerte colaboración entre ellos en este tema. Estos hallazgos responden a la pregunta de investigación Q3.

La Figura 2.5 ilustra los países donde se basan los investigadores en esta área de investigación. El diagrama se muestra en forma de mapa de calor, donde un color más oscuro denota que se han publicado más artículos desde ese país. También se indica el número de artículos revisados para los países con 3 o más artículos.

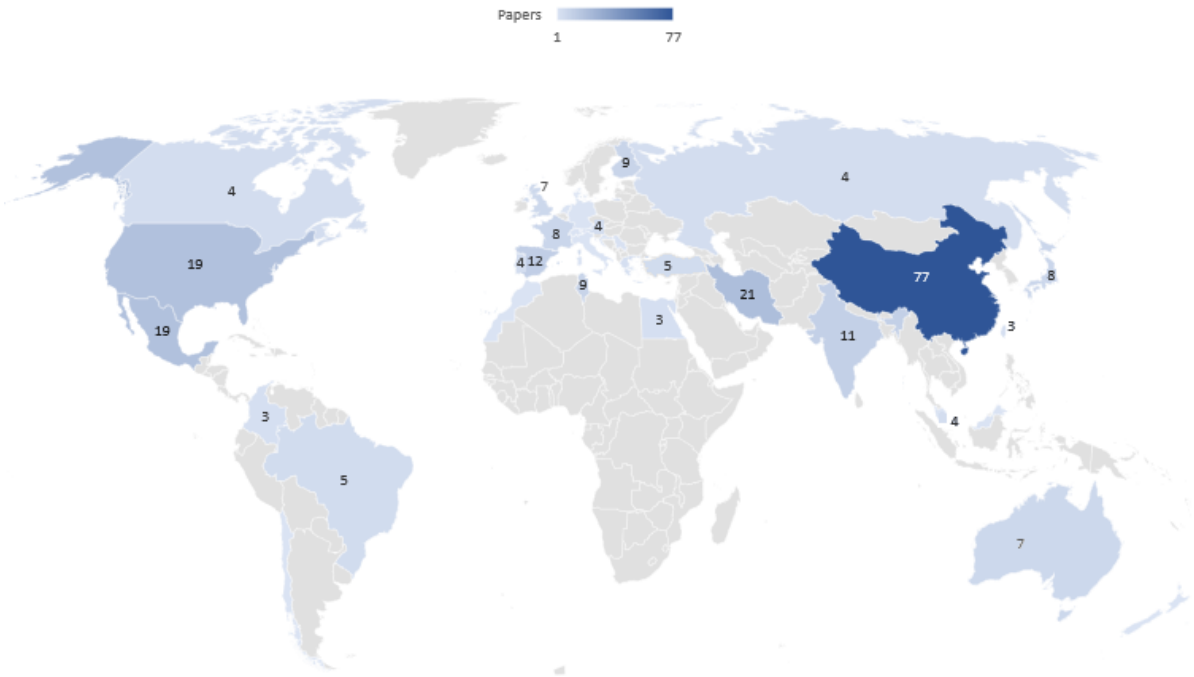


Figura 2.5: Países desde donde se han propuesto metaheurísticas para resolver problemas binivel

La Figura 2.5 muestra que los 10 países cuyos autores más contribuyen en este campo de investigación son, en orden descendente: China (77), Irán (21), México (19), Estados Unidos (19), España (12), India (11), Finlandia (9), Túnez (9), Francia (8) y Japón (8). En un análisis por continentes, cuatro países asiáticos dominan el top 10, seguidos por tres países europeos y dos de América del Norte. Esto responde a la pregunta de investigación Q4.

2.3.4 Selección de Categoría

En primer lugar, para identificar los nombres reportados y los componentes específicos de las metaheurísticas, se realiza un análisis profundo de cada una de ellas. Como se mencionó en la Sección 2.1, las metaheurísticas no solo se clasifican por sus nombres, sino también por sus componentes, por lo que examinamos tanto los nombres como los componentes. Específicamente, para cada componente de la Tabla 2.1, analizamos si una metaheurística dada incluye cada componente en su diseño. Asimismo, registramos si la metaheurística opera sobre una sola solución o un conjunto de soluciones o agentes. También recopilamos información sobre el uso de una implementación paralela de la metaheurística. Utilizando estos datos, construimos una tabla que respalda la respuesta a las preguntas Q5 y Q6. La tabla completa se incluye en el material suplementario en línea.

En segundo lugar, se analiza la manera en que las metaheurísticas publicadas han abordado los aspectos específicos binivel del problema. Por ejemplo, si la metaheurística explora soluciones del nivel superior o no. Si lo hace, se registra la forma en que se obtiene la reacción del seguidor. Si no lo hace, puede emplearse un modelo de un solo nivel para reformular el BPP, y una metaheurística puede diseñarse para resolverlo. Además, en lo que respecta al seguidor, es importante identificar si se resuelve o no su problema, parametrizado en una decisión fija del líder. Por ejemplo, se puede realizar una aproximación, considerar estigmergia entre conjuntos de soluciones u otras aproximaciones. Finalmente, se revisan las suposiciones sobre la solución única del problema del seguidor. Si esto último no está garantizado, pueden asumirse los enfoques optimista o pesimista. También se revisa esta característica crucial de los problemas binivel. Este análisis responde a las preguntas de investigación Q7-Q10.

2.3.5 Evaluación del Material

Ahora, procedemos a la evaluación de las características de los artículos revisados. Primero, se lleva a cabo un análisis en relación con los nombres y componentes de las metaheurísticas para resolver BPPs. Luego, se presenta una revisión de los aspectos fundamentales de los problemas binivel que deben incluirse en un método de solución propuesto.

Metaheurísticas para resolver BPP's

Recordemos que el BPP lineal-lineal se clasifica como NP-duro. Aparte de los BPPs lineal-lineal, los problemas asociados al líder y al seguidor pueden ser de diferentes tipos. Se han considerado diferentes combinaciones de problemas enteros, binarios, mixtos-enteros, no lineales, multiobjetivo, estocásticos, entre otros, en cada nivel de decisión. Para clasificar los BPPs resueltos por las metaheurísticas, primero diferenciamos entre problemas deterministas y estocásticos. Si el problema es determinista, verificamos el número de funciones objetivo que se están considerando. Específicamente, identificamos si es un problema monoobjetivo o multiobjetivo. Si se considera una función monoobjetivo, diferenciamos aún más los problemas como lineales o no lineales. En el caso de un problema lineal, procedemos a identificar si es continuo (lineal), entero, binario o mixto-entero. Por otro lado, si el problema involucra estocasticidad, múltiples funciones objetivo o componentes no lineales, entonces se clasifica como Estocástico, Multiobjetivo o No lineal, respectivamente. La jerarquía de esta clasificación se muestra en la Figura 2.6.

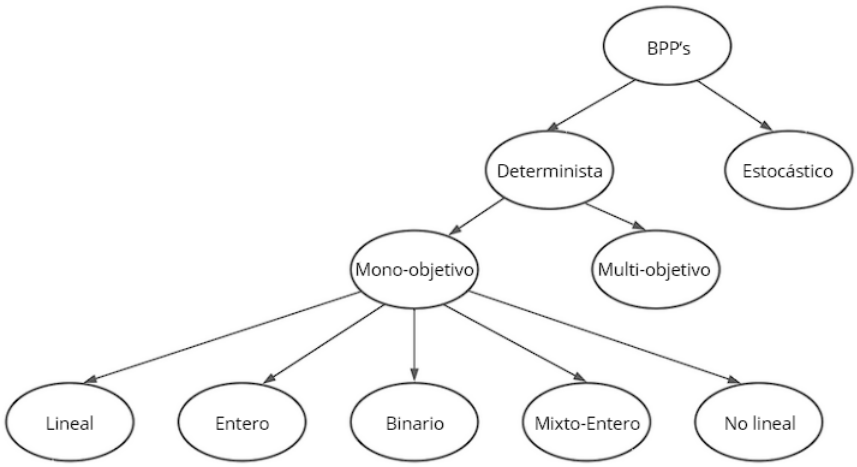


Figura 2.6: Jerarquía utilizada para clasificar los BPP resueltos por las metaheurísticas

Nivel Superior / Nivel Inferior	Lineal	Entero	Binario	Mixto-entero	No lineal	Multiobjetivo	Estocástico
Lineal	38	2	1	1	1	3	-
Entero	-	3	-	1	2	1	-
Binario	15	4	35	2	2	2	1
Mixto-entero	4	-	4	12	3	2	1
No lineal	4	-	2	3	122	1	4
Multiobjetivo	5	2	1	2	8	12	2
Estocástico	-	-	2	-	4	-	6

Tabla 2.2: Diferentes combinaciones de problemas estudiados en los trabajos revisados.

En la Tabla 2.2, se muestra que el BPP más común para el cual se han diseñado metaheurísticas considera un problema no lineal para el líder y otro problema no lineal para el seguidor. Este tipo de BPPs se estudia en 122 de los 263 trabajos revisados, lo cual es esperado debido a la jerarquía en la clasificación realizada. Específicamente, funciones objetivo no lineales, restricciones no lineales, problemas de equilibrio, problemas de desigualdad variacional, entre otros, se agrupan en esta categoría. Otros tipos comunes son el lineal-lineal y el binario-binario, estudiados en 38 y 35 trabajos, respectivamente. Nótese que diferentes tipos de BPPs pueden ser resueltos por la misma metaheurística en un trabajo, lo cual es la razón por la que la suma de números en la Tabla 2.2 es mayor que 263. La complejidad de los BPPs es evidente, y el uso de metaheurísticas está justificado debido a su naturaleza compleja.

Existen otros tipos específicos de BPPs diferentes a los descritos anteriormente. Por ejemplo, BPPs con variables difusas [205] y [118], fraccionales lineales [56], fraccionales no lineales [192], fraccionales multiobjetivo [284], cuadráticos fraccionales [154], por tramos [91]. Por brevedad, los enfoques cuadráticos, no convexos, bilineales, minimax y otros similares se incluyen en la categoría de problemas no lineales.

Se proponen 307 metaheurísticas en los 263 trabajos revisados. Hay 259 metaheurísticas puras y 48 métodos híbridos. Se representan en la Figura 2.7, en la cual el tamaño de cada círculo es proporcional al número de veces que aparece. El color indica metaheurísticas que comparten componentes comunes según la literatura. Un enlace entre ambos círculos representa un algoritmo que hibridiza estas dos metaheurísticas. A incluye una tabla con los nombres completos de las metaheurísticas y sus acrónimos mostrados. Como se ve en esta figura, los algoritmos genéticos (GA) son las metaheurísticas más utilizadas para resolver BPPs, seguidos por la optimización por enjambre de partículas (PSO) y los algoritmos evolutivos (EA). Los EAs engloban las estrategias evolutivas, algoritmos genéticos y algunas otras metaheurísticas basadas en poblaciones como los algoritmos meméticos.

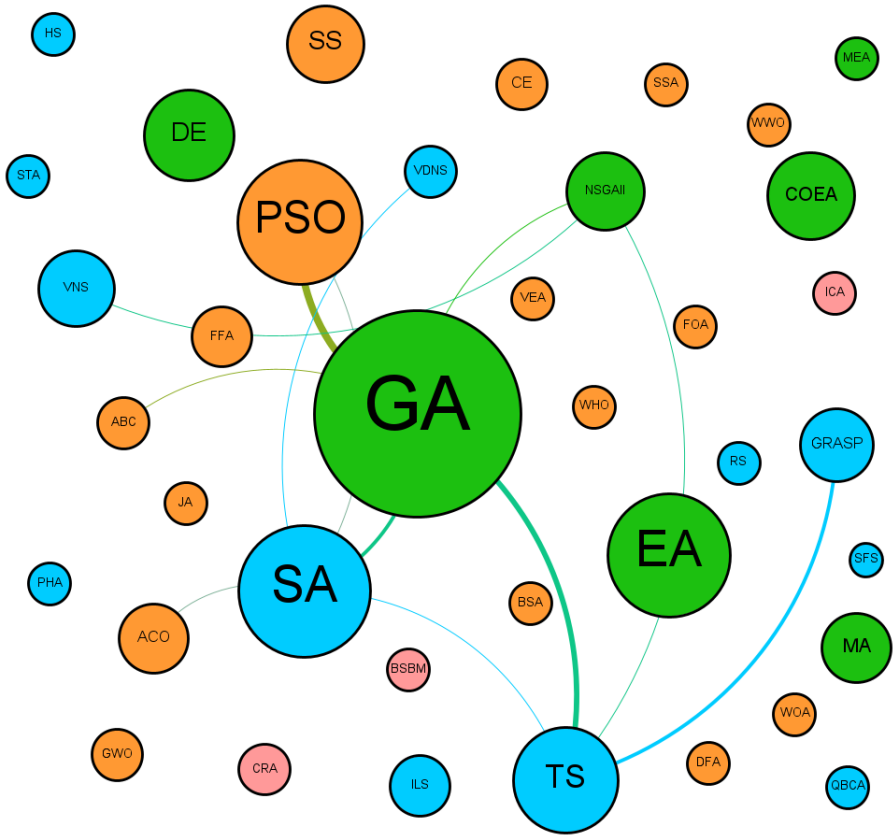


Figura 2.7: Metaheurísticas utilizadas para resolver BPPs y sus hibridos

Para comprender los diseños algorítmicos predominantes, vamos un paso más allá y consideramos las metaheurísticas como una lista de componentes que se combinan, independientemente de su nombre. Aquellos diseños que aparecen en al menos cinco artículos se incluyen en la Tabla 2.3. Esta tabla tiene tres columnas: la primera es la lista de componentes que se combinan en un diseño dado, la segunda es el nombre utilizado para este diseño algorítmico en la literatura, y la tercera es una lista completa de los artículos que lo utilizan, incluido un recuento del número de veces que se ha utilizado en los artículos revisados. Tenga en cuenta que el recuento de frecuencia y el número de artículos pueden no ser los mismos, ya que algunos artículos informan más de una metaheurística. Los algoritmos genéticos simples que recurren a la generación aleatoria de soluciones, la recombinación de soluciones, la modificación aleatoria (es decir, mutación) y la competencia entre conjuntos de soluciones (evolución) aparecieron como el diseño algorítmico preferido en 79 metaheurísticas. El segundo diseño más preferido, con 25 metaheurísticas, es la optimización de enjambre de partículas (PSO), que tiene la generación aleatoria de soluciones, el movimiento de vectores diferenciales y la colaboración entre un conjunto

de soluciones como sus tres componentes clave. Si se agrega la aceptación de soluciones infactible a este método, el recuento llega a un total de 30 metaheurísticas con el diseño de PSO de la última fila de la tabla, que tiene cinco artículos que lo utilizan (Optimización de enjambre de partículas con soluciones infactibles).

Los algoritmos genéticos mejorados añaden otros componentes como la aceptación de soluciones no factibles o búsqueda local para producir algoritmos meméticos. Estos diseños mejorados aumentan el recuento de métodos basados en algoritmos genéticos a 106 metaheurísticas. Finalmente, el recuento llega a 129 si se consideran diseños algorítmicos más complejos: algoritmos genéticos co-evolutivos (con la colaboración de poblaciones de soluciones de nivel superior e inferior), medidas de distancia para mantener la diversidad poblacional en algoritmos genéticos multiobjetivo y algoritmos meméticos con archivado de soluciones.

Las metaheurísticas clásicas de solución única completan la tabla con el recocido simulado (generación aleatoria de soluciones, modificación aleatoria, aceptación de movimientos no mejorados) y la búsqueda tabú (generación aleatoria de soluciones, búsqueda local, memoria, aceptación de movimientos no mejorados) como los otros diseños algorítmicos preferidos. Sin embargo, hay que señalar que estas metaheurísticas de solución única representan solo 11 métodos en los diseños algorítmicos preferidos.

Como se puede observar en la Tabla 2.3, los diseños más populares de algoritmos corresponden a metaheurísticas que, en cada iteración, mantienen una población de soluciones o agentes para explorar ampliamente el espacio de búsqueda de las BPPs (por ejemplo, algoritmos genéticos, optimización por enjambre de partículas y estrategias evolutivas). Entre todos los artículos en la base de datos, el porcentaje de metaheurísticas que emplean una población de soluciones (85 %) es más de cinco veces mayor que el porcentaje de metaheurísticas de solución única (15 %), como se muestra en la Figura 2.8a. Las metaheurísticas basadas en poblaciones aprovechan el principio de paralelismo implícito inherente a este tipo de enfoque. Sin embargo, como se ilustra en la Figura 2.8b, las implementaciones paralelas explícitas de metaheurísticas para las BPPs son relativamente raras y representan solo el 5 % de los métodos propuestos [84, 65, 218, 209, 293, 244, 48, 49, 83, 85, 86, 366, 336].

En este punto, la respuesta a la pregunta Q5 está clara: las metaheurísticas basadas en poblaciones son las más ampliamente utilizadas para resolver BPPs, en particular, los métodos más populares son los algoritmos evolutivos con un conjunto de soluciones en competencia o la optimización por enjambre de partículas con un conjunto de agentes que colaboran en el proceso de búsqueda.

Diseño de metaheurística	Nombre	Referencias
Generación aleatoria de soluciones / Recombinación de soluciones / Modificación aleatoria / Competencia entre conjuntos de soluciones (evolución)	Algoritmo genético	Wu [369],Liu [206],Yin [381],Hejazi et al. [151],Yin [382],Ceylan and Bell [82],Huang and Liu [155],Wang et al. [361],Karoonsoontawong and Waller [171],Sun et al. [331],Wang et al. [355],Arroyo and Fernández [29],Osman et al. [264],Xu et al. [372],Zhang et al. [397],Yamada et al. [374],Mesbah et al. [244],Fan and Machemehl [122],Kuo and Han [182],Menasri et al. [241],Hosseinia [154],Arroyo and Fernández [30],Camacho-Vallejo et al. [66],Li [192],Liang and Miikkulainen [203],Camacho-Vallejo et al. [67],Guo [142],Sinha et al. [315],Suryan et al. [333],Du et al. [118],Sinha et al. [316],Xiao et al. [370],Alves and Antunes [19],Ait Laamim et al. [8],Amirtaheri et al. [21],Camacho-Vallejo and Garcia-Reyes [65],Cheraghalipour et al. [94],Calvete et al. [61],Watada et al. [363],Sinha et al. [318],Saeidi-Mobarakeh et al. [287],Behnia et al. [38],Huang et al. [157],Louati et al. [209],Ye et al. [380],Soares et al. [321],Leiber and Reinhart [191],Khanduzi and Rastegar [173],Ma et al. [216],Adibi [6],Saghehei et al. [288],Camacho-Vallejo et al. [69],Cheraghalipour and Roghanian [93],Gao and Ma [130],Maleki et al. [220], Hammami et al. [146],Yang et al. [375],Lee et al. [189],Qing-cheng and Zhong-zhen [276],Tingfa et al. [344],Zhang and Liu [393],Min and Guo [247],Saharidis et al. [289],Zhong and Lin [400],Song et al. [322],Zhang et al. [392],Chen and Yang [89],Yang et al. [378],Jiang et al. [164],Feng et al. [124],Gao et al. [132],Yu et al. [383],Ziar et al. [403],Li et al. [201],Sinha et al. [316]. **Total: 79**
Generación aleatoria de soluciones / Movimiento diferencial de vectores / Colaboración entre conjuntos de soluciones o agentes (estigmergia)	Optimización por enjambre de partículas	Lin et al. [205],Kuo and Huang [183],Marić et al. [225],Kheirkhah et al. [174],Marić et al. [226],Carrasqueira et al. [77],Du et al. [118],Cheraghalipour et al. [94],Behnia et al. [37],Yuan et al. [388],Martínez et al. [233],Behnia et al. [38],Rizk-Allah and Abo-Sinna [284],Soares et al. [321],Esfahani et al. [121],Hayashi et al. [149], Tabrizi et al. [335],Ganesan et al. [129], [363],[220],Wang et al. [360],Gao and Liu [131],Fathollahi-Fard et al. [123]. **Total: 25**
Generación aleatoria de soluciones / Recombinación de soluciones / Modificación aleatoria / Competencia entre conjuntos de soluciones (evolución) / Aceptación de soluciones no factibles	Algoritmo genético con soluciones no factibles	Sakawa [294],Nishizaki et al. [258],Nishizaki and Sakawa [257], Li and Wang [194]Li and Wang [193],Hecheng and Yuping [150],Calvete et al. [56],Calvete et al. [57],Li and Wang [195], Li et al. [198], Li et al. [200], Calvete and Galé [55], Wang et al. [362], Sinha et al. [309],Yang and Chen [379]. **Total: 17**
Generación aleatoria de soluciones / Recombinación de soluciones / Modificación aleatoria / Búsqueda local / Competencia entre conjuntos de soluciones (evolución)	Algoritmo memético	Yamada et al. [374],Sinha et al. [310],Bostian et al. [48],Handoko et al. [147],Zhong et al. [399],Maldonado-Pinto et al. [219],Guo et al. [143],Biesinger et al. [44],Casas-Ramírez et al. [80],Mejía-de Dios et al. [239]. **Total: 10**
Generación aleatoria de soluciones / Recombinación de soluciones / Modificación aleatoria / Competencia entre conjuntos de soluciones (evolución) / Colaboración entre conjuntos de soluciones o agentes (estigmergia)	Algoritmo genético co-evolutivo	Oduguwa and Roy [261], Legillon et al. [190], Kheirkhah et al. [174], Carrasqueira et al. [77], Kieffer et al. [175], Camacho-Vallejo and Garcia-Reyes [65],[292],Tagawa et al. [336], Said et al. [293]. **Total: 9**
Generación aleatoria de soluciones / Recombinación de soluciones / Modificación aleatoria / Competencia entre conjuntos de soluciones (evolución) / Distancia entre soluciones	Algoritmo genético (multiobjetivo) con manejo de diversidad	Chen et al. [88], Sinha [305], Frantsev et al. [127], Wang et al. [359], Bostian et al. [49],Whittaker et al. [366], Ruano-Daza et al. [286], Jerbi et al. [160]. **Total: 8**
Generación aleatoria de soluciones / Modificación aleatoria / Aceptación de movimientos no mejorados	Recocido simulado	Du et al. [118],Ghaffarinasab and Motallebzadeh [137],Chalmardi and Camacho-Vallejo [87], Sun et al. [332],Khanduzi and Rastegar [173],Maleki et al. [220]. **Total: 6**
Generación aleatoria de soluciones / Recombinación de soluciones / Modificación aleatoria / Búsqueda local / Competencia entre conjuntos de soluciones (evolución) / Memoria	Algoritmo memético con archivado de soluciones	Sinha et al. [309], Biesinger et al. [41], Biesinger et al. [42], Biesinger et al. [43]Biesinger et al. [44],Mamun et al. [221]. **Total: 6**
Generación aleatoria de soluciones / Búsqueda local / Memoria / Aceptación de movimientos no mejorados	Búsqueda Tabú	Rajesh et al. [278], Lan et al. [186],Lukač et al. [215], Aksen and Aras [10],Aksen and Aras [11]. **Total: 5**
Generación aleatoria de soluciones / Movimiento diferencial de vectores / Colaboración entre conjuntos de soluciones o agentes (estigmergia) / Aceptación de soluciones no factibles	Optimización por enjambre de partículas con soluciones no factibles	Li et al. [202],Jiang et al. [165], Hosseini et al. [153],Tawhid and Paluck [342],Zobaa et al. [405]. **Total: 5**

Tabla 2.3: Diseños de algoritmos más populares en la solución de BPPs con metaheurísticas

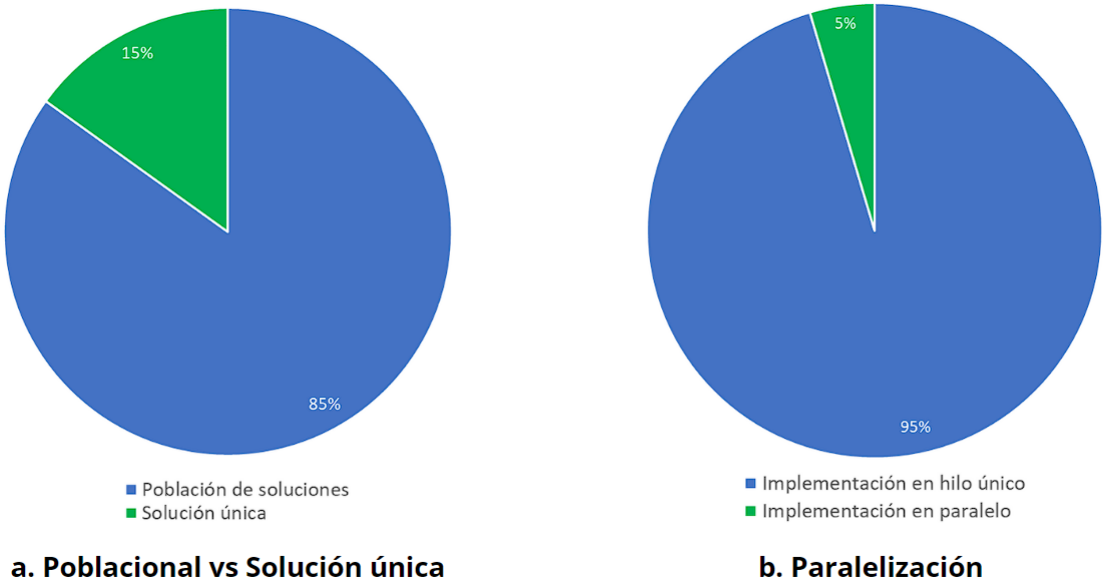


Figura 2.8: Características importantes de las metaheurísticas para BPP

Ahora, dirigimos nuestra atención al número de componentes en las metaheurísticas y su aparición individual en los documentos de la base de datos. La Figura 2.9 representa esta información. En esta revisión, incluimos dieciséis componentes diferentes que pueden ser utilizados para diseñar una metaheurística. Sin embargo, la Figura 2.9a muestra que el número de componentes utilizados en una metaheurística es mucho menor. Incluso los métodos más complejos incluyen solo la mitad del total que habíamos considerado. Por ejemplo, la metaheurística que tiene el conjunto más grande de componentes en su diseño es un algoritmo memético desarrollado por [401] para la solución de un problema de enrutamiento de vehículos. Este método hibrida un algoritmo genético y búsqueda tabú con los siguientes componentes: construcción de soluciones iniciales, recombinación y modificación aleatoria de soluciones, competencia entre conjuntos de soluciones (tomada de algoritmos genéticos), y búsqueda local, memoria, aceptación de movimientos no mejoradores y reinicio de búsqueda (tomados de búsqueda tabú), y la distancia entre soluciones para controlar la diversidad de la población. En contraste, los diseños más simples incluyen solo dos componentes, que son: (i) estrategias evolutivas (1+1) (y sus variantes) con generación aleatoria de soluciones iniciales y su modificación aleatoria (mutación) [156, 93]; (ii) metaheurísticas que construyen soluciones iniciales y las mejoran con búsqueda local de manera variable en vecindad [265, 104] o utilizando procedimientos de búsqueda adaptativa aleatorizados codiciosos (GRASP, por sus siglas en inglés) [117]; y, finalmente, (iii) un método de optimización de colonia de hormigas que recurre a la construcción y colaboración entre soluciones (estigmergia) [58]. Excepcionalmente, una búsqueda aleatoria presentada en [171] solo tiene un componente.

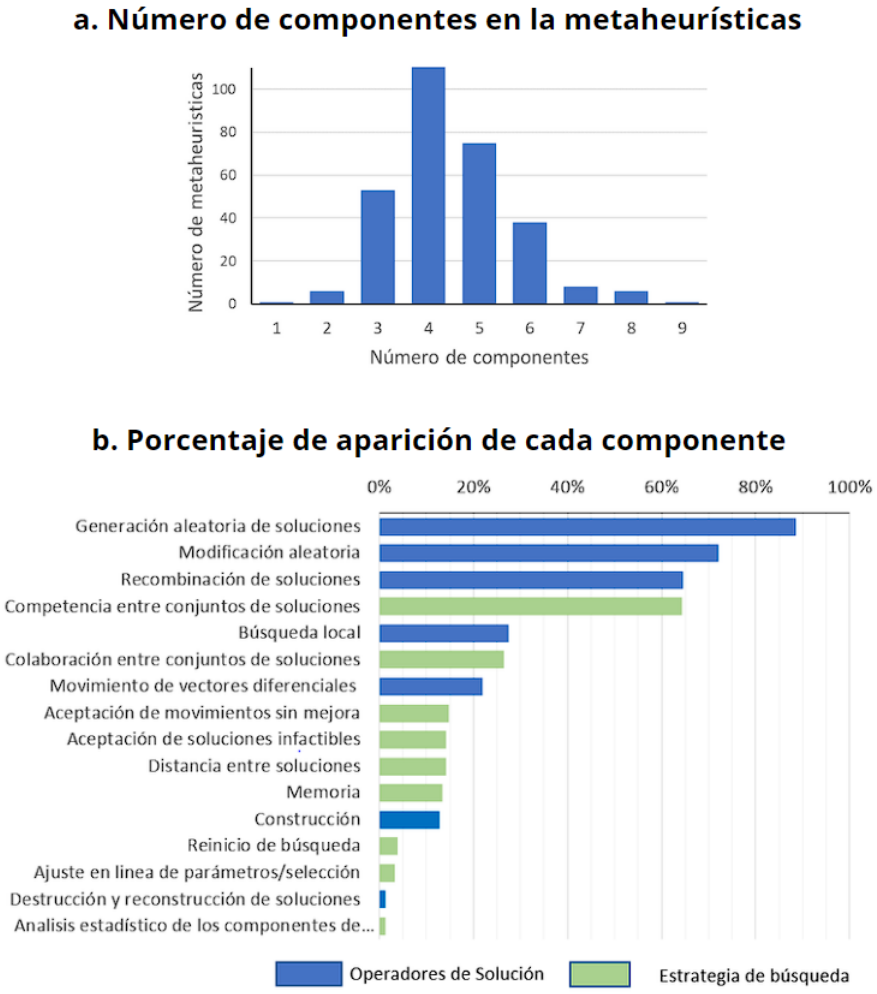


Figura 2.9: Número de componentes para cada metaheurística y su frecuencia de aparición

Interesantemente, la Figura 2.9.b muestra que el componente más popular en la solución de BPPs es la generación aleatoria de soluciones, presente en 89 % de las metaheurísticas diseñadas para resolver estos problemas. Este resultado se puede explicar por la dificultad de utilizar otros mecanismos para la generación de soluciones iniciales.

El segundo componente más importante para el diseño de las metaheurísticas para BPPs es la modificación aleatoria de soluciones, presente en 71 % de los métodos. Este componente aparece en forma de operadores de mutación en algoritmos evolutivos (estrategias evolutivas, algoritmos genéticos y evolución diferencial, entre otros) y como operadores de perturbación de metaheurísticas con búsqueda local. En ambos casos, se utiliza principalmente como un mecanismo de diversificación. También es la estrategia de exploración de vecindario en el diseño clásico de las metaheurísticas de recocido simulado. Es importante destacar que los dos componentes más populares se encuentran tanto en metaheurísticas de población como en metaheurísticas de solución única.

El tercer componente más popular es la recombinación de soluciones, presente en 64 % de los métodos. Esto es bastante natural dado que los algoritmos genéticos son la metaheurística más popular para la solución de BPPs. Sin embargo, este mecanismo de intensificación de la búsqueda también es importante en otras metaheurísticas de población, como la evolución diferencial [218, 25, 402, 179, 299, 24, 35, 199], y los métodos de búsqueda dispersa (*scatter search*) [141, 270, 68].

Dado la popularidad de las metaheurísticas evolutivas para la solución de BPPs, la primera estrategia de control que aparece entre los componentes más populares es la competencia entre conjuntos de soluciones, presente en el 64 % de los métodos revisados. La competencia de soluciones en cada iteración, donde se seleccionan soluciones con valores objetivos altos de la población existente, es la estrategia de control de búsqueda que caracteriza a los algoritmos evolutivos [36]. Estos métodos incluyen algoritmos genéticos, estrategias evolutivas, algoritmos meméticos, búsqueda dispersa, evolución diferencial y muchas otras metaheurísticas bioinspiradas [208]. La función objetivo del nivel superior es el criterio utilizado en el proceso de selección en la mayoría de las metaheurísticas evolutivas para la solución de BPPs, y el elitismo, donde se mantiene una proporción de las mejores soluciones en la población en todo momento, es un elemento clave de una metaheurística evolutiva efectiva [1]. En consecuencia, la mayoría de los algoritmos evolutivos para BPPs adoptan este principio.

Otras extensiones interesantes y elementos clave deben ser resaltados. Primero, los enfoques co-evolutivos (ver Tabla 2.5) adoptan una estrategia diferente. En estos métodos, la población de soluciones seguidoras utiliza su correspondiente función objetivo en el proceso de selección. Además, se debe idear un mecanismo para el intercambio de soluciones/información entre las poblaciones del líder y del seguidor. Segundo, se encontraron algunos algoritmos genéticos/meméticos de estado estable en la revisión [15, 41, 42, 43, 44]. En esta variante de los algoritmos genéticos, solo se deriva una nueva solución candidata en cada iteración para reemplazar otra solución ya presente en la población [36]. Tercero, aparte del algoritmo memético presentado en [401], las estrategias explícitas de gestión de diversidad para la selección de soluciones que ingresarán a la población se utilizan exclusivamente en métodos de búsqueda dispersa [68, 270, 141, 78, 128]. Cuarto, para los BPPs multiobjetivo, los conceptos de dominación y eficiencia deben incluirse en el proceso de selección, y NSGAII es el algoritmo genético multiobjetivo más comúnmente utilizado para esta clase de BPPs [110, 312]. Finalmente, cuando se permiten soluciones no factibles, la incompatibilidad de las soluciones debe ser considerada en el proceso de selección [108].

La búsqueda local es el cuarto componente más popular, presente en 27 % de los métodos. Este mecanismo de intensificación es el componente distintivo de las meta-

heurísticas de solución única como la búsqueda tabú [10, 11, 215, 186, 278, 136, 364, 15, 181, 52, 105, 16, 117, 266, 267, 70, 169, 401], procedimientos de búsqueda aleatoria adaptativa basados en estrategias greedy [117, 214], búsqueda local iterada [341, 90] y búsqueda en vecindarios variables. En esta última metaheurística, se exploran más de una vecindad durante la búsqueda, lo cual es común al resolver problemas de ubicación y enrutamiento [10, 104, 225, 226, 105, 17, 226, 63]. Como se muestra en la Tabla 2.3, también es importante como un mecanismo de hibridación en el diseño de algoritmos meméticos.

Otros dos componentes aparecen en más del 20 % de las metaheurísticas para BPPs; la colaboración entre conjuntos de soluciones (27 %) y los movimientos diferenciales de vectores (22 %). Estos dos componentes utilizados juntos corresponden al diseño de la optimización por enjambre de partículas (PSO, por sus siglas en inglés), que es muy popular en la solución de BPPs, como se muestra en la Tabla 2.3. En la optimización por enjambre de partículas, las soluciones colaboran en el cálculo del operador de velocidad que mueve aleatoriamente cada solución hacia la mejor (es decir, el componente de movimiento diferencial de vectores del PSO).

La colaboración también aparece en otras metaheurísticas. Por ejemplo, la colaboración entre agentes es la característica principal de la optimización por colonia de hormigas [58, 118], y la colaboración entre las poblaciones de soluciones de los niveles superior e inferior es la característica principal de los enfoques co-evolutivos revisados en la Tabla 2.5.

El movimiento de una solución hacia otra solución utilizando operaciones de vectores también es un componente clave de la evolución diferencial como su nombre sugiere [218, 25, 402, 179, 299, 24, 35, 199]. Path relinking también se puede ver como una especie de movimiento diferencial de vectores en un espacio discreto, pero este componente se utiliza raramente en la solución de BPPs [219]. Además, como se señala en [208], el movimiento diferencial de vectores también se utiliza en varios algoritmos inspirados en la naturaleza que imitan o extienden el diseño del PSO: optimización de moscas de la fruta [357], algoritmo de luciérnagas [323, 240, 207], algoritmo de optimización de ballenas [211], algoritmo de colonia artificial de abejas [91, 363], algoritmo de lobo gris [252, 121, 123], algoritmo de enjambre de salpas [123] y algoritmo de libélula [220].

Cinco componentes más aparecen en más del 10 % de los métodos:la aceptación de movimientos no mejorados (15 %), la aceptación de soluciones inviables (14 %), el uso de la distancia entre soluciones (14 %), la memoria (13 %) y la construcción (13 %).

Como se muestra en la Tabla 2.3, la aceptación de movimientos no mejorantes es clave en el diseño de métodos como la búsqueda tabú y el recocido simulado, por lo que

no se discutirá de nuevo. Por otro lado, la aceptación de soluciones inviables, que violan algunas de las restricciones del problema, se utiliza en metaheurísticas para BPPs en dos enfoques principales. El primero utiliza un término de penalización (en el nivel superior) para guiar la búsqueda hacia soluciones factibles [258, 257, 294, 194, 193, 195, 150, 57, 200, 55, 362, 309, 136, 202, 299, 165, 204, 354, 199, 211, 279, 312, 314, 342, 405, 56]. El segundo mecanismo no requiere penalización y se utiliza principalmente en la selección de nuevas soluciones en metaheurísticas de población, por ejemplo, en la selección por torneo de algoritmos genéticos [108]: se prefieren las soluciones factibles sobre las no factibles, entre las soluciones factibles se prefiere aquella con la mejor función objetivo, pero entre las soluciones no factibles se prefiere aquella con la menor infactibilidad [24, 351, 395, 396, 316, 310, 25, 180].

La distancia entre soluciones se utiliza principalmente en metaheurísticas multiobjetivo en forma de distancias de agrupamiento para preservar la diversidad de la población y el archivo de soluciones no dominadas [286, 88, 359, 49, 366, 110, 111, 394, 395, 20, 144, 267, 5, 62, 3, 4, 306, 371, 279, 308, 48, 312, 314]. Del mismo modo, en las metaheurísticas de una sola objetivo, las distancias entre soluciones también se utilizan como criterios de terminación [305, 127], como control de penalización por violación de restricciones [354], como medida de diversidad de la población de soluciones para guiar la búsqueda [160, 259, 323, 240, 211, 207, 12, 401, 384] e incluso como guía para modelos sustitutos [316, 25]. Las metaheurísticas de búsqueda de dispersión y los enfoques mejorados de búsqueda tabú son conocidos por el uso de medidas de distancias como un control determinista de la diversidad del conjunto de referencia o conjunto de soluciones que utilizan [52, 68, 141, 78, 270, 128].

La memoria aparece en un 13 % de las metaheurísticas analizadas. Aparte de su uso clásico en la búsqueda tabú [136, 364, 52, 16, 70, 169, 401, 278, 186, 215, 15, 181, 10, 11, 105, 117, 266, 269, 391, 302], también se utiliza para almacenar soluciones en un archivo en metaheurísticas multiobjetivo [267, 308, 252, 7] para acelerar las evaluaciones computacionalmente costosas en el nivel inferior [113, 41, 159, 304, 221, 309, 212, 312, 313, 42, 43, 44] y para guiar a los operadores de las metaheurísticas, por ejemplo, recombinación de soluciones [227, 83] o construcción [63].

La construcción de soluciones utilizando los datos del problema se utiliza en un 13 % de las metaheurísticas analizadas en la revisión de muchas maneras diferentes. La construcción aleatorizada es un componente clave para obtener soluciones iniciales en el mecanismo de arranque múltiple de GRASP [117, 214]. La construcción de soluciones también se utiliza para inicializar la búsqueda en metaheurísticas de una sola solución que mejoran iterativamente una solución inicial [181, 105, 265, 17, 341, 70, 63, 90].

La construcción aleatorizada también se utiliza como alternativa para generar poblaciones de alta calidad en algoritmos genéticos y meméticos [229, 227, 248, 92, 62, 246, 401], enjambres en la optimización por enjambre de partículas [118], conjuntos de referencia en la búsqueda por dispersión [68, 78, 270] y otras metaheurísticas de población [79, 259, 252]. Finalmente, la construcción aleatorizada de soluciones también es un componente clave en la optimización por colonia de hormigas. En esta metaheurística, el conjunto de hormigas construye soluciones de manera iterativa, guiado por la feromona y la información codiciosa del problema [58, 60, 283, 23, 269].

Search restart está presente en solo un 4 % de las metaheurísticas analizadas. Este componente se utiliza casi exclusivamente en métodos similares a la búsqueda tabú como un mecanismo de diversificación [136, 364, 52, 16, 70, 169, 401, 302]. Las únicas excepciones son en [134, 225, 226, 12] donde se introducen reinicios en otras metaheurísticas.

El ajuste de parámetros en línea también aparece en un 4 % de los métodos, para ajustar parámetros o aproximaciones de nivel inferior en metaheurísticas adaptativas o modelos sustitutos [358, 356, 111, 299, 20, 204, 144, 238, 214, 197, 302, 123].

Finalmente, dos componentes se utilizan escasamente en la solución de BPPs con metaheurísticas y sus versiones híbridas: la destrucción y construcción iterativa de soluciones (1 %) [90, 4, 3, 5], y el análisis estadístico de los componentes de la solución (1 %) [79, 304, 63]. Como resultado, las metaheurísticas basadas en la búsqueda adaptable de vecindario grande, los métodos de entropía cruzada o las metaheurísticas reactivas son bastante escasas.

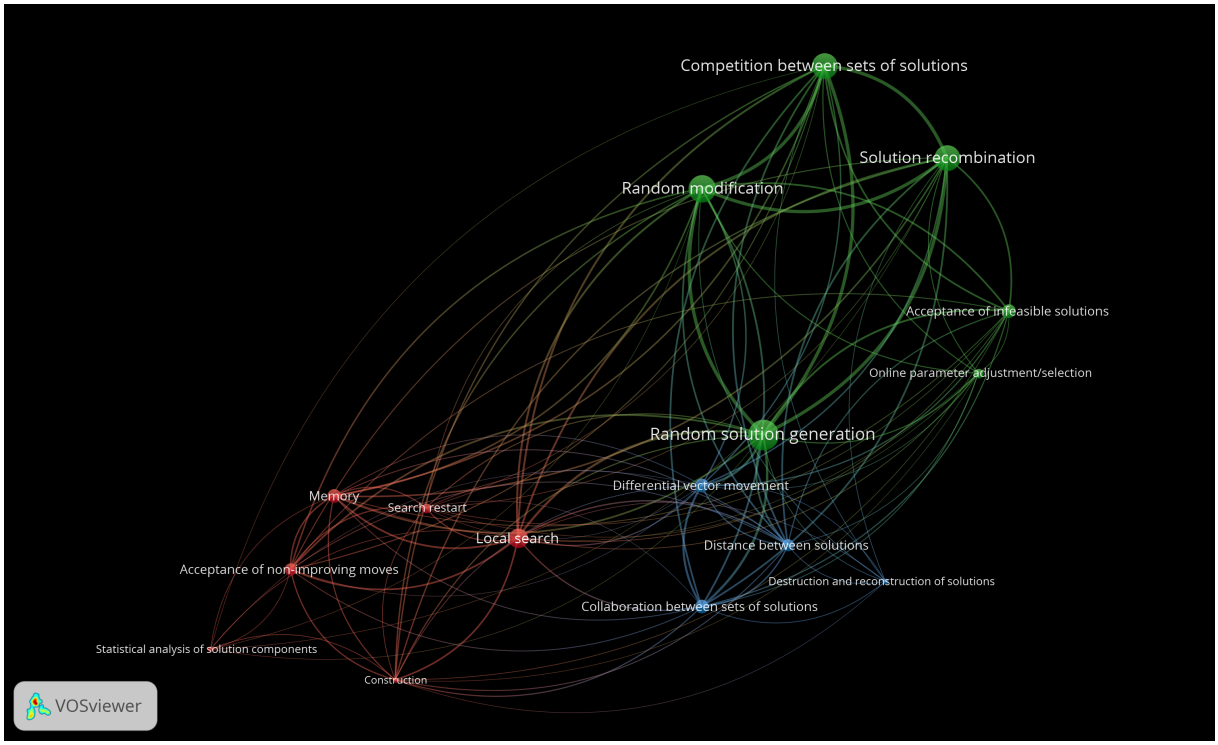


Figura 2.10: Mapa de los componentes de metaheuristicas para BPP

Para resumir el análisis, la Figura 2.10 presenta un mapa de los diferentes componentes creado con el software VOSviewer [348]. Los componentes que están cerca uno del otro y con el mismo color se utilizan con frecuencia juntos, los componentes centrales son los que aparecen más, mientras que los componentes en la periferia aparecen menos. En este punto, utilizando un umbral del 20 %, la respuesta a la Pregunta 6, que son los principales componentes que suelen aparecer en las metaheurísticas dedicadas a resolver BPPs, es la siguiente: generación aleatoria de soluciones (89 %), modificación aleatoria (71 %), recombinación de soluciones (64 %), competencia entre conjuntos de soluciones (evolución) (64 %), búsqueda local (27 %), colaboración entre conjuntos de soluciones/agentes (estigmergia) (27 %) y movimiento de vectores diferenciales (22 %). Es notable que la mayoría de estos componentes son operadores de soluciones y las dos estrategias de control de búsqueda en la parte superior son la evolución y la estigmergia.

Aspectos binivel de las metaheuristicas revisadas

Basándonos en las ideas descritas en las Secciones 2.2 y 2.2.1, ahora nos dirigimos a la manera en que se consideran los aspectos conceptuales binivel en cada metaheurística revisada. Recuerde que principalmente se siguen cuatro enfoques para resolver un Problema de Programación Binivel (BPP, por sus siglas en inglés): el enfoque anidado, una

reformulación equivalente de un solo nivel, un esquema co-evolutivo y una transformación en un problema biobjetivo.

El enfoque anidado es el más común cuando se diseña una metaheurística para resolver un BPP. En este enfoque, se resuelve el problema del seguidor para cada decisión fija del líder, ya sea de manera óptima o heurística. En la solución óptima, se utilizan enfoques que incluyen métodos de programación matemática, algoritmos exactos y enfoques exhaustivos, entre otros. En el enfoque heurístico, el problema del seguidor se resuelve de manera aproximada utilizando métodos que incluyen relajación, heurísticas, sustitutos, simulación y redes neuronales, entre otros.

Como se mencionó en la Sección 2.2.1, para que un problema de nivel binivel esté bien planteado, se debe demostrar o discutir la unicidad de la reacción del seguidor. Si el problema no está bien planteado, se debe realizar una suposición, ya que el problema del seguidor tiene múltiples soluciones óptimas para una decisión fija del líder. Se pueden utilizar enfoques optimistas o pesimistas.

La Tabla 2.4 muestra todas las metaheurísticas revisadas agrupadas en dos columnas según si el problema del seguidor se resuelve mediante métodos óptimos o aproximados. Dentro de cada columna, las metaheurísticas se agrupan según si se asegura una única solución óptima, si se asume un enfoque optimista o pesimista, o si no se proporciona información.

Como se puede observar en la columna correspondiente a la reacción óptima del seguidor, la práctica más común es omitir la manera en que se manejan soluciones óptimas únicas o múltiples. En 102 de los 263 articulos revisados, no se proporciona información sobre este aspecto clave del problema binivel. Una solución óptima única del problema del seguidor para una decisión dada del líder solo se considera en 9 documentos. Los enfoques optimista y pesimista solo se han asumido en 6 y 3 documentos, respectivamente.

De manera similar, para la reacción aproximada del seguidor, en 94 de los articulos revisados no hay información sobre la manera en que se manejan múltiples soluciones óptimas del seguidor. Además, solo un documento asume una solución única del problema del seguidor bajo este enfoque, 11 documentos asumen la versión optimista y no hay documentos que consideren la versión pesimista.

Otro enfoque al diseñar metaheurísticas para resolver BPPs es reformular el BPP en un modelo de un solo nivel equivalente primero, y luego desarrollar una metaheurística para la reformulación. Este es el caso de 20 de los 263 articulos revisados. Estos articulos se enumeran en la primera columna de la Tabla 2.5.

Tabla 2.4: Metaheuristicas aninadas para resolver BPPs

Reacción del Seguidor	Optima	Aproximada
Unica **Total: 10**	Liu [206], Nishizaki et al. [258], Yin [382], Maldonado-Pinto et al. [219], Casas-Ramírez and Camacho-Vallejo [78], Díaz et al. [117], Casas-Ramírez et al. [80], Alanís-López et al. [12], Calvete et al. [63]. **Total: 9**	Yang and Wu [376]. **Total: 1**
Optimista **Total: 16**	Li and Wang [194], Casas-Ramírez et al. [79], Alves and Antunes [19], Nucamendi-Guillén et al. [259], Calvete et al. [61], Camacho-Vallejo et al. [69] **Total: 6**	Halter and Mostaghim [145], Zhang et al. [394], Sinha et al. [308], Angelo and Barbosa [23], Camacho-Vallejo et al. [67], Casas-Ramírez et al. [79], Mejía-de Dios and Mezura-Montes [237], Camacho-Vallejo and Garcia-Reyes [65], Abbassi et al. [5], Karaja et al. [169], Abbassi et al. [4] **Total: 10**
Pesimista **Total: 3**	Ren et al. [279], Alekseeva et al. [16], Alves and Antunes [19] **Total: 3**	- **Total: 0**
No hay información **Total: 196**	Mathieu et al. [235], Wen and Huang [364], Yin [381], Ceylan and Bell [82], Karoonsoontawong and Waller [171], Sun et al. [331], Zhao and Gu [398], Zhu et al. [402], Lee et al. [189], Qing-cheng and Zhong-zhen [276], Li and Wang [193],Li and Wang [194], Koh [179], Marinakis et al. [229], Wang et al. [358], Lukač et al. [215], Hecheng and Yuping [150], Alekseeva et al. [15], Arroyo and Fernández [29], Calvete et al. [57], Kuo and Huang [183], Osman et al. [264], Xu et al. [372], Zhang et al. [397], Yamada et al. [374], Chen et al. [88], Li and Wang [195], Küçükaydın et al. [181], Min and Guo [247], Calvete et al. [58], Zhong and Lin [400], Kuo and Han [182], Li and Wang [196], Fan and Machemehl [122], Mesbah et al. [244], Miandoabchi and Farahani [246], Sinha [305], Wang et al. [362], Aksen and Aras [10], Song et al. [322], Brotcorne et al. [52], Davydov et al. [104], Marić et al. [225], Aksen and Aras [11], Calvete et al. [60], Arroyo and Fernández [30], Zhang et al. [392], Koh [180], Alves and Costa [20], Baskan and Ceylan [35], Biesinger et al. [41], Davydov et al. [105], Panin et al. [265], Lin et al. [204], Wang et al. [354], Camacho-Vallejo et al. [66], Wang et al. [359], Bostian et al. [48], Aliakbarian et al. [17], Islam et al. [158], Biesinger et al. [42], Marić et al. [226], Camacho-Vallejo et al. [68], González Velarde et al. [141], Zhong et al. [399], Biesinger et al. [43], Sinha et al. [315], Pérez Posada et al. [270], Parvasi et al. [266], Xu et al. [373], Huang et al. [156], Chaabani et al. [85], Jia et al. [162], Ait Laamim et al. [8], Biesinger et al. [44], Chalmardi and Camacho-Vallejo [87], Zhang et al. [391], Yuan et al. [388], Chen et al. [92], Shouwen et al. [302], Huang et al. [157], Calvete et al. [62], Soares et al. [321], Gao and Liu [131], Ye et al. [380], Chen et al. [90], Maleki et al. [220], Lüer-Villagra et al. [214], Sun et al. [332], Momenitabar et al. [252], Camacho-Vallejo et al. [70], Ganesan et al. [129], Khanduzi and Rastegar [173], Gao et al. [132], Ma et al. [217], Peng et al. [269], Yu et al. [384], Yu et al. [383], Saghehei et al. [288], Gao and Ma [130], Jiang et al. [164], Ziar et al. [403] **Total: 102**	Wu [369], Sahin and Ciric [291], Nishizaki and Sakawa [257], Karoonsoontawong and Waller [171], Li et al. [202], Lan et al. [186], Marinakis and Marinaki [227], Lin et al. [205], [306], Yang and Chen [379], Saharidis et al. [289], Gallo et al. [128], Gao et al. [134], Feng et al. [124], Frantsev et al. [127], Tilahun et al. [343], Segundo et al. [299], Xu et al. [371], Angelo et al. [24], Calvete et al. [59], Deb et al. [113], Menasri et al. [241], Zhang et al. [395], Marinakis and Marinaki [228],Angelo et al. [25], Sinha et al. [309], Sinha et al. [310], Sinha et al. [311], Gupta and Ong [144], Biesinger et al. [42], Liang and Miikkulainen [203], Menasri et al. [242], Bostian et al. [49], Chaabani et al. [83], Handoko et al. [147], Lu et al. [212], Sinha et al. [313], Sinha et al. [312], Sinha et al. [314], Guo [142], Guo et al. [143], Suryan et al. [333], Islam et al. [159], Carrasqueira et al. [77], Chaabani et al. [84], Du et al. [118], Wang et al. [357], Sinha et al. [316], Sinha et al. [316], Whittaker et al. [366], Ghaffarinasab and Motallebzadeh [137], Chen and Yang [89], Ruano-Daza et al. [286], Yang et al. [378], Xiao et al. [370], Amirtaheri et al. [21], Tabrizi et al. [335], Singh et al. [304], Tang et al. [341], Wang et al. [353], Biesinger et al. [44], Cheraghalipour et al. [94], Magalhães and Barbosa [218], Behnia et al. [37], Paul and Rather [267], Agor and Özaltın [7], Hammami et al. [146], Yang et al. [375], Wang et al. [360], Mejía-de Dios and Mezura-Montes [238], Song et al. [323], Watada et al. [363], Chaabani et al. [86], Sinha et al. [318], Behnia et al. [38], Jia et al. [163], Tagawa et al. [336], Memarpour et al. [240], Leiber and Reinhart [191], Li et al. [201], Abbassi et al. [3],Said et al. [292],Rizk-Allah and Abo-Sinna [284], Louati et al. [209], Hossain et al. [152],Mamun et al. [221], Ma et al. [216], Saghehei et al. [288], Cheraghalipour and Roghanian [93], Mejía-de Dios et al. [239], Zhou et al. [401], Zobaa et al. [405], Lu et al. [211], Fathollahi-Fard et al. [123] **Total: 94**

Existen 19 articulos que han propuesto metaheurísticas bajo el enfoque de coevolución. En estos algoritmos, se consideran dos poblaciones, es decir, una población de soluciones del líder y otra de soluciones del seguidor. La información entre las poblaciones se intercambia de manera específica durante el proceso iterativo. Estos documentos se muestran en la segunda columna de la Tabla 2.5.

Tabla 2.5: Otros enfoques al diseñar metaheuristicas para resolver BPPs

Metaheuristicas para la reformulación de un solo nivel de un BPP	Metaheuristicas Coevolutivas
Gendreau et al. [136], Sakawa [294], Hejazi et al. [151], Yulong and Yusheng [389], Wang et al. [361], Tingfa et al. [344], Wang et al. [355], Wang et al. [356], Zhang and Liu [393], Li et al. [198], Li et al. [200], Jiang et al. [165], Wan et al. [351], Li et al. [199], Esfahani et al. [121], Hosseini et al. [153], Hayashi et al. [149], Li and Zhang [197], Hosseinia [154], Tawhid and Paluck [342] **Total: 20**	Oduguwa and Roy [261], Rajesh et al. [278], Deb and Sinha [110], Deb and Sinha [111], Campos-Rodríguez et al. [74], Legillon et al. [190], Mokhlesian and Zegordi [250], Kheirkhah et al. [174], Zhang et al. [396], Kieffer et al. [175], Camacho-Vallejo and Garcia-Reyes [65], Mohamadi et al. [248], Saeidi-Mobarakeh et al. [287], Rikhtegar et al. [283], Liu [207], Chen et al. [91], Adibi [6], Said et al. [293], Jerbi et al. [160] **Total: 19**

Vale la pena mencionar que en las metaheurísticas diseñadas para resolver las reformulaciones de un solo nivel y las co-evolucionarias, no se considera el caso de múltiples soluciones óptimas del problema del seguidor. Por lo tanto, no se realiza ni la suposición optimista ni la pesimista. Además, hay algunos artículos que han propuesto diferentes metaheurísticas para resolver el mismo problema. Esto implica que el mismo artículo puede tener dos enfoques diferentes, por ejemplo, uno anidado-óptimo y otro anidado-heurístico.

El enfoque biobjetivo no se ha considerado para desarrollar metaheurísticas para resolver BPPs. Sin embargo, es importante señalar que hay muchas metaheurísticas para resolver BPPs biobjetivos. Este último es muy diferente de su conceptualización. Para aclarar, no se han propuesto metaheurísticas para resolver BPPs a través de su transformación en un problema biobjetivo. Pero, los BPPs biobjetivos se han resuelto mediante metaheurísticas (consulte la Tabla 2.2).

Además de estos cuatro enfoques principales, se han considerado otros enfoques diferentes y novedosos. Por ejemplo, en [56] se propone un algoritmo genético que opera con puntos extremos de un BPP lineal. De manera similar, en [55] se sigue la misma idea, pero el problema del seguidor considera múltiples objetivos simultáneamente. En ese caso, se consideran soluciones débilmente eficientes. Otro enfoque es diseñar la metaheurística para manejar las soluciones del seguidor. En [155], se desarrolla un algoritmo enumerativo para el líder y se utiliza un algoritmo genético para obtener la reacción del seguidor. Además, en [192], se propone un algoritmo genético que maneja las bases asociadas con el problema del seguidor. En esa investigación, existe una solución óptima única del problema del seguidor parametrizado. Otro enfoque diferente se presenta en [19], en el que el seguidor

enfrenta un problema biobjetivo. El frente de Pareto se obtiene mediante un método exacto, pero se caracterizan las soluciones extremas. Es decir, se identifican las soluciones optimistas, pesimistas, recompensantes y engañosas, y se proporcionan al líder como la posible reacción del seguidor. Se diseña un algoritmo genético para explorar el espacio de soluciones del líder.

En este punto, somos capaces de responder a las preguntas de investigación Q7-Q10. El enfoque más común considerado en el tema bajo revisión es el enfoque anidado. En total, se ha considerado en 225 de los 263 artículos revisados (86 %). Esto responde a la Q7. Q8 también se responde afirmativamente; es decir, se han propuesto metaheurísticas para resolver reformulaciones de un solo nivel de problemas binivel en 20 artículos (8 %).

Para responder a la Q9, solo se analiza el enfoque anidado. Por un lado, el problema del seguidor parametrizado se resuelve de manera óptima en 120 de los 263 artículos revisados (46 %). Por otro lado, la reacción del seguidor se aproxima en 105 artículos (40 %). Por lo tanto, obtener soluciones viables en el sentido estricto del problema binivel es ligeramente más preferido que considerar aproximaciones. Finalmente, se discute la respuesta a la Q10. Como se puede ver en la Tabla 2.4, una gran mayoría de las metaheurísticas propuestas para resolver BPPs omiten el problema fundamental de la existencia de múltiples soluciones óptimas del seguidor. Esto ocurre en 196 de los 263 artículos revisados (75 %). Tenga en cuenta que algunos artículos consideran más de un enfoque diferente al resolver el problema del seguidor, en este caso, se cuentan solo una vez. La unicidad se asume o se demuestra solo en 10 artículos. Los enfoques optimistas y pesimistas se asumen en 16 y 3 artículos, respectivamente.

2.4 Discusión

A partir del análisis realizado en esta revisión, identificamos algunos hallazgos relevantes al diseñar metaheurísticas para resolver BPPs. Por ejemplo, las metaheurísticas basadas en poblaciones son preferidas sobre las de una sola solución. Además, en ausencia de buena información para guiar los procedimientos constructivos, la opción más popular para inicializar la búsqueda en las metaheurísticas para BPPs es la generación aleatoria de soluciones iniciales, independientemente del enfoque de metaheurística seguido.

A partir de los resultados obtenidos por las metaheurísticas basadas en poblaciones, se puede inferir que son una opción confiable para resolver este tipo de problemas complejos. Naturalmente, la selección de la metaheurística específica depende del problema bajo estudio. Los algoritmos evolutivo son el enfoque más común para muchos problemas, mientras que la optimización por enjambre de partículas está diseñada específicamente

para problemas continuos. Además, las mejoras de estas dos metaheurísticas, como permitir soluciones infactibles, la hibridación con métodos de búsqueda local y el uso de enfoques co-evolutivos, también son prácticas comunes en la solución de BPPs.

Sin embargo, es importante mencionar que no todas las metaheurísticas basadas en poblaciones encontradas en nuestra revisión de la literatura ofrecen una opción clara y adecuada para resolver un BPP. En esta revisión, también encontramos metaheurísticas basadas en metáforas que no ofrecen ideas novedosas o utilizan terminología no estándar para presentar las metaheurísticas [27]. Por ejemplo, algunas de ellas se inspiraron en volcanes [153], caballos salvajes [405], reacciones químicas [84, 5], y lluvia de ideas [259], entre otros. Esta práctica agrega un nivel de oscuridad y requiere un considerable tiempo y esfuerzo para entender cómo funcionan estos algoritmos [208]. De manera similar, también se encontraron algunas metaheurísticas basadas en metáforas controvertidas ([365, 81, 73]) como búsqueda armónica [286], algoritmo de enjambre de salpas [123], lobo gris [121], ballena [211] y optimización de luciérnagas [240, 207, 323]. Como se menciona en [27], las metaheurísticas para BPPs deben presentar su método utilizando la terminología normal y estándar de optimización, y en este caso, también los aspectos inherentes de la optimización binivel.

Otro aspecto positivo que vale la pena enfatizar es el esfuerzo significativo realizado para reducir el tiempo de cómputo. Es bien sabido que el enfoque anidado es computacionalmente costoso, por lo que muchos de los artículos revisados se centran en este aspecto. El objetivo es disminuir el esfuerzo computacional, lo cual se puede lograr mediante el desarrollo de métodos eficientes para resolver el problema del seguidor, o para aproximarlo. Las aproximaciones del seguidor han captado la atención de los investigadores en los últimos años a través de funciones de respuesta o métodos de sustitución [147], [314], [23], [366], [363], [238], [221]. Sin embargo, estas aproximaciones deben ser validadas de manera rigurosa ya sea en términos teóricos o experimentales.

Otro aspecto positivo que vale la pena enfatizar es el esfuerzo significativo realizado para reducir el tiempo de cómputo. Es bien sabido que el enfoque anidado es computacionalmente costoso, por lo que muchos de los artículos revisados se centran en este aspecto. El objetivo es disminuir el esfuerzo computacional, lo cual se puede lograr mediante el desarrollo de métodos eficientes para resolver el problema del seguidor, o para aproximarlo. Las aproximaciones del seguidor han captado la atención de los investigadores en los últimos años a través de funciones de respuesta o métodos de sustitución [147], [314], [23], [366], [363], [238], [221]. Sin embargo, estas aproximaciones deben ser validadas de manera rigurosa ya sea en términos teóricos o experimentales.

Cuando se aproxima la respuesta del seguidor, se debe realizar una iteración final para obtener la respuesta óptima. En otras palabras, la reacción del seguidor se aproxima para guiar la búsqueda (con soluciones semifactibles binivel), pero al final, se consideran soluciones estrictamente factibles en el sentido binivel, como en [41], [219].

Junto con las buenas prácticas, a través de este análisis también se identificaron algunas prácticas no tan buenas o deficientes. La más crítica que identificamos es la falta de rigor al manejar la reacción del seguidor. Esta preocupación es triple. La primera se refiere a las soluciones del problema del seguidor obtenidas de manera aproximada. Ya se ha discutido que estas soluciones no son estrictamente factibles en el sentido binivel, pero la conveniencia de este enfoque al resolver problemas complejos del seguidor es notable. En otras palabras, se considera una reacción racional"del seguidor. Sin embargo, estas reacciones aproximadas deben medirse o clasificarse de alguna manera. La mayoría de las metaheurísticas propuestas que resuelven el problema del seguidor utilizando otra metaheurística no lo evalúan ni establecen condiciones basadas en la calidad de la solución obtenida. Tener una reacción subóptima del seguidor puede tener un impacto significativo en la función objetivo del líder. Esta es la razón principal por la que se critica la falta de rigor al aproximar la solución del problema del seguidor. La segunda preocupación es que el espacio de reacción racional del seguidor no está debidamente caracterizado. Es decir, si la reacción del seguidor es única o si tiene múltiples soluciones óptimas. Si ocurre lo último, entonces se debe hacer una suposición: se debe usar el enfoque optimista, pesimista u otro indicado. Si no se hace ninguna suposición, el problema binivel no está bien planteado. Además, la manera en que se obtienen todas las reacciones óptimas múltiples del seguidor debe indicarse claramente. Como se puede ver en la Sección 2.3.5.2, la mayoría de los artículos revisados no especifican la forma en que se están considerando las múltiples reacciones óptimas del seguidor. Esta es una práctica común que no debe adoptarse como práctica estándar. Muchas de las metaheurísticas revisadas para resolver BPPs no son reproducibles debido a la falta de información sobre estos dos problemas importantes. La tercera preocupación está relacionada con los BPPs en los que el seguidor tiene como objetivo optimizar más de un objetivo simultáneamente. En ese caso, se proporciona una frontera de Pareto como solución del problema del seguidor. Por lo tanto, se necesita una suposición crucial con respecto a la solución elegida por el seguidor. Se espera elegir la que mejor se adapte o peor se adapte al líder (optimista o pesimista, respectivamente). Sin embargo, hay artículos que no detallan la forma en que se selecciona la reacción del seguidor de la frontera de Pareto, como en [306], [312], [394], [144], [233], [207]. Particularmente, en [267], [169], los autores se refieren a la solución "mejor comprometida", pero no se incluye información que especifique el significado de ese término.

En nuestra opinión, a pesar de la manera relajada en que se ha informado la consideración de la reacción del seguidor en la literatura, es un aspecto clave al diseñar metaheurísticas para resolver BPPs. A partir de los resultados obtenidos en esta revisión, buscamos establecer una línea base al obtener la reacción del seguidor de manera aproximada y, lo que es más importante, cuando puede haber múltiples reacciones óptimas del seguidor para una decisión dada del líder.

El caso en el que se está considerando un problema multiobjetivo en el problema del seguidor merece un tratamiento especial en lo que respecta a la solución de la frontera de pareto asociada que se considerará como la reacción del seguidor.

Finalmente, es importante destacar que debido a la relación inherente entre el líder y el seguidor, es necesario obtener la reacción del seguidor. En muchos casos, esta reacción se obtiene de manera óptima mediante algoritmos exactos de programación matemática. Por otro lado, las soluciones del líder se obtienen comúnmente mediante el uso de una metaheurística. Este enfoque coordinado puede encajar en la clasificación de matheurísticas [222], pero se ha seguido desde 1994, cuando se propuso la primera metaheurística para resolver un BPP. Esta es la razón por la que no se hace una diferenciación entre estos dos enfoques en la solución de BPP. No obstante, recientemente se han propuesto algunos métodos que exploran heurísticamente el espacio de búsqueda utilizando modelos de programación matemática (reducidos) para los problemas del líder y del seguidor (por ejemplo, el método de búsqueda kernel presentado en [297]). Estos métodos se ajustan a la categoría de matheurísticas y deberían explorarse más a fondo.

2.5 Oportunidades para usar metaheuristicas en optimización binivel

A partir de los resultados obtenidos de la exhaustiva revisión llevada a cabo y presentada anteriormente, se han extraído algunas conclusiones respecto a las buenas y malas prácticas al desarrollar metaheurísticas para resolver BPPs. También es posible señalar enfoques prometedores y áreas de oportunidad que han sido identificadas en esta revisión.

La primera sugerencia de investigación es diseñar metaheurísticas basadas en una sola solución o utilizando la búsqueda local como componente principal. Como se puede observar en los resultados presentados anteriormente, la gran mayoría de las metaheurísticas para resolver BPPs se basan en poblaciones. Creemos que considerar la reacción del seguidor antes de evaluar la función objetivo del líder desalienta el uso de este enfoque.

En problemas de optimización de un solo nivel, un movimiento dentro de un vecindario predefinido puede evaluarse de manera directa. Esto no es el caso para los BPPs. Por lo tanto, los procedimientos de búsqueda local pueden ser computacionalmente costosos, y una búsqueda guiada puede ser impredecible dependiendo de la reacción del seguidor a una decisión fija del líder. No obstante, la adaptación exitosa de metaheurísticas basadas en búsqueda local (como búsqueda local iterada o búsqueda en vecindario variable) para resolver BPPs podría ser una dirección de investigación prometedora. De manera similar, los enfoques constructivos de metaheurísticas que construyen una solución aleatoria y mejoran (por ejemplo, procedimientos de búsqueda adaptativa aleatoria) o aquellos que destruyen y reconstruyen iterativamente las soluciones (como búsqueda codiciosa iterada y búsqueda de vecindario grande adaptativa) son enfoques menos explorados en la solución de BPPs. Estos merecen una atención importante por parte de la comunidad de investigación en metaheurísticas debido a su aplicación exitosa en otros dominios [224, 330].

Además, esta revisión ha identificado algunos enfoques no explorados en el campo de las metaheurísticas basadas en poblaciones. En particular, las metaheurísticas basadas en poblaciones diseñadas para evaluaciones costosas de soluciones, como Scatter Search [281], podrían ofrecer una buena alternativa a los algoritmos genéticos utilizados principalmente en la solución de BPPs. Se encontraron pocos artículos que utilizaran este enfoque, por lo que se necesita investigación adicional en esta dirección. Otros diseños de metaheurísticas basadas en poblaciones que permanecen sin explorar para la solución de BPPs son los algoritmos meméticos con gestión de población [326] y los algoritmos genéticos con claves aleatorias sesgadas [140]; su idoneidad para la solución de este tipo de problemas debería ser explorada. En una dirección complementaria, los algoritmos de estimación de distribución [187], que explotan la población de soluciones de manera diferente a los algoritmos genéticos, constituyen una tercera dirección de investigación prometedora en el campo de las metaheurísticas basadas en poblaciones.

Recientemente, ha habido una considerable tendencia a desarrollar métodos surrogados con el objetivo de reducir la alta carga computacional de las metaheurísticas anidadas para resolver BPPs. Esta es otra dirección de investigación prometedora que se ha identificado y que puede aplicarse tanto en metaheurísticas basadas en poblaciones como en soluciones individuales.

La hibridación de metaheurísticas con otras técnicas, como métodos exactos, programación de restricciones y aprendizaje automático, puede llevar a una combinación poderosa [340], [170]. Por ejemplo, en [209], se diseñan redes neuronales convolucionales para resolver el problema del seguidor. Creemos que existen otras técnicas que se pueden utilizar para aproximar la reacción del seguidor o desarrollar modelos sustitutos. Algunos autores ya han explorado esta alternativa utilizando bosques aleatorios [203] y otras

herramientas de aprendizaje automático menos populares [147]. En consecuencia, estas técnicas de aprendizaje automático pueden mejorar el rendimiento computacional de las metaheurísticas al resolver BPPs.

Con el objetivo de aprovechar los avances tecnológicos en computadoras, la implementación de cómputo paralelo dentro de las metaheurísticas es otra dirección de investigación que merece atención. Es decir, implementar y ejecutar diferentes procesos de una metaheurística de manera paralela para reducir el tiempo de cálculo. Esto ya ha sido explorado en [218], [65], [4], con buenos resultados reportados.

Otra dirección de investigación importante, que ya se ha discutido en profundidad en la Sección 2.4, es la manera en que se debe considerar la reacción del seguidor en problemas binivel con objetivos múltiples. En particular, cuando el problema del seguidor resulta en un problema con múltiples objetivos. En este caso, se obtiene un frente de Pareto al resolver el problema del seguidor parametrizado. Por lo tanto, una de estas soluciones eficientes debe ser proporcionada al líder como la reacción del seguidor. Se deben establecer supuestos bien fundamentados, como en [19]. En nuestra opinión, no es suficiente simplemente afirmar que se elige la "mejor solución comprometida". Esto impide la reproducibilidad de los algoritmos propuestos, lo cual debe ser de la máxima importancia en esta área de investigación.

Una dirección de investigación final y muy importante que identificamos en esta revisión es el desarrollo de marcos computacionales para la solución de problemas binivel que fomenten la replicación, reutilización y diseño automatizado de metaheurísticas para estos problemas [334]. Utilizando los componentes más populares identificados en esta revisión como bloques de construcción, tales marcos podrían beneficiarse de la investigación realizada durante casi 30 años en la solución de problemas binivel.

Capítulo 3

Un problema binivel de localización de Hubs 5G que balance la conexión de usuarios

Las redes de telecomunicaciones se enfrentan a diversos desafíos en la toma de decisiones para optimizar el rendimiento y la eficiencia de estas. Uno de estas decisiones se centra en la distribución de usuarios conectados a hubs en la redes. Esta tarea es importante para garantizar que las redes no se sobresaturen manteniendo un servicio de calidad.

Este capitulo presenta un problema binivel de localización de hubs 5G que busca analizar la situación descrita. Aqui el nivel superior busca instalar hubs para que los usarios se conecten manteniendo un equilibrio, evitando la sobresaturación y cumpliendo una serie de características necesarias para el problema. Por otro lado el nivel inferior busca conectarse a estos hubs minimizando un costo asociado a conectarse a algun hub. Para resolver este problema se utilizará un algoritmo genetico aninado desarrollando un esquema de solución, realizando una experimentación númerica para evaluar la eficiencia y robustez del algoritmo en distintos escenarios. Los resultados buscan proporcionar una visión clara sobre las alternativas y permitirá obtener conclusiones relevantes para el proceso de toma de decisiones en redes de telecomunicaciones.

El resto del capitulo se dividirá de la siugiente manera. En la sección 3.1 se mostrará un modelo matematico que represente la situación del problema. En la sección 3.2 se hablará de los componentes del algoritmo genetico anidado. Finalmente en la sección 3.3 se mostraran los resultados y se tendrá una discusión de estos.

3.1 Modelo Matemático

Basándose en la situación bajo estudio descrita en la sección anterior, se van a definir los conjuntos, parámetros y variables de decisión involucrados en el problema para formular un modelo matemático. Sea I el conjunto de sitios potenciales para habilitar un centro de distribución, sea J el conjunto de diferentes tipos de centros de distribución y sea K el conjunto de usuarios que desean conectarse a los centros habilitados.

Se considera un costo fijo de instalación f_{ij} para habilitar un centro de distribución de tipo $j \in J$ en el sitio $i \in I$. Además, se asocia un costo de alquiler r_{ijk} para el usuario $k \in K$ que utiliza el centro de distribución de tipo $j \in J$ en el sitio $i \in I$. Se considera un presupuesto limitado b para el diseño e implementación de la red, y se establece una capacidad q_j que limita la conectividad de los usuarios permitidos en el centro de distribución de tipo $j \in J$. Además, se debe habilitar un número mínimo de centros de distribución de tipo $j \in J$, que se denota como p_j. Para implementar una red funcional de centros de distribución, se debe conectar un número mínimo de usuarios, que se denota como ρ_{min}.

Las variables de decisión del nivel superior son:

$$y_{ij} = \begin{cases} 1 & \text{if the hub of type } j \in J \text{ is enabled at site } i \in I. \\ 0 & \text{otherwise} \end{cases}$$

Por otro lado, las variables de decisión asociadas al nivel inferior son:

$$x_{ijk} = \begin{cases} 1 & \text{if user } k \in K \text{ is connected to the hub of type } j \in J \text{ enabled at potential site } i \in I \\ 0 & \text{otherwise} \end{cases}$$

Además, las variables continuas auxiliares l_j y u_j denotan el número mínimo y máximo de usuarios conectados a centros de distribución de tipo $j \in J$, respectivamente.

El modelo de programación bi-nivel resultante es el siguiente:

$$\min_{y,x} \max_{j \in J} \{u_j - l_j\} \tag{3.1}$$

$$\text{s.t.} \quad \sum_{i \in I} \sum_{j \in J} f_{ij} y_{ij} \leq b \tag{3.2}$$

$$\sum_{i \in I} y_{ij} \geq p_j, \qquad \forall j \in J \tag{3.3}$$

$$\sum_{j \in J} y_{ij} \leq 1, \qquad \forall i \in I \tag{3.4}$$

$$l_j \leq \sum_{i\in I}\sum_{k\in K} x_{ijk} + |K|(1 - \sum_{i\in I} y_{ij}), \qquad \forall j \in J \tag{3.5}$$

$$u_j \geq \sum_{i\in I}\sum_{k\in K} x_{ijk}, \qquad \forall j \in J \tag{3.6}$$

$$y_{ij} \in \{0,1\} \qquad \forall i \in I, j \in J, k \in K \tag{3.7}$$

donde para una variable y fija, la variable x debe ser la solución óptima del siguiente problema:

$$\min_{x} \quad \sum_{i\in I}\sum_{j\in J}\sum_{k\in K} r_{ijk}x_{ijk} \tag{3.8}$$

$$\text{s.t.} \quad \sum_{k\in K} x_{ijk} \leq q_j y_{ij} \quad \forall i \in I, \forall j \in J \tag{3.9}$$

$$\sum_{i\in I}\sum_{j\in J}\sum_{k\in K} x_{ijk} \geq |K|\rho_{min} \tag{3.10}$$

$$\sum_{i\in I}\sum_{j\in J} x_{ijk} \leq 1 \quad \forall k \in K \tag{3.11}$$

$$x_{ijk} \in \{0,1\} \qquad \forall i \in I, j \in J, k \in K \tag{3.12}$$

El problema del nivel superior está definido por las Ecuaciones (3.1)-(3.7) y tiene como objetivo equilibrar el número de usuarios conectados a los centros habilitados. En particular, se busca el equilibrio para cada tipo de centro de distribución, y en consecuencia, se obtiene una red de centros equilibrada. Esto se indica en la Ecuación (3.1), donde se busca el mínimo de los máximos centros desequilibrados. La restricción (3.2) impone el presupuesto disponible para diseñar la red de centros de distribución. La restricción (3.3) garantiza que se habiliten un número mínimo de cada tipo de centros, y la restricción (3.4) indica que un solo centro de cualquier tipo puede estar habilitado en cada sitio potencial. El número mínimo y máximo de usuarios conectados a un centro se calcula en las Ecuaciones (3.5) y (3.6), respectivamente. La naturaleza binaria de las variables de decisión del nivel superior se indica en la Ecuación (3.7).

El problema del nivel inferior, definido por las Ecuaciones (3.8)-(3.12), está parametrizado en una decisión del nivel superior. Su función objetivo busca minimizar el costo de alquiler para conectar a los usuarios a los diferentes centros habilitados, y se encuentra en la Ecuación (3.8). La restricción (3.9) indica la capacidad de los centros específicos en términos de los usuarios conectados a ellos. Se garantiza un número mínimode usuarios conectados para garantizar la funcionalidad de la red de centros mediante la restricción (3.10). Además, los usuarios deben estar conectados a un solo centro habilitado de cual-

quier tipo, lo cual se asegura mediante la restricción (3.11). Por último, la naturaleza binaria de las variables de decisión del nivel inferior se indica en la Ecuación (3.12).

Para tener definido un problema de programación bi-nivel de manera adecuada, el nivel inferior debe tener una única solución óptima para una decisión del nivel superior. En caso contrario, es decir, cuando existen múltiples soluciones óptimas en el nivel inferior, se puede asumir un enfoque optimista o pesimista [114]. El primero consiste en asumir un enfoque cooperativo. En otras palabras, de todas las soluciones óptimas múltiples del nivel inferior, se elige aquella que mejor se ajuste a la función objetivo del nivel superior. El razonamiento detrás de este enfoque es que todas las soluciones óptimas del nivel inferior conducen al costo de alquiler mínimo, pero se selecciona aquella solución que resulta en una red más equilibrada. El enfoque pesimista asume un comportamiento opuesto, es decir, el tomador de decisiones del nivel superior se prepara para el peor escenario debido a la reacción del nivel inferior. En este capítulo, se asume el enfoque optimista.

3.2 Un algoritmo genetico anidado

En esta sección se presenta el desarrollo de un algoritmo evolutivo y los componentes utilizados para el problema. Los problemas bi-nivel se consideran desafiantes desde el punto de vista computacional, clasificándose como NP-duros, incluso en su forma más simple, conocida como el caso lineal-lineal. Además, en el problema abordado en esta investigación, el nivel inferior del modelo presenta una estructura no convexa, lo cual agrega complejidad adicional a su resolución. Estas características hacen que no sea factible utilizar un enfoque tradicional de reformulación de un solo nivel para encontrar una solución óptima. Basado en estos dos incovenientes un algoritmo evolutivo anidado se propone para resolver el problema.

Particularmente, los algoritmos evolutivos (EAs, por sus siglas en inglés) han demostrado ser efectivos para resolver problemas complejos en diferentes contextos. Por ejemplo, se pueden encontrar EAs para resolver problemas de ubicación de instalaciones en referencias como [138], [350] y [236]. Además, se han desarrollado EAs para resolver problemas bi-nivel en [307], [361] y [309]. Cabe mencionar que, en los problemas bi-nivel, es necesario resolver de manera óptima el nivel inferior para una decisión dada del nivel superior. Este enfoque, conocido como enfoque anidado, se utiliza para obtener soluciones factibles en problemas bi-nivel. A continuación, se presenta un esquema general de un EA.

Algoritmo 3.2.1: Algoritmo Evolutivo Aninado

```
1  Procedure Algoritmo evolutivo
2      Crear una población inicial de individuos aleatorios
3      Fase de reparación
4      Solución del nivel inferior
5      Evaluación del fitnes
6      while criterio de parada no alcanzado do
7          Fase de Selección
8          Fase de cruza
9          Fase de reparación
10         Fase de mutación
11         Solución del nivel inferior
12         Actualizar la población en una manera elitista
13     end
14 end
```

A continuación, se presenta una explicación detallada de cada uno de los componentes involucrados en el propuesto Algoritmo Evolutivo Anidado (NEA, por sus siglas en inglés). Como se puede observar en el pseudocódigo mostrado en 3.2.1, el algoritmo crea una población inicial de individuos y se aplica una fase de reparación para asegurar que se cumplan las restricciones del nivel superior. Con el fin de obtener soluciones factibles binivel, se resuelve de manera óptima el nivel inferior para cada individuo. Luego, se realiza la evaluación de aptitud de los individuos. Se selecciona un subconjunto de los individuos con mejor aptitud para aplicar operadores evolutivos, como el cruce y la mutación. Estos operadores generan nuevos individuos y el proceso se repite. El objetivo de este esquema evolutivo es mejorar los individuos a lo largo de las generaciones.

3.2.1 Población Inicial

Como se mencionó anteriormente, la población inicial de individuos se crea de manera aleatoria. Para lograr esto, se consideran vectores de tamaño $|I|$, cuyos componentes son elementos del conjunto $J \cup 0$. Por lo tanto, un vector contiene números enteros que indican el tipo de hub habilitado en el i-ésimo sitio potencial. En caso de que aparezca un cero, implica que no se habilitan hubs en ese sitio potencial.

Se presenta un ejemplo de la codificación utilizada en la Fig. 3.1, en el cual se consideran 8 sitios potenciales y 3 tipos diferentes de hubs. El individuo ilustrado indica que

se habilitan dos hubs de tipo 1 en el segundo y octavo sitio potencial. De manera similar, se habilitan hubs de los tipos 2 y 3 en el tercer y sexto sitio potencial, respectivamente. Además, se observa que hay cuatro sitios potenciales en los que no se habilitan hubs.

Figura 3.1: Ilustración de la codificación del individuo.

A pesar de que la codificación considerada garantiza que un sitio potencial se puede utilizar para habilitar solo un tipo de hub, es posible que no se cumplan otras restricciones. Los hubs habilitados de cada tipo pueden no alcanzar su número mínimo requerido o exceder el presupuesto disponible. Por lo tanto, se requiere una fase de reparación.

3.2.2 Fase de Reparación

Una vez que se crea un individuo, la factibilidad debe ser garantizada. Si el individuo es factible, entonces se incluye en la población. En caso contrario, el individuo sufre algunos cambios estructurados con el objetivo de lograr la factibilidad. Como se mencionó anteriormente, la infactibilidad puede deberse a que no se cumpla el número mínimo de hubs habilitados o se exceda el presupuesto.

Ilustremos la fase de reparación propuesta con un ejemplo. Considere el individuo representado en la Fig. 3.1, y suponga que el número mínimo de hubs habilitados de tipo 2 y 3 es dos ($p_2 = p_3 = 2$). Es evidente que el individuo no cumple con esa restricción específica. Por lo tanto, se selecciona aleatoriamente un sitio potencial sin un hub para habilitar el tipo de hub específico. La selección de los tipos de hubs se realiza de manera lexicográfica, es decir, primero se habilita el hub de tipo 2 y, en segundo lugar, el hub de tipo 3. Una ilustración se muestra en la Fig. 3.2.

Figura 3.2: Ilustración de un individuo reparado.

Es importante mencionar que durante el proceso de habilitación de hubs faltantes, se verifica el presupuesto. Si se excede el presupuesto y no se han habilitado los hubs mínimos requeridos, el individuo es descartado.

3.2.3 Solución óptima del nivel inferior.

Para evaluar la aptitud de un individuo, se necesita la conexión de los usuarios a los hubs habilitados. Recordemos que la conexión se decide a nivel inferior, donde se minimizan los costos de conexión. En este caso, una vez que se conocen los diferentes tipos de hubs habilitados, el problema resultante a nivel inferior debe resolverse de manera óptima.

Específicamente, corresponde a un problema de programación binaria, que puede resolverse mediante un software de optimización comercial. Si el problema resultante a nivel inferior es infactible, se descarta el individuo asociado. Esto puede deberse a la falta de capacidad para satisfacer la restricción de funcionalidad de la red.

3.2.4 Evaluación del Fitness.

La aptitud asociada a un individuo se obtiene a partir de la función objetivo del nivel superior. En otras palabras, se trata del equilibrio de los usuarios conectados a los centros habilitados.

Es evidente que la evaluación de la aptitud se puede realizar después de resolver el nivel inferior.

3.2.5 Selección

La selección de los individuos que entran en la fase de cruce se realiza de manera elitista. Se selecciona un subconjunto de individuos a través de un torneo y se selecciona aquel con la mayor cantidad de torneos ganados.

3.2.6 Cruza

Para generar nuevos individuos se aplica la fase de cruza. Los individuos en el subconjunto élite se emparejan al azar con otro individuo de la población. Para cada par de individuos (padres), se realiza un cruce de un solo punto para generar dos descendientes. El primer descendiente tiene las características del primer padre antes del punto de cruce y las características del segundo padre después del punto de cruce. De manera inversa, se crea el segundo descendiente. Un ejemplo se ilustra en la Fig. 3.3.

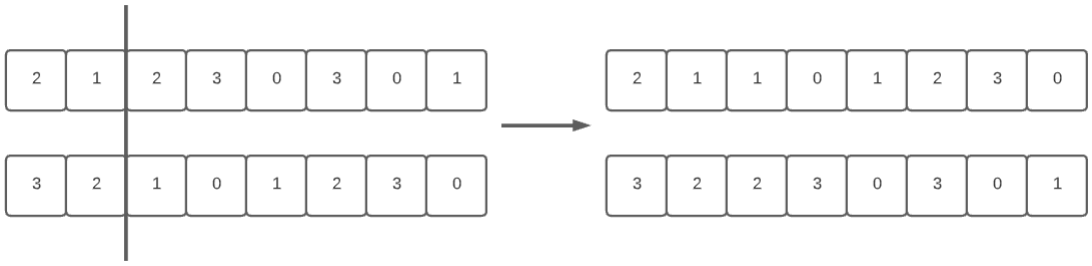


Figura 3.3: Ilustración del cruce de un solo punto.

El cruce implementado puede crear individuos infactibles. Por lo tanto, se aplica la fase de reparación descrita previamente. Si los descendientes son viables, entonces se almacenan, de lo contrario, se descartan.

3.2.7 Mutación

Los nuevos individuos creados en la fase de cruce pueden sufrir una mutación. Esta fase agrega diversidad al EA y promueve una amplia exploración del espacio de búsqueda. La mutación propuesta consiste en la selección aleatoria de dos componentes de un individuo e intercambiarlos entre sí. Un ejemplo se muestra en la Fig. 3.4, en el cual el segundo componente del individuo es intercambiado con el penúltimo.

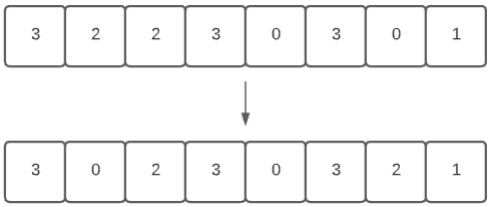


Figura 3.4: Ilustración de la mutación.

Cabe destacar que la mutación implementada mantiene la factibilidad en cuanto al número mínimo de hubs habilitados de cada tipo. Sin embargo, el presupuesto puede excederse. En este caso, se explora otro intercambio entre componentes para realizar la mutación. El proceso se detiene cuando se logra una mutación exitosa. En el caso en que se hayan explorado todas las posibles intercambios y no se obtenga un individuo factible, la mutación no se aplica y el individuo se incluye en la nueva población.

3.2.8 Población Actualizada

En este punto, tenemos la población original de individuos y la población de los recién creados. Ambas poblaciones se fusionan de manera elitista. Es decir, los individuos con un mejor valor de aptitud se incluyen en la población que reemplaza a la anterior.

Al hacer esto, mantenemos a los mejores individuos que ya han sido creados e incluimos a los recién creados para mejorar la aptitud de la población. A pesar de este esquema elitista, el proceso evolutivo no está sesgado hacia óptimos locales, ya que la diversidad se proporciona mediante la fase de mutación.

El proceso evolutivo se repite hasta que se alcanza un número predefinido de generaciones, que es el criterio de detención considerado en el NEA propuesto.

3.3 Resultados

Dado que no existen estudios previos relacionados con el problema considerado, no existen instancias de referencia disponibles. Por lo tanto, para evaluar el rendimiento del algoritmo NEA propuesto, se ha desarrollado un generador de instancias aleatorias pero controladas para crear dos conjuntos principales de diferentes instancias de prueba. Los tamaños de las instancias se muestran en la Tabla 3.1, esto es, 50 posibles sitios y 50 usuarios, y 75 posibles sitios y 100 usuarios. Cada conjunto se puede dividir en dos subconjuntos, en los cuales se varía la conectividad mínima de la red y se deben habilitar diferentes números de hubs diferentes. Se crean un total de 60 instancias para llevar a cabo la experimentación computacional.

Sitios Potenciales	Usuarios	Tipos de Hubs	p_{min}	p_j
50	50	3	0.5	[5, 3, 2]
				[2, 3, 5]
				[2, 6, 2]
			0.75	[5, 3, 2]
				[2, 3, 5]
				[2, 6, 2]
75	100	3	0.5	[5, 5, 5]
				[7, 5, 3]
				[4, 4, 7]
			0.75	[5, 5, 5]
				[7, 5, 3]
				[4, 4, 7]

Tabla 3.1: Características de las instancias

3.3.1 Parámetros del algoritmo NEA

Con el fin de abordar la naturaleza estocástica del algoritmo propuesto, se llevan a cabo múltiples ejecuciones del NEA para la misma instancia. Además, el NEA involucra diferentes parámetros, tales como el tamaño de la población, el número de torneos en el cruce, la probabilidad de ingresar al cruce y la probabilidad de ingresar a la mutación. Asimismo, los criterios de detención involucran dos parámetros, es decir, el número total de generaciones y el número de generaciones consecutivas sin mejorar al individuo actual.

Aunque existen estudios que indican que los algoritmos evolutivos convergen a pesar de la calibración de los parámetros [319], decidimos investigar el rendimiento del NEA bajo diferentes configuraciones de parámetros.

Se realizaron extensas pruebas preliminares para identificar los parámetros más adecuados del NEA al resolver el problema aquí estudiado. Por lo tanto, fijamos el tamaño de la población en 100 individuos. La selección de los individuos se lleva a cabo después de completar 5 torneos. La mitad de los individuos mejor clasificados ingresan al cruce con una probabilidad del 0.65. Luego, los descendientes entran en la fase de mutación con una probabilidad del 0.2. El número máximo de generaciones se establece en 100. Sin embargo, si la aptitud del individuo actual no mejora en 10 generaciones consecutivas, el NEA se detiene.

Los resultados preliminares indicaron que el valor de los parámetros involucrados en el NEA desempeña un papel clave en la eficiencia del algoritmo propuesto. Cabe mencionar que los parámetros seleccionados se encuentran dentro del rango indicado en las conclusiones encontradas en [319], lo que indica que podrían funcionar adecuadamente.

3.3.2 Resultados Computacionales

El NEA está implementado en el lenguaje de programación FICO Xpress Mosel. Además, el problema de nivel inferior se resuelve utilizando el Solver FICO Xpress 8.9. Toda la experimentación computacional se lleva a cabo en una estación de trabajo que utiliza un procesador Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz con 16 GB de RAM, bajo el sistema operativo Windows 11 Professional. Como se mencionó anteriormente, debido a la estocasticidad involucrada en el NEA, se realizan 10 ejecuciones para cada instancia.

Los resultados obtenidos de la experimentación computacional se muestran en las Tablas 3.2 y 3.3. La etiqueta $ww \times xxx.yy.zz$ de cada instancia indica el número de sitios potenciales (ww), el número de usuarios (xxx), el conjunto de centros de cada tipo que

deben habilitarse (yy) y el número de la instancia con estas características específicas (zz). En la columna etiquetada como F_{mejor}, se informa el mejor valor obtenido para la función objetivo de nivel superior. Para medir la dispersión de los valores obtenidos, se miden lo siguiente: el promedio (F_{prom}) y el peor (F_{peor}) valor de la función objetivo. El número de ejecuciones en las que el NEA alcanzó el mejor valor objetivo se incluye en # Best. Luego, el tiempo promedio requerido (en segundos) se incluye en la columna Tiempo$_{prom}$. Además, el número de centros habilitados adicionalmente en comparación con el mínimo requerido se muestra en las tres columnas siguientes. Por ejemplo, en $Extra_{min}$ se indica el mayor número de centros adicionales habilitados en esa instancia particular. De manera correspondiente, $Extra_{prom}$ y $Extra_{max}$ contienen el número promedio y máximo de centros adicionales habilitados. Finalmente, se muestra en la columna Conectados el número total de usuarios conectados.

Instance	F_{mejor}	F_{prom}	F_{peor}	# Best	Tiempo$_{prom}$	Extra$_{min}$	Extra$_{prom}$	Extra$_{max}$	Conectados
50×50.B1.01	0	0.2	1	8	41.989	0	0.1	1	25
50×50.B1.02	0	0.1	1	9	41.904	0	0.2	1	25
50×50.B1.03	0	0.3	1	7	48.398	0	0.5	1	25
50×50.B1.04	0	0	0	10	43.643	0	0	0	25
50×50.B1.05	0	0.2	1	8	32.085	0	0	0	25
50×50.B1.06	0	0.3	1	7	41.264	0	0.3	1	38
50×50.B1.07	0	0.1	1	9	45.195	0	0.1	1	38
50×50.B1.08	0	0.2	2	9	52.362	0	0.3	1	38
50×50.B1.09	1	1	1	10	44.962	0	0.2	1	38
50×50.B1.10	1	1.2	2	8	55.133	0	0	0	38
50×50.B2.01	1	1.1	2	9	46.855	0	0.4	1	25
50×50.B2.02	1	1.3	2	7	70.895	0	0.1	1	25
50×50.B2.03	1	1.1	2	9	58.995	0	0.5	1	25
50×50.B2.04	1	1.2	2	8	45.21	0	0.1	1	25
50×50.B2.05	1	1.4	2	6	48.117	0	0	0	25
50×50.B2.06	1	1.6	2	4	43.755	0	0.1	1	38
50×50.B2.07	0	0.3	2	8	64.05	0	0.2	1	38
50×50.B2.08	1	1.3	3	8	64.119	0	0.4	1	38
50×50.B2.09	1	1.4	3	7	52.125	0	0	0	38
50×50.B2.10	1	1.3	3	8	43.506	0	0.1	1	38
50×50.B3.01	1	1	1	10	50.11	0	0.5	1	25
50×50.B3.02	1	1	1	10	40.364	0	0	0	25
50×50.B3.03	0	0.3	2	8	47.084	0	0.6	1	25
50×50.B3.04	1	1.2	2	8	41.369	0	0.1	1	25
50×50.B3.05	0	0	0	10	47.321	0	0.1	1	25
50×50.B3.06	1	1.4	2	6	52.724	0	0.2	1	38
50×50.B3.07	0	0.4	2	7	49.168	0	0	0	38
50×50.B3.08	1	1.1	2	9	51.313	0	0.2	1	38
50×50.B3.09	1	1.3	3	8	44.56	0	0.1	1	38
50×50.B3.10	1	1.5	3	7	64.153	0	0	0	38

Tabla 3.2: Resultados para las instancias de tamaño 50×50

3.3.3 Discusión

En la Tabla 3.2 se muestran los resultados para el primer conjunto de instancias, es decir, el conjunto que contiene 50 hubs potenciales y 50 usuarios. Se puede observar que solo en 12 instancias se logra un equilibrio perfecto de usuarios. Sin embargo, en el resto de las instancias se obtiene un valor de 1, lo que indica que la mayor diferencia de usuarios conectados a centros corresponde a un usuario. El peor equilibrio es 3, y se obtiene en las instancias 08, 09 y 10 del tipo B2, y 09 y 10 del tipo B3. Coincidentemente, todas estas instancias consideran un p_{min}=0.75, lo que implica que al menos 38 de los 50 usuarios deben estar conectados. Para las instancias en las que se logra un equilibrio perfecto, el número de ejecuciones que lograron ese valor es mayor a 5. Por ejemplo, en las instancias 07 del tipo B1 y las instancias 05 y 07 del tipo B3, se informa el mayor número de equilibrios perfectos (con 9, 6 y 6, respectivamente).

Otro aspecto conveniente de la experimentación numérica es que para las instancias 09 del tipo B1, y las instancias 01 y 02 del tipo B3, el mejor equilibrio se obtiene en todas las diez ejecuciones.

En cuanto al tiempo computacional requerido para cada instancia, es inferior a 1 minuto en casi todas ellas. Solo las instancias 02, 07 y 08 del tipo B2, y la instancia 10 del tipo B3 presentaron tiempos de ejecución promedio mayores a un minuto, es decir, reportaron tiempos entre 64.05 y 70.895 segundos.

Es importante enfatizar que para las 30 instancias se habilita el número mínimo de hubs. Sin embargo, en el peor caso, para 22 de las 30 instancias se habilita un hub adicional además del indicado en el vector p_j. De manera similar, el número de usuarios conectados es el mínimo requerido, es decir, 25 y 38 para p_{min}=0.5 y p_{min}=0.75, respectivamente. Por lo tanto, sería interesante realizar un análisis para evaluar la conveniencia de habilitar más hubs o conectar más usuarios.

Los resultados obtenidos para las instancias de gran tamaño se muestran en la Tabla 3.3. En general, se logra un equilibrio perfecto en más de la mitad de las 30 instancias. Sin embargo, dado que el tamaño de la instancia aumentó, el *desbalance* también aumentó a un máximo de dos. Como era de esperar, el tiempo computacional requerido aumentó. Ahora, se tarda entre 1 y 2 minutos, en promedio, en resolver cada instancia. En este caso, las instancias 75×100.B2.07 y 75×100.B1.10 son las que tienen el tiempo mínimo y máximo, es decir, 71.5 y 127.0 segundos, respectivamente. De manera similar a las instancias de tamaño mediano, en la mayoría de las ejecuciones se habilita el número mínimo de hubs. En promedio, solo en 20 de las 30 instancias se necesita un centro adicional. Además, el número mínimo de usuarios siempre está conectado, es decir, no se obtiene ningún

beneficio para el tomador de decisiones de nivel superior al conectar un usuario adicional. Este es un hallazgo interesante que puede ser analizado en el problema de nivel superior.

Instance	F_{mejor}	F_{prom}	F_{peor}	# Best	Tiempo	Extra$_{min}$	Extra$_{avg}$	Extra$_{max}$	Connected
75×100.B1.01	2	2.2	3	8	121.443	0	0.7	1	50
75×100.B1.02	1	1.3	3	8	97.68	0	0.4	1	50
75×100.B1.03	1	1.2	3	9	93.981	0	0.6	2	50
75×100.B1.04	2	2.3	3	7	106.294	0	0.6	1	50
75×100.B1.05	2	2.1	3	9	101.687	0	0.3	1	50
75×100.B1.06	2	2.2	3	8	93.785	0	0.6	2	75
75×100.B1.07	0	0.6	3	7	84.878	0	0.1	1	75
75×100.B1.08	2	2.1	3	9	98.925	0	0.3	1	75
75×100.B1.09	2	2.4	4	7	98.58	0	0.4	2	75
75×100.B1.10	2	2.2	3	8	127.042	0	0.7	2	75
75×100.B2.01	2	2.1	3	9	106.787	0	0.6	2	50
75×100.B2.02	1	1.4	3	7	87.581	0	0	0	50
75×100.B2.03	2	2	2	10	88.137	0	0.4	1	50
75×100.B2.04	1	1.3	3	8	103.928	0	1.2	2	50
75×100.B2.05	2	2.1	3	9	114.68	0	0.6	1	50
75×100.B2.06	1	1.3	3	8	96.151	0	0.3	1	75
75×100.B2.07	1	1.2	3	9	71.526	0	0	0	75
75×100.B2.08	2	2.3	3	7	86.256	0	0.3	1	75
75×100.B2.09	2	2	2	10	97.411	0	0.5	1	75
75×100.B2.10	1	1.4	3	8	88.809	0	0.4	1	75
75×100.B3.01	2	2.2	3	8	105.271	0	0.3	1	50
75×100.B3.02	2	2.1	3	9	89.339	0	0.4	1	50
75×100.B3.03	2	2	2	10	106.922	0	0.1	1	50
75×100.B3.04	2	2.2	3	8	96.9	0	0.5	2	50
75×100.B3.05	2	2.1	3	9	98.896	0	0.2	1	50
75×100.B3.06	2	2.4	3	6	109.415	0	0.5	1	75
75×100.B3.07	2	2.3	3	7	83.265	0	0.2	1	75
75×100.B3.08	2	2.2	3	8	87.392	0	0.4	1	75
75×100.B3.09	2	2.7	5	7	122.905	0	1	2	75
75×100.B3.10	2	2.1	4	9	100.25	0	0.4	1	75

Tabla 3.3: Resultados para las instancias de tamaño 75×100

Durante la experimentación computacional se consideraron diferentes valores de p_j. En otras palabras, se exploró el impacto de tener más hubs de tipo pequeño que centros de tipo grande. Esto se puede observar en la Tabla 3.1, en la cual se muestran tres configuraciones diferentes para cada tamaño de las instancias.

Por ejemplo, habilitar más hubs de tipo pequeño no garantiza obtener un equilibrio perfecto. Se puede observar en las tablas 3.2 y 3.3 que habilitar hubs de tipo mediano y grande ofrece más opciones de conexión a los usuarios, lo que resulta en una red más equilibrada. Esto prevalece a pesar de que los hubs de tipo mediano y grande son más costosos que los de tipo pequeño. Particularmente, observa las instancias 50×50.B1.07 y 50×50.B3.07, ambas lograron un equilibrio perfecto de usuarios, a pesar de tener habilitados diferentes tamaños de hubs. Otro caso notable corresponde a las instancias que logra-

ron un equilibrio casi perfecto. Por ejemplo, las instancias 75×100.B1.02 y 75×100.B2.02. Esta es la razón por la cual se afirma que el uso de diferentes tipos de hubs conduce no solo a un equilibrio perfecto o casi perfecto, sino que también tiene en cuenta las conexiones finales de los usuarios.

Además, en muchas instancias se logra un equilibrio perfecto cuando se habilitan hubs adicionales, es decir, se utilizan más hubs de los mínimos requeridos en la red. Esto podría deberse a la cantidad de alternativas que se ofrecen a los usuarios, quienes eligen la mejor opción para ellos.

Dada esta situación en la que existe una clara jerarquía entre los tomadores de decisiones, se aplica un enfoque de programación bi-nivel. Como resultado de esto, se propone un novedoso modelo bi-nivel para estudiar esta interesante situación.

Basándose en las características del problema y debido a la falta de software especializado en programación bi-nivel, se propone un algoritmo evolutivo anidado para resolver el problema. Este algoritmo mantiene un equilibrio entre la diversificación y la intensificación, ya que involucra componentes elitistas y estocásticos durante la evolución. Para evaluar la eficiencia del algoritmo propuesto, se crea un conjunto grande de diferentes instancias.

Los resultados obtenidos muestran la conveniencia de utilizar el algoritmo propuesto para aproximar soluciones de buena calidad. El algoritmo resuelve el problema en un tiempo relativamente corto, incluso para instancias grandes. Las desviaciones obtenidas son pequeñas, lo que implica que el algoritmo evolutivo propuesto es estable y robusto.

Capítulo 4

Un algoritmo co-evolutivo para el problema binivel de localización del árbol de hubs con precios fijos.

Actualmente, en el mundo en el que vivimos, damos por sentadas muchas cosas, como el poder viajar, el internet, el transporte, etc. pero para lograr esto la logistica detrás de las redes de transporte y comunicación es muy grande.Uno de los pilares de estas redes es el uso de centros de distribución llamados hubs, estos hubs se encargan de reunir productos, personas, etc, y redirigirlos a su destino. Sin embargo, la ubicación óptima de estos es crucial para un buen uso de la red. Además, en la creación y uso de estas redes existen dos agentes que están involucrados. Por un lado, tenemos el agente que se encarga de la creación de la red y, por el otro lado, el agente que usa esta red. Es claro que por lo general los objetivos de estas dos personas son distintos. El primer agente puede buscar maximizar la ganancia, mientras que por otro lado el segundo agente quiere usar la red a costo mínimo. Todos estos factores hacen que la programación binivel sea una herramienta útil para modelar esta situación, sin embargo, debido a la dificultad de estos problemas, el desarrollo de algoritmos para resolverlo es muy importante.

En este capitulo, mostraremos un nuevo enfoque para modelar y resolver un problema de localización de hubs con precios a través de la programación binivel. Además se mostrará un novedoso enfoque para resolverlo a través de un algoritmo co-evolutivo. La principal novedad de nuestro enfoque es la manera en como las poblaciones co-evolucionan se considedran. Se adaptaran tres conocidos datasets para resolver estos problemas, se analizarán resultados y se darán futuras recomendaciones para estos problemas.

El resto del capitulo se divide de la siguiente manera: En la sección 4.1 se dará un pequeño vistazo en desarrollo de los problemas de localización de hubs. En la sección 4.2

se presenta el problema y el modelo binivel propuesto. Por otro lado en la sección 4.3 se detallará a fondo el algoritmo co-evolutivo propuesto. Por último, la sección 4.4.2 se enfoca en la experimentación computacional donde los resultados obtenidos se resumen, analizan y se discuten.

4.1 Antecedentes:Problemas de Localización de Hubs

La problemas de localización de hubs (HLP por sus siglas en inglés) es un área activa de investigación, impulsada por sus aplicaciones prácticas y los desafíos matemáticos que plantea. El problema principal implica determinar las posiciones óptimas de nodos llamados hubs y los arcos que los conectan, con el objetivo de minimizar tanto los costos de configuración como los costos de enrutamiento para un conjunto de commodities. Los HLP fueron introducidos por primera vez en trabajos seminales como [262, 263]. A lo largo de los años, han surgido varias variantes del problema, incluyendo capacitados [102, 243, 285], no capacitados (costos fijos) [99, 345, 349], asignación simple y múltiple [46, 119, 249], y objetivos de cobertura [268], así como aquellos que consideran incertidumbre [100], entre otros.

Una de estas variantes, conocida como problema de localización del árbol de hubs (THLP por sus siglas en inglés) fue introducida en [97, 98]. El THLP implica determinar las posiciones de un número especificado (p) de hubs entre un conjunto dado de nodos, junto con los enlaces de hub asociados, para formar una red de hubs en forma de grafo de árbol. El objetivo es minimizar los costos de envío total para un conjunto predefinido de commodities. Esta configuración de red en forma de árbol ha demostrado su utilidad en varios contextos de diseño de redes de hubs [256, 327]. Aunque el THLP fue inicialmente introducido como un problema de localización de hubs con asignación única, adoptamos una versión de asignación múltiple del problema, que es más realista en aplicaciones prácticas.

El THLP ha ganado una creciente atención en la última década, lo que ha llevado al desarrollo de diversos enfoques de solución. En [234], los autores introducen un novedoso enfoque que combina descomposición de Benders con ramificación y corte para resolver el problema. Otro enfoque que utiliza un algoritmo genético es propuesto en [271]. Además, se han explorado varias variantes del THLP. En [275], en lugar de minimizar la suma total de los costos de envío, se propone un enfoque de agregación generalizada basado en diferentes operadores. En [45], se permite que ciertos nodos de hub sean mejorados, lo

que resulta en una reducción de los costos de transporte para las rutas que utilizan esos hubs. Además, [172] analiza una versión de asignación múltiple del THLP.

En estos estudios, el objetivo es diseñar una red de hubs que maximice la ganancia derivada del uso de la red. En consecuencia, los parámetros de entrada deben incluir los precios pagados por los usuarios por utilizar la red de hubs. Estos precios influyen significativamente en el uso de la red y, por lo tanto, en su rentabilidad. Tanto precios altos como bajos pueden afectar la ganancia del tomador de decisiones. Balancear los precios se vuelve crucial para optimizar tanto la satisfacción del usuario como las ganancias del administrador de la red. Por ejemplo, [213] aborda un problema de ubicación de hubs con precios, donde nuevas empresas deben determinar cómo gestionar y extender una red de hubs incompleta. Proporcionan una formulación de programación lineal entera mixta y desarrollan un algoritmo genético para resolver el problema. De manera similar, en [120], los autores abordan un problema de ubicación de hubs con precios, modelándolo como un problema de programación binivel no lineal. Linealizan el modelo y lo mejoran mediante reducciones de variables y desigualdades válidas.

Es claro que el desarrollo de esta área es importante para el continuo mejoramiento de las redes de transporte y comunicación a través de la incorporación de la localización de hubs y precios para estos problemas, además que incorporar nuevas variantes nos da un abanico de posibilidades para distintos contextos que nos podamos encontrar.

4.2 Modelo Matemático

Esta sección está dedicada a introducir el problema bajo análisis y establecer la notación utilizada a lo largo de este capitulo. Además, presentamos la formulación matemática de programación binivel propuesta para el problema.

Sea $G = (I, E)$ una red, donde $I = 1, \ldots, m$ es el conjunto de nodos hub potenciales y E es el conjunto de enlaces entre ellos (los arcos interhub potenciales). Para cada $e = (i, j) \in E$, denotamos por f_{ij} el costo de activar el arco en la red troncal, siempre y cuando los nodos hub i y j también estén activados. Sin pérdida de generalidad, asumimos que G es una red completa, ya que cualquier conexión faltante puede ser representada fijando altos costos de activación para aquellos arcos que no están disponibles en la red.

La demanda de servicio está representada por un conjunto de commodities definidas sobre pares de usuarios en un conjunto J, indexadas por un conjunto C. Sea $\mathcal{C} = a = (o_a, d_a, w_a) : a \in C$ el conjunto de commodities, donde la terna (o_a, d_a, w_a) indica que una cantidad de flujo $w_a > 0$ debe ser enviada desde un origen $o_a \in J$ hasta el

destino $d_a \in J$ a través del conjunto de arcos activados en E. Se asume que activar un arco $e = i, j$ en E permite el uso bidireccional del arco i, j, es decir, los arcos (i, j) y (j, i) pueden ser utilizados. El conjunto A denota los arcos inducidos por el conjunto de arcos E, definido como $A = (i, j), (j, i) : i, j \in E$.

Una vez que la red está establecida, los commodities pueden ser enviados a través de la red sin restricciones en el uso de uno o más arcos. Así, las rutas para los commodities tienen la forma $o_a \to k_1 \to k_2 \to \cdots \to k_{\ell(a)} \to d_a$, donde $k_1, \ldots, k_{\ell(a)}$ son hubs activos en la red, y $k_1, k_2, k_2, k_3, \ldots, k_{\ell(a)-1}, k_{\ell(a)}$ son arcos activados entre hubs.

Como se mencionó anteriormente, dos agentes diferentes están involucrados simultáneamente en la toma de decisiones para el BTHLPwP. Por un lado, el líder decide qué nodos y enlaces en G deben ser activados resultando en una red de hubs en forma de árbol y además los precios para utilizar la red construida. Cada commodity $a \in \mathcal{C}$ que atraviesa la red a través del arco (i, j) paga al líder una cantidad proporcional al flujo w_a, denominada $w_a \pi_{ij}$ (donde π_{ij} es el precio pagado por unidad de flujo para atravesar el arco). Además, el mantenimiento de la red incurre en costos. Denotamos por g_i el costo unitario incurrido por el líder cuando el commodity a utiliza la red al ingresar al nodo $i \in I$. El objetivo del líder es maximizar el beneficio general, que consiste en los ingresos totales menos los costos incurridos por todas las commodities que utilizan la red. En contraste, una vez que el líder ha determinado la estructura de la red hub, el seguidor envía los commodities a costo mínimo.

En nuestro problema de programación matemática, las decisiones del líder se modelan mediante los siguientes conjuntos de variables, representando sus tres principales decisiones: activación de hubs, activación de enlaces entre hubs y fijación de precios.

- $\pi_{i\ell} \geq 0$: Precio por usar el arco $\{i, \ell\}$, for $i, \ell \in I$.
- $y_i = \begin{cases} 1 & \text{si el hub } i \text{ se abre.} \\ 0 & \text{si no} \end{cases}, \forall i \in I.$
- $z_{i\ell} = \begin{cases} 1 & \text{si el arco } (i, \ell) \text{ entre hubs es parte de la estructura de red} \\ 0 & \text{si no} \end{cases}, \forall i, \ell \in I.$

Por otro lado, dado la red de hubs y los precios determinados por el líder, el seguidor (o los seguidores) decide(n) sobre la utilización de la red de hubs. Un commodity $a \in \mathcal{C}$ puede optar por ser enviado a través de la red de hubs a través de los arcos de la red. Denotamos por $d_{o_a i}$ el costo unitario de recolección para el primer nodo donde a entra a la red hub, y por $d_{i d_a}$ el costo unitario de distribución desde el último nodo hub en la

ruta hasta el destino de a. Asumimos que el costo unitario de transporte para a es π_{ij} (el precio decidido por el líder para usar la red). Si el seguidor decide enviar el commodity a directamente (o usando otro modo de transporte) entre su origen y destino sin usar la red de hubs, se incurre en un costo de c_a. Así, el objetivo del seguidor es determinar cómo enviar los commodities para minimizar la suma de los costos de envío, ya sea utilizando la red hub establecida por el líder o enviando directamente el commodity sin usar la red de hubs. En el primer caso, el uso de la red implica un beneficio para el líder, mientras que en el segundo caso, el líder no recibe ninguna recompensa. Cabe destacar que dado que diferentes commodities están involucradas en las decisiones del seguidor, cada uno de ellos decide de manera independiente cómo ser enviada.

Las distintas decisiones del seguidor de nuestro problema de optimización son modeladas con el siguiente conjunto de variables:

- $x_{ai\ell} = \begin{cases} 1 & \text{si el commodity } a \text{ is es enviado a través del arco } (i,\ell) \text{ de la red.} \\ 0 & \text{si no} \end{cases}$
- $x^1_{ai} = \begin{cases} 1 & \text{si el commodity } a \text{ se envía inicialmente a través del hub } i \text{ de la red.} \\ 0 & \text{otherwise} \end{cases}$
- $x^2_{ai} = \begin{cases} 1 & \text{si el commodity } a \text{ se envía al final por el hub } i \text{ de la red} \\ 0 & \text{si no} \end{cases}$
- $q_a = \begin{cases} 1 & \text{si el commodity } a \text{ es enviado directamente entre los clientes} \\ 0 & \text{otherwise} \end{cases}$

Utilizando los conjuntos de variables y parámetros mencionados anteriormente, proponemos el siguiente programa binivel para modelar el BTHLPwP:

$$\text{máx} \quad \sum_{i,j\in I} \pi_{ij} \sum_{a\in\mathcal{C}} w_a x_{aij} - \sum_{a\in\mathcal{C}} \sum_{i\in I} w_a g_i x^1_{ai} - \sum_{i,j\in I} f_{ij} z_{ij} \tag{4.1}$$

$$\text{s.t.} \quad \sum_{i\in I} y_i = p, \tag{4.2}$$

$$\sum_{i,j\in I} z_{ij} = p - 1, \tag{4.3}$$

$$z_{ij} \le y_i, \qquad \forall i,j \in I \tag{4.4}$$

$$z_{ij} \le y_j, \qquad \forall i,j \in I \tag{4.5}$$

$$\sum_{j\in I\setminus S} z_{ij} \ge 1 \qquad \forall S \subseteq I, i \in S \tag{4.6}$$

$$y_i, z_{ij} \in \{0,1\}, \pi_{ij} \geq 0, \qquad \forall i,j \in I \tag{4.7}$$

donde para unas variables y_i, z_{ij}, π_{ij} las variables x^1_{ai}, x^2_{aj}, x_{aij} y q_a deben resolver el siguiente problema de optimización:

$$\text{mín} \quad \sum_{a\in\mathcal{C}} w_a\Big(\sum_{i\in I}(d_{o_a i}x^1_{ai} + d_{id_a}x^2_{ai}) + \sum_{i,j\in I}\pi_{ij}x_{aij} + c_a q_a\Big) \tag{4.8}$$

$$\text{s.t.} \quad \sum_{i\in I} x^1_{ai} + q_a = 1, \qquad \forall a \in \mathcal{C}, \tag{4.9}$$

$$\sum_{i\in I} x^2_{ai} + q_a = 1, \qquad \forall a \in \mathcal{C}, \tag{4.10}$$

$$\sum_{j\in I} x_{aij} + x^1_{ai} - \sum_{j\in I} x_{aji} - x^2_{ai} = 0, \qquad \forall a \in \mathcal{C}, i \in I, \tag{4.11}$$

$$x^1_{ai} \leq y_i, \qquad \forall a \in \mathcal{C}, i \in I, \tag{4.12}$$

$$x^2_{ai} \leq y_i, \qquad \forall a \in \mathcal{C}, i \in I, \tag{4.13}$$

$$x_{aij} \leq z_{ij}, \qquad \forall a \in \mathcal{C}, i,j \in I, \tag{4.14}$$

$$x_{aij}, x^1_{ai}, x^2_{ai}, q_a \in \{0,1\}, \qquad \forall a \in \mathcal{C}, i,j \in I. \tag{4.15}$$

La función objetivo del líder (4.1) maximiza el beneficio total del uso de la red (ingresos menos costos). Las restricciones (4.2) y (4.3) aseguran que exactamente p hubs y $p-1$ arcos entre hubs sean activados. Las restricciones (4.4) y (4.5) impiden la activación de arcos de hubs entre hubs no abiertos. La restriccion (4.6) impide la creación de subtours en la red, las cual, junto con las restricciones anteriores, aseguran que la red de hubs resultante sea un árbol.

Las restricciones (4.8) a (4.15) definen el problema del seguidor. La función objetivo (4.8) representa el costo de envio a minimizar basado en las decisiones del líder. Las restricciones (4.9) y (4.10) aseguran que se utilice un nodo hub único para el envio inicial y final de los commodities, o que la commodity sea enrutada directamente sin usar la red hub. Las restricciones (4.11) representan las restricciones de conservación del flujo, asegurando que cada commodity sea enviado adecuadamente si elige usar la red de hubs. Finalmente, las restricciones (4.12), (4.13) y (4.14) previenen el envio de commodities a través de hubs o arcos no activados.

Una vez que la red de hubs (las variables y y z) y los precios (las variables π) son conocidos, el problema del seguidor se simplifica a un problema de flujo de costo mínimo con flujos unitarios. Este problema puede resolverse eficientemente (en tiempo polinomial)

al separarlo para cada commodity y resolver $|\mathcal{C}|$ problemas de camino más corto usando el algoritmo de Dijkstra. Si el costo del camino más corto para una commodity a es menor que c_a, la commodity es enrutada a través de la red hub; de lo contrario, la commodity es enrutada directamente (o usando otro modo de transporte), es decir, $q_a = 1$. Es importante determinar una postura si se obtienen múltiples caminos más cortos para una commodity específica dada la decisión del líder. En este caso, se asume el enfoque optimista [168], [114]. Es decir, se asume que el camino más corto que produce el mayor beneficio para el líder es la decisión del seguidor.

Aunque se puede desarrollar una reformulación equivalente de un solo nivel para el modelo de optimización utilizando las condiciones de optimalidad estándar de KKT, resolver el problema de manera exacta plantea un desafío computacional significativo. La presencia de desigualdades no lineales, junto con la necesidad de considerar desigualdades de cortes (4.6), impide el uso de solvers comerciales para resolver instancias del mundo real. Aunque es posible diseñar un enfoque de relajación y fijación para evitar la incorporación de un número exponencial de restricciones e incluirlas según sea necesario dentro de un enfoque de corte y ramificación, la cantidad de cortes e iteraciones es computacionalmente prohibitiva incluso para instancias de tamaño pequeño. Además, la reformulación de un solo nivel requiere incorporar constantes grandes M para linealizar algunas de las expresiones no lineales (y no convexas). Como se observa en diferentes trabajos [177, 176, 272], determinar las constantes grandes M apropiadas es tan desafiante como resolver el problema binivel en sí mismo.

Por lo tanto, dadas las limitaciones prácticas de la reformulación mencionada anteriormente, en la siguiente sección nos enfocamos en proponer un algoritmo metaheurístico novedoso para obtener soluciones de alta calidad dentro de un tiempo computacional razonable.

4.3 Un Algoritmo Co-evolutivo para el BTHLPwP

En esta sección se presentará el algoritmo co-evolutivo propuesto para resolver el problema. Los algoritmos co-evolutivos (Co-EAs por sus siglas en inglés) son una clase particular de algoritmos evolutivos (EA's, por sus siglas en inglés) usados para resolver problemas de optimización de manera heurística inspirados por la evolución de las especies.

En los algoritmos evolutivos, una población de individuos es identificada basada en el problema y pasan a través de operadores genéticos que buscan mejorar la aptitud de estos individuos. Estos operadores consisten de una cruza, donde los individuos son combinados para generar descendencia y una mutación, donde la descendencia pasa a través

de pequeños cambios para agregar diversidad a la población. Un criterio de supervivencia se aplica y el proceso se itera durante un cierto número de generaciones. El objetivo de los operadores en los EA's es encontrar soluciones que mejoren la aptitud de la población mientras se mantiene la diversidad entre los individuos. Los EA's han demostrado grandes ventajas en resolver de manera eficiente problemas que son costosos de resolver. [320], [103], [387].

En los Co-EAs, se consideran simultáneamente dos o más poblaciones, y la evolución se logra a través de interacciones entre individuos de diferentes poblaciones. Existen dos tipos principales de Co-EAs según la interacción entre los individuos: cooperativos y competitivos. En los Co-EAs cooperativos, ambas poblaciones de individuos evolucionan intercambiando información para facilitar su mejora, mientras que en los Co-EAs competitivos, los individuos de una población evolucionan a expensas de la otra.

Los Co-EAs han demostrado ser exitosos en la resolución de problemas de programación binivel, lo cual se atribuye a la división de las variables de decisión entre el líder y el seguidor [260], [64], [296]. En tales problemas, se crea una población para cada nivel de toma de decisiones (líder y seguidor), y ambas poblaciones evolucionan mientras intercambian información entre ellas. Sin embargo, el algoritmo que proponemos no se adhiere a este enfoque convencional. En su lugar, las dos poblaciones consideradas están asociadas con las decisiones del líder, y el problema del seguidor se resuelve óptimamente cuando es necesario. Por lo tanto, la co-evolución entre poblaciones ocurre únicamente en el espacio de decisión del líder.

Como se mencionó anteriormente, las decisiones del líder abarcan dos aspectos: (1) la construcción de la red de hubs bajo una topología en forma de árbol, y (2) la determinación de los precios unitarios del flujo para utilizar la red. Dada su estrecha relación, cada decisión se asigna a su propia población. La primera población consiste en árboles formados por los hubs seleccionados, mientras que la segunda comprende vectores que representan los precios impuestos por el uso de la red. Se facilita un intercambio asincrónico de información entre estas poblaciones.

Para detallar, inicialmente se genera una población aleatoria de árboles. Posteriormente, con base en los árboles generados, se resuelve óptimamente el problema del seguidor, determinando así los límites inferiores y superiores aceptables para los precios óptimos. A continuación, se generan vectores de precios aleatorios dentro de los intervalos calculados, con los extremos representando los límites inferior y superior. Luego, se puede evaluar la aptitud de un individuo, permitiendo que ambas poblaciones evolucionen de manera independiente. Cuando se actualizan los precios, se desencadena la co-evolución

para revisar los intervalos aceptables para los precios. Este proceso iterativo continúa hasta que se cumple un criterio de parada predeterminado.

Un diagrama de flujo de algoritmo Co-EA propuesto se muestra en la Fig 4.1:

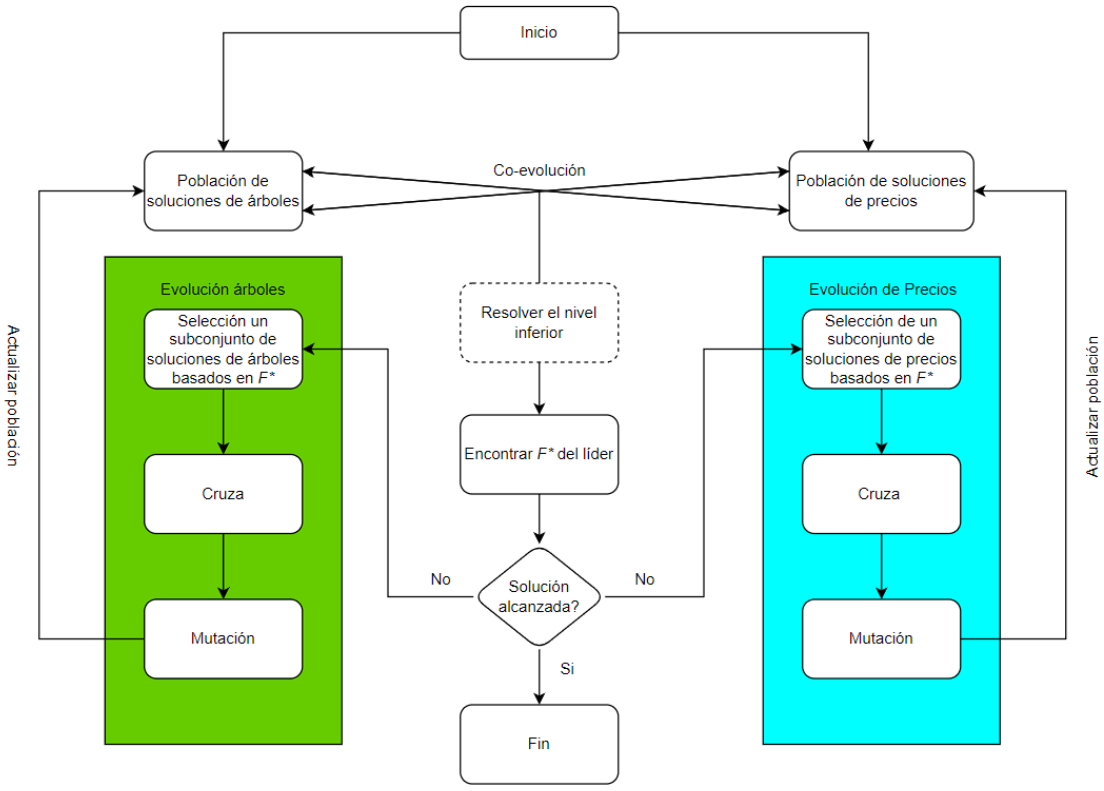


Figura 4.1: Diagrama de flujo del algoritmo propuesto.

A continuación se detallarán cada uno de los componentes involucrados en este algoritmo.

4.3.1 Construyendo soluciones binivel factibles

El algoritmo comienza creando dos poblaciones. La primera población se basa en la red de hubs, que está formada por los centros abiertos y los enlaces que conectan bajo una topología de árbol. La segunda población se basa en los precios que selecciona el líder. La combinación de soluciones en estas dos poblaciones dará lugar a una solución completa del líder, y se debe resolver el problema del seguidor para evaluar la aptitud. Cabe destacar que ambas poblaciones evolucionan de manera independiente, pero existe un intercambio de información controlado por el *operador co-evolutivo*.

4.3.2 Creación de redes estructuradas en forma de árbol.

Primero, se construyen redes de hubs en forma de árbol para subconjuntos seleccionados de centros. Estas decisiones están representadas por las variables de decisión y y z en nuestro modelo de programación matemática. Recordemos que dado un conjunto de sitios potenciales $|I|$, se deben ubicar exactamente p hubs. Esto se representa mediante un vector binario y de longitud $|I|$, donde 1 indica que el hub está ubicado, y 0 en caso contrario.

Basándose en los hubs ubicados, se construyen árboles aleatorios en estos nodos (z), codificados utilizando sus matrices de adyacencia. Cabe mencionar que estas matrices son simétricas, ya que el flujo enviado a través de los hubs es bidireccional. Una representación de esta solución parcial del líder se muestra en la Fig. 4.2.

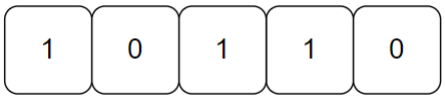


Figura 4.2: Ejemplo con 5 sitios potenciales y 3 hubs a abrir.

4.3.3 Creación de los precios

Para crear una población aleatoria de vectores de precios (π), se utilizan los árboles dados por las variables y y z. Dado que $\pi \geq 0$, el límite inferior es trivial. Por otro lado, se calcula un límite superior para cada arista en el árbol y luego se identifican los intervalos aceptables para generar los precios. La lógica detrás de esta idea es evitar desperdiciar esfuerzo estableciendo precios excesivos que desanimen a los clientes a usar la infraestructura de la red; es decir, las mercancías se enviarían directamente del origen al destino sin generar ningún beneficio para el líder. Una representación de estos precios es a través de un arreglo bidimensional como en la Figura 4.3.

0	2	5	4	3
2	0	8	4	1
5	8	0	5	9
4	4	5	0	4
3	1	9	4	0

Figura 4.3: Ejemplo de una solución de precios. Cada entrada representa el precio de usar un arco específico. Por ejemplo, si el seguidor usa el arco (1,2), entonces el precio asociado por usar ese arco es 2.

Dado que los precios no pueden ser mayores que el costo directo menos el costo de entrada y el costo de salida, fijamos esa diferencia como el límite superior. Si el precio excede ese valor, se desanima al seguidor de usar la red de hubs. Finalmente, generamos un vector de precios generando valores entre los límites inferior y superior.

4.3.4 Resolviendo el problema del seguidor

Para nuestro problema estamos empleando un esquema anidado en el Co-EA propuesto, lo que significa que el problema del seguidor debe resolverse de manera óptima para cada decisión del líder. Este marco garantiza que obtengamos soluciones factibles para el BTHLPwP.

Dado que el procedimiento para resolver el problema del seguidor se llamará repetidamente en el algoritmo Co-EA, es crucial implementar un método eficiente para lograrlo. Recordemos que el problema del seguidor está parametrizado por las variables del líder y, z y π. Una vez que estas variables están fijas, el problema del seguidor resulta en una serie de problemas de camino más corto (tantos como commodities tengamos), que pueden resolverse eficientemente mediante el conocido algoritmo de Dijkstra. Con esta estrategia, se obtienen valores factibles para las variables del seguidor x, x^1, x^2 y q

4.3.5 Evaluando la aptitud de los individuos

Una vez que se construye una población inicial de redes en forma de árbol (y, z) y una población de vectores de precios (π), se debe definir una estrategia de emparejamiento. Como se explicó anteriormente, la creación de un vector de precios está asociada con

un árbol específico. Por lo tanto, el criterio de emparejamiento es sencillo, y la tupla es (π, y, z). El problema del seguidor resultante para (π, y, z) se resuelve óptimamente para obtener $(x,\ x^1,\ x^2,\ q)$. Luego, se puede evaluar la función objetivo del líder $F(\pi, y, z)$, definida en la expresión (4.1). En otras palabras, se mide la aptitud del individuo (π, y, z).

4.3.6 Evolución de la población de redes en forma de árbol

Una solución parcial del líder, que consiste en la red en forma de árbol, evoluciona a través de un algoritmo evolutivo estándar (EA), el cual ha demostrado ser efectivo en la resolución de problemas similares [377]. En este proceso, los individuos con mejor aptitud tendrán más oportunidades de entrar en los operadores evolutivos para crear nuevos individuos. A continuación se describe este proceso:

- **Población inicial**: Los individuos se generan como se describió anteriormente.
- **Selección**: Basado en la aptitud de los individuos, se eligen pares de manera elitista y pseudoaleatoria.
- **Cruce**: Implementamos un método de cruce de dos puntos para generar descendencia. En este método, se seleccionan dos padres. Luego, se identifican aleatoriamente dos posiciones en el vector de solución de uno de los padres, y la descendencia hereda características de ambos padres. Una ilustración del cruce se muestra en la Figura 4.4.
- **Fase de reparación**: El operador de cruce puede llevar a individuos infactibles, los cuales se reparan en esta fase. La infactibilidad es causada por abrir un número diferente de los p hubs requeridos. Si el número de hubs abiertos es menor que p, abrimos aleatoriamente un nuevo hub. El proceso se repite hasta alcanzar exactamente p hubs abiertos. Por otro lado, si el número de hubs abiertos excede p, cerramos aleatoriamente un hubs. De manera similar, el proceso se repite hasta tener un individuo factible.
- **Mutación**: Para añadir diversidad al proceso evolutivo, se consideran nuevos árboles en el operador de mutación. Dado que el número de diferentes árboles de n nodos es n^{n-2}, usamos una nueva configuración de arcos conectando un árbol.

Una ilustración de los operadores evolutivos implementados para la evolución de la soluciones de redes en forma de árbol se muestra a continuación:

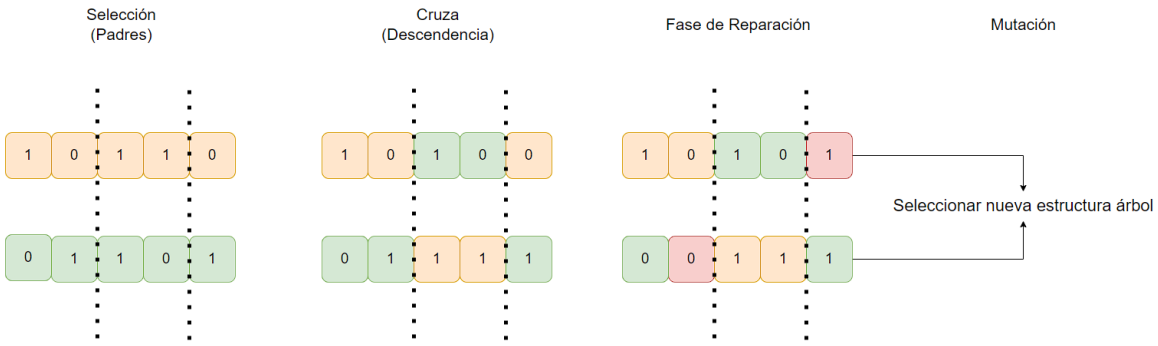


Figura 4.4: Ilustración de los operadores evolutivos usados en las redes de forma de árbol con $p = 3$.

4.3.7 Evolución de la población de los precios

De manera similar a lo anterior, seguimos un enfoque elitista, lo que significa que los individuos con mejor aptitud tendrán mejores oportunidades de sobrevivir y cruzarse. La descendencia heredará características de ambos padres y, potencialmente, puede someterse a una mutación para introducir diversidad.

Los detalles de este proceso se describen a continuación:

- **Población inicial**: Los individuos (vectores de precios) se generan de manera acotada pero aleatoria, siguiendo el procedimiento descrito anteriormente.
- **Selección**: La aptitud se mide de acuerdo con la función objetivo del líder. Se selecciona un número fijo de individuos de manera elitista.
- **Cruce**: Se emparejan individuos para realizar el cruce. Los pares se crean seleccionando un individuo del conjunto elitista y otro de la población inicial. Un par está formado por dos padres y generará dos nuevos individuos (descendencia). El cruce se realiza separando el vector de precios en cuadrantes y recombinándolos para crear la nueva descendencia.
- **Mutación**: En esta fase, se considera la descendencia generada. Separamos la descendencia en cuadrantes e identificamos el cuadrante con el tamaño más pequeño. Luego, aumentamos los valores de este cuadrante añadiendo un pequeño valor.

El proceso evolutivo de los vectores de precios se muestra en la Figura 4.5

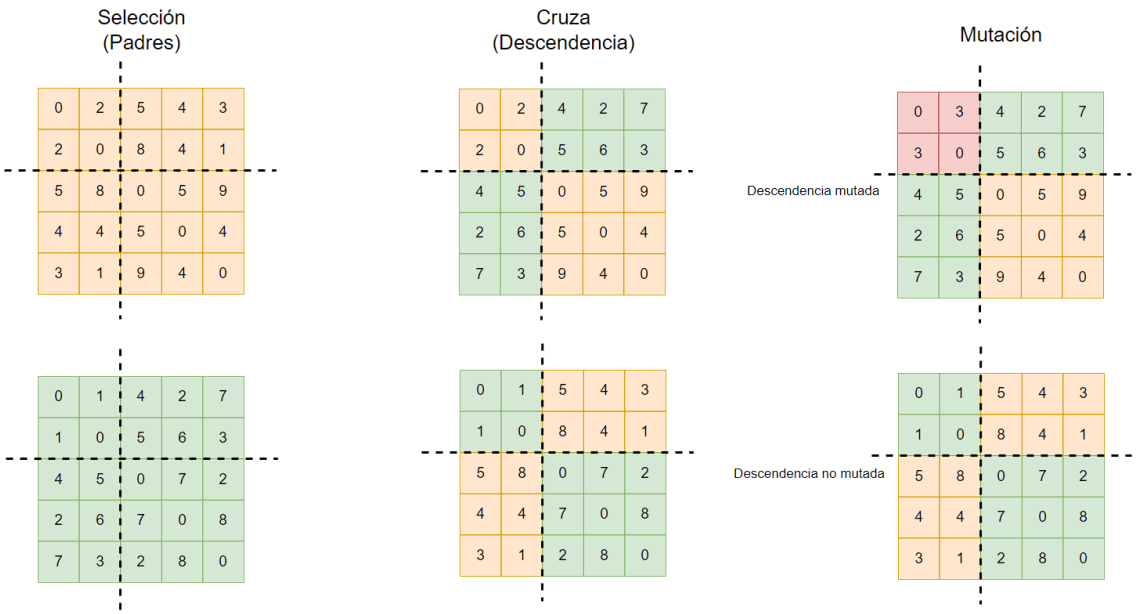


Figura 4.5: Ilustración de los operadores evolutivos utilizados en los vectores de precios

4.3.8 Co-evolución entre poblaciones

Como ya se mencionó, la co-evolución cooperativa implica aprovechar las características de una población para mejorar la otra. Bajo este marco, una vez que se ha actualizado la población de árboles, se utilizan para llevar a cabo nuevamente el proceso de encontrar límites en los vectores de precios. Luego, la población de los nuevos vectores de precios evoluciona, y se utilizarán para actualizar la población de redes con estructura de árbol.

Específicamente, los nuevos vectores de precios se consideran conjuntamente con la población actual de árboles, y se vuelve a resolver el problema del seguidor. Al hacer esto, se identifican los hubs con mayor frecuencia (los que aparecen más frecuentemente en las soluciones), y esa información se usa para crear nuevos árboles. Es evidente que hay un intercambio de información de la población de precios a la población de árboles. Además, los nuevos árboles se utilizarán para calcular nuevos límites en los precios y, en consecuencia, una nueva población de precios. Por lo tanto, también existe un intercambio de información de la población de árboles a la población de precios; es decir, el proceso co-evolutivo se realiza en dos direcciones.

El primer criterio de parada del Co-EA implica alcanzar, por segunda vez consecutiva, un número predeterminado de iteraciones sin reportar una mejora en la función objetivo del líder. Al alcanzar este número predeterminado de iteraciones por primera vez, se agrega un componente de diversificación. Específicamente, la población actual de

árboles se reemplaza por una nueva población de árboles generada de manera greedy. El algoritmo continúa su proceso con el objetivo de mejorar las soluciones hasta que se alcance un número máximo de generaciones. Si se alcanza este número máximo de generaciones, se aplica el otro criterio de parada y el algoritmo termina.

4.4 Resultados

En esta sección, reportaremos los resultados de los experimentos computacionales realizados para analizar el desempeño del Co-EA propuesto para resolver el BTHLPwP. En nuestros experimentos, adaptamos los conocidos conjuntos de datos AP, CAB y TR de instancias para problemas de localización de hubs propuestos por [352]. Los tamaños de las instancias se muestran en la Tabla 4.1. Se seleccionaron instancias con 10, 15, 20 y 25 sitios potenciales y clientes. Para las instancias con 10 sitios y clientes potenciales, debemos ubicar 3 y 5 hubs. Para el resto (15, 20 y 25 sitios potenciales), ubicamos 3, 5 y 7 hubs, respectivamente. Cada una de estas instancias se crea utilizando 5 costos de construcción diferentes para medir el impacto de usar diferentes valores. Con estas configuraciones, tenemos 55 instancias para cada uno de los tres conjuntos de datos considerados (AP, CAB y TR), dando un total de 165 instancias.

Instancias	Sitios Potenciales	Clientes	p	# de Instancias
AP	10	10	3, 5	10
	15	15	3, 5, 7	15
	20	20	3, 5, 7	15
	25	25	3, 5, 7	15
CAB	10	10	3, 5	10
	15	15	3, 5, 7	15
	20	20	3, 5, 7	15
	25	25	3, 5, 7	15
TR	10	10	3, 5	10
	15	15	3, 5, 7	15
	20	20	3, 5, 7	15
	25	25	3, 5, 7	15
			Total	165

Tabla 4.1: Resumen de las instancias probadas

4.4.1 Parámetros usados en el Co-EA

En el algoritmo propuesto, algunos parámetros deben ser identificados y calibrados para mejorar su desempeño. Específicamente, el tamaño de la población de árboles, el

tamaño de la población de precios, las probabilidades de cruce y mutación, el número de iteraciones sin mejora y el número máximo de generaciones. Para determinar los parámetros en nuestros experimentos computacionales, se realizaron pruebas preliminares. Sin embargo, es importante señalar que [319] discute los resultados de experimentos a gran escala enfocados en la robustez de la configuración de parámetros para algoritmos evolutivos. Concluyen que, en general, la configuración de parámetros no es relevante para ejecuciones lo suficientemente largas.

Para el Co-EA propuesto, se genera una población inicial de 50 árboles y precios. Los árboles y los precios evolucionan a través de un EA. La probabilidad de entrar en el cruce para cada población se establece en 0,8. Luego, el resultado del cruce entra en la fase de mutación para cada población con una probabilidad de mutación de 0,2. Estos valores están dentro del rango de valores recomendados en [319] como valores pertinentes. El algoritmo termina después de haber realizado 40 iteraciones o cuando se registran 15 iteraciones sin mejora en la solución incumbente.

4.4.2 Resultados Computacionales

El Co-EA se implementó en Python 3.9. Los experimentos computacionales se llevaron a cabo utilizando un procesador Intel(R) Core i5-10400F con 16 GB de RAM bajo Windows 10 Professional. Debido a la aleatoriedad del algoritmo, se realizaron 10 corridas para cada instancia. La etiqueta *n*-*p*-`l` indica el número de sitios potenciales (n)– que coincide con el número de usuarios en estas instancias, el número de hubs a localizar (p) y un indicador de instancia (`l`)– que identificamos con las letras `A`, `B`, `C`, `D`, `E`. Denotamos por F_{best} la mejor ganancia del líder obtenida en las 10 corridas. Las ganancias promedio y peor se denotan como F_{avg} y F_{worst}, respectivamente. Además, `CPUTime` corresponde al tiempo computacional requerido (en segundos). Los resultados detallados de nuestros experimentos se muestran en las tablas 4.2, 4.4, y 4.3.

Para mejorar la legibilidad de los resultados obtenidos, en la Tabla 4.5 proporcionamos un resumen donde se muestran las diferentes desviaciones obtenidas entre las ganancias del líder. Esta tabla está organizada en tres partes, cada una para los diferentes tipos de instancias utilizadas en la experimentación (CAB, AP y TR). Para cada una de ellas, separamos los resultados por número de sitios potenciales (n) y número de hubs a localizar (p). En la columna **Dev. Best-Average** se calcula la desviación entre la mejor ganancia encontrada y la ganancia promedio en las 10 corridas. Asimismo, calculamos la desviación entre la mejor ganancia encontrada y la peor en las diez ejecuciones, que está representada por **Dev. Best-Worst**. Para cada una de estas dos desviaciones, se reportan la desviación mínima, promedio y máxima en las cinco instancias. En la columna **Avg.**

CPU Time reportamos los tiempos de ejecución promedio para resolver cada una de las instancias.

4.4.3 Discusión

En esta subsección, discutimos los resultados obtenidos por la experimentación computacional presentando en la subsección anterior.

	CAB dataset			
	F_{best}	F_{avg}	F_{worst}	Time
10-3-A	4259253.4	4254401.4	4251222.4	632.57
10-3-B	4247766.5	4244774.6	4240301.2	709.70
10-3-C	4249089.3	4244736.7	4240632.5	696.36
10-3-D	4258067.3	4255067.7	4251455.3	622.88
10-3-E	4267256.4	4262889.0	4260863.7	550.02
10-5-A	6098463.4	6095526.8	6093092.6	669.96
10-5-B	6091267.3	5942705.2	5845213.4	1110.76
10-5-C	6061902.4	6005023.0	5978727.6	851.53
10-5-D	6086105.1	5906174.0	5716329.9	984.38
10-5-E	6108980.9	5914112.3	5728952.8	845.56
15-3-A	8145316.5	8102254.4	8046555.3	1901.87
15-3-B	8137482.8	8054477.5	7749293.4	1941.44
15-3-C	8155967.0	7833093.3	7530094.9	1256.22
15-3-D	8144527.6	7990885.0	7711455.5	1586.16
15-3-E	8123876.2	8099005.0	8054983.6	1686.30
15-5-A	12448048.3	11873499.6	11000691.4	1546.99
15-5-B	13176528.1	12595400.9	12090089.7	1568.12
15-5-C	13169829.9	12819402.0	12592376.5	1918.94
15-5-D	13582671.6	12982760.7	11318531.7	1818.19
15-5-E	12967571.8	12542077.7	12172454.6	2110.03
15-7-A	15257662.1	15045892.7	14732619.8	1877.66
15-7-B	14590726.0	14113249.5	13538403.8	2357.25
15-7-C	16312487.8	15625346.0	15113727.6	1982.57
15-7-D	14916659.2	13929796.2	12897178.8	2833.21
15-7-E	16059345.9	15792027.2	15445874.2	1897.40
20-3-A	9070604.4	8892267.7	8629928.0	2616.02
20-3-B	9030977.6	8967006.5	8795570.7	2958.74
20-3-C	9113839.6	9046953.7	9012763.7	2604.50
20-3-D	9058215.7	8918393.8	8754388.4	2691.05
20-3-E	9074493.5	9020300.9	8990988.2	2329.63
20-5-A	16030351.1	15319494.3	13724008.8	3817.41
20-5-B	15865982.9	14615737.0	12458174.4	2710.42
20-5-C	16809531.1	15840305.4	15225731.7	2692.39
20-5-D	15694646.9	14723165.4	12810320.2	3141.84
20-5-E	14058902.3	13673225.6	13253373.3	2922.98
20-7-A	20684534.0	19499673.2	18404423.2	4485.34
20-7-B	18698127.6	17595696.0	16098511.1	4982.45
20-7-C	19193747.4	17525916.0	16942224.1	3936.65
20-7-D	19661403.2	18232927.3	17004331.6	5260.12
20-7-E	19531918.8	18499174.4	17756908.5	4402.06
25-3-A	34829601.6	32699049.2	31573433.8	4335.33
25-3-B	36131738.8	34118777.1	32909578.3	4593.57
25-3-C	29806673.9	27885836.9	27074793.5	3427.82
25-3-D	32744119.4	30235457.3	29029392.7	4248.72
25-3-E	32936708.6	30337613.3	29237501.3	5239.91
25-5-A	54146259.5	50547866.5	47760269.6	3487.63
25-5-B	58702228.3	54792658.3	53015842.0	2316.54
25-5-C	55118073.8	52399443.1	50224210.3	2684.22
25-5-D	53033637.8	49964221.6	48769904.5	2296.00
25-5-E	52495040.6	50387553.6	47297297.5	2444.80
25-7-A	62147953.4	59949895.3	57313392.1	1922.45
25-7-B	67577551.3	64862020.3	62964921.8	3629.07
25-7-C	61402557.6	59319755.7	57165867.9	1697.37
25-7-D	66428867.8	62060115.2	57466630.1	3075.42
25-7-E	60214940.8	55691251.6	52665205.5	2102.27

Tabla 4.2: Ganancias obtenidas por las instancias en el CAB dataset.

	AP dataset			
	F_{best}	F_{avg}	F_{worst}	Time
10-3-A	2801108.4	2799039.2	2794353.5	740.96
10-3-B	2756303.9	2751970.9	2750046.1	866.94
10-3-C	2799882.3	2796349.0	2793642.9	590.61
10-3-D	2828197.3	2810343.6	2747398.2	641.92
10-3-E	2759539.7	2756193.9	2753244.8	822.68
10-5-A	6128109.8	6124543.8	6121152.2	661.14
10-5-B	5637891.2	5635165.6	5624588.2	644.62
10-5-C	5795511.3	5792747.7	5789989.6	864.83
10-5-D	5434321.6	5395870.9	5249757.2	587.40
10-5-E	5898239.4	5894885.1	5884947.8	654.22
15-3-A	2363901.0	2321453.1	2261387.1	1245.86
15-3-B	2110428.0	2019521.3	1906484.9	1156.53
15-3-C	2178031.1	2133200.6	1989716.2	1775.48
15-3-D	2055614.8	2035286.1	2023197.0	1554.87
15-3-E	2206165.4	2170968.5	2088966.6	1199.58
15-5-A	3481137.6	3306742.8	3166685.9	994.57
15-5-B	3381136.0	3251673.9	3001814.4	1275.86
15-5-C	3695346.3	3464646.4	3206934.7	1364.48
15-5-D	3765335.7	3498210.5	3229047.4	1038.45
15-5-E	3802335.9	3625477.4	3499851.4	1328.65
15-7-A	5081509.2	4904700.3	4585683.2	1214.99
15-7-B	5256885.4	4891183.4	4627195.6	1065.55
15-7-C	5194416.2	4980734.0	4863180.0	1529.91
15-7-D	5216286.4	4942316.1	4674229.3	1588.16
15-7-E	5243457.6	4954913.0	4785401.6	1672.15
20-3-A	1727146.4	1607768.3	1448735.2	3421.67
20-3-B	1341549.5	1261038.5	1084856.7	3303.99
20-3-C	1708392.1	1665371.5	1569430.8	4027.12
20-3-D	1789467.5	1676864.1	1560421.4	3592.27
20-3-E	1484997.2	1424310.9	1262536.4	2934.09
20-5-A	2676001.0	2477167.9	2250298.9	2043.06
20-5-B	2351173.9	2156002.0	2033531.1	1599.73
20-5-C	2825040.6	2548833.0	2369628.0	2553.90
20-5-D	2750797.8	2570415.3	2402453.4	2870.03
20-5-E	2289712.0	2141306.2	1979386.3	1243.46
20-7-A	3579742.3	3338546.2	3123937.7	1895.37
20-7-B	3020644.2	2590051.4	2197051.0	1837.68
20-7-C	3835699.8	3487008.2	3240761.1	2108.99
20-7-D	3754231.8	3588942.1	3410012.1	2130.84
20-7-E	3614007.0	3264023.0	3077947.5	2030.16
25-3-A	628357.6	581291.2	527339.5	3528.41
25-3-B	59206.7	55669.1	53331.9	4129.41
25-3-C	528845.3	502342.9	470727.8	3638.24
25-3-D	151750.3	141483.9	134991.9	4003.68
25-3-E	659895.1	640746.6	621820.4	3590.87
25-5-A	618331.4	569067.0	530136.7	3415.44
25-5-B	768147.8	697933.4	658195.6	2260.43
25-5-C	788717.1	680796.7	583266.1	2381.17
25-5-D	565834.5	537919.2	517152.0	2179.37
25-5-E	798573.9	710236.6	634899.6	4431.76
25-7-A	954290.9	844318.6	741908.9	2402.75
25-7-B	983592.9	871437.3	809940.7	4464.87
25-7-C	951696.7	829687.1	711225.4	1943.72
25-7-D	571456.1	510610.3	399415.7	2628.07
25-7-E	692426.6	612181.8	529469.8	3122.75

Tabla 4.3: Ganancias obtenidas por las instancias en el AP dataset.

	TR dataset			
	F_{best}	F_{avg}	F_{worst}	Time
10-3-A	95418682.4	94832530.6	94423578.3	724.53
10-3-B	87441200.0	85788571.9	82824425.8	657.29
10-3-C	88103685.3	86722958.1	83698513.4	631.81
10-3-D	92486860.5	91900095.1	90056286.3	628.87
10-3-E	88106286.8	85743568.3	83131819.7	583.61
10-5-A	177315374.7	170635127.7	164125659.8	826.06
10-5-B	172031522.9	166477630.8	162955201.3	538.30
10-5-C	170585908.6	164136644.7	154242212.9	484.51
10-5-D	175117723.0	171798338.3	167686529.4	697.30
10-5-E	173005646.0	163290970.2	155312761.7	277.07
15-3-A	119324076.0	116144817.9	105519347.8	781.97
15-3-B	128340715.4	120061582.7	108215940.7	979.41
15-3-C	127700811.7	119948186.8	108375891.1	862.32
15-3-D	118985900.1	113239660.0	104155891.3	670.69
15-3-E	128763897.2	126394171.3	123024395.3	869.07
15-5-A	193412221.8	182949258.8	169540058.2	875.75
15-5-B	195975104.2	185575307.3	170012851.4	818.90
15-5-C	189429899.4	184114986.4	180753325.0	611.00
15-5-D	188707534.3	182219165.7	166622158.2	736.11
15-5-E	191275200.5	181739421.3	168532902.4	665.37
15-7-A	235865259.5	228685574.4	223722501.0	556.18
15-7-B	235772679.4	224815946.3	215322562.5	822.64
15-7-C	233459820.7	225357597.4	217048783.2	580.71
15-7-D	231783185.0	225189514.6	216223983.9	726.26
15-7-E	231893261.4	218457252.7	206740979.4	492.68
20-3-A	186179900.9	182644751.5	179762837.9	1054.83
20-3-B	193143003.0	184001414.5	179168527.5	1078.17
20-3-C	183594265.1	179741455.2	173560526.8	890.26
20-3-D	182817601.3	180558918.0	174187948.5	1149.74
20-3-E	183904422.5	180790522.8	175386529.0	878.49
20-5-A	261392222.7	254470625.0	245264344.2	1056.78
20-5-B	279316231.5	261229755.0	239359595.3	1045.39
20-5-C	281862536.8	265215734.8	242919242.8	1278.62
20-5-D	269553664.8	261733596.4	248553444.1	1078.98
20-5-E	271139782.6	258972489.8	237399363.4	948.62
20-7-A	325492874.5	304337900.7	295520676.8	917.83
20-7-B	324057641.8	315736451.0	305076175.1	1228.90
20-7-C	320132949.8	309141722.4	299045907.2	1564.99
20-7-D	343738770.3	322023092.0	302299410.8	1344.81
20-7-E	339386602.4	318261086.5	299541945.7	1582.04
25-3-A	339851039.5	323015570.6	309506247.8	1600.61
25-3-B	337039428.4	331379691.7	323352414.7	1841.10
25-3-C	350934406.8	337494525.9	329460237.4	1491.73
25-3-D	331016936.5	323805948.8	316971175.2	1575.33
25-3-E	352806147.9	332430646.0	320983962.8	2986.70
25-5-A	417354855.1	411368709.3	400009664.8	1985.44
25-5-B	436830253.8	411440347.0	384347789.2	959.72
25-5-C	427628111.3	413690475.0	395123809.7	914.54
25-5-D	437892147.0	432626352.3	429154351.9	1179.15
25-5-E	443568803.0	428414759.7	415727926.8	1492.85
25-7-A	518219749.3	515977438.3	513874338.1	1670.53
25-7-B	485133826.8	470758914.2	459338944.9	1348.12
25-7-C	492533621.8	485726855.5	481268428.8	1468.07
25-7-D	512207463.3	498803553.3	491250311.6	1977.71
25-7-E	511094340.2	491936200.7	453731335.1	1467.03

Tabla 4.4: Ganancias obtenidas por las instancias en el TR dataset.

Durante la experimentación, notamos que para todas las instancias, la convergencia del Co-EA con respecto al diseño de la red fue más rápida que la de las decisiones de

precios. Este hallazgo está respaldado por las desviaciones reportadas. Cabe destacar que para las instancias más pequeñas, la desviación fue inferior al 1 %. Sin embargo, a medida que las instancias aumentaron de tamaño, el aspecto combinatorio del problema de diseño de redes en forma de árbol condujo a un aumento en las desviaciones reportadas. En estos casos, la desviación máxima aumentó hasta un 10 %. Esto puede atribuirse al hecho de que, dado que las soluciones de precios son vectores de valores reales, su convergencia es más difícil de lograr. Además, las instancias de conjuntos de datos más difíciles de resolver fueron las de AP, que reportaron la máxima desviación entre todas las instancias.

Instances	n	p	**Dev Best-Average**			**Dev Best-Worst**			**Av. CPU Time**
			Min	Average	Max	Min	Average	Max	
CAB	10	3	0.07	0.09	0.11	0.15	0.17	0.20	642.30
		5	0.05	1.91	3.19	0.09	3.56	6.22	892.43
	15	3	0.31	1.54	3.96	0.85	3.96	7.67	1674.39
		5	2.66	3.87	4.62	4.38	9.41	16.67	1792.45
		7	1.39	3.43	6.62	3.44	7.07	13.54	2189.61
	20	3	0.6	1.10	1.97	0.92	2.57	4.86	2639.98
		5	2.74	5.40	7.88	5.73	13.88	21.48	3057.01
		7	5.29	6.57	8.69	9.09	11.85	13.9	4613.32
	25	3	5.57	6.73	7.89	8.92	10.00	11.34	4369.07
		5	4.01	5.60	6.66	8.04	9.66	11.79	2645.83
		7	3.39	5.00	7.51	6.83	9.50	13.49	2485.31
AP	10	3	0.07	0.22	0.63	0.22	0.75	2.86	645.22
		5	0.05	0.18	0.71	0.1	0.81	3.4	564.65
	15	3	0.99	2.15	4.31	1.58	5.91	9.66	832.69
		5	3.83	5.36	7.09	7.96	11.13	14.24	741.43
		7	3.48	5.06	6.96	6.38	9.45	11.98	635.69
	20	3	2.52	5.16	6.91	8.13	14.23	19.13	1010.3
		5	6.48	7.70	9.78	12.66	14.35	16.12	1081.68
		7	4.40	8.83	14.25	9.17	15.90	27.27	1327.71
	25	3	2.90	5.63	7.49	5.77	10.76	16.08	1899.09
		5	4.93	9.35	13.68	8.6	16.74	26.08	1306.34
		7	10.65	11.59	12.82	17.65	23.76	30.11	1586.29
TR	10	3	0.61	1.47	2.68	1.04	3.92	5.65	732.62
		5	1.90	3.66	5.62	4.24	7.35	10.23	682.44
	15	3	1.84	4.36	6.45	4.46	11.86	15.68	1386.46
		5	2.81	4.39	5.41	4.58	10.75	13.25	1200.4
		7	2.84	3.95	5.79	5.15	7.68	10.85	1414.15
	20	3	1.24	2.33	4.73	3.45	5.10	7.24	3455.83
		5	2.65	4.48	6.48	6.17	10.90	14.31	2062.04
		7	2.57	5.00	6.5	5.86	9.09	12.06	2000.61
	25	3	1.68	3.68	5.78	4.06	6.47	9.02	3778.12
		5	1.2	3.02	5.81	2.0	6.41	12.01	2933.63
		7	0.43	2.22	3.75	0.84	4.752	11.22	2912.43

Tabla 4.5: Resumen de las desviaciones encontradas

Como era de esperar, el tiempo computacional aumenta a medida que aumenta el número de sitios potenciales (n) y/o el número de hubs a localizar (p). Sin embargo, a pesar del incremento en el tamaño de las instancias, el crecimiento en el tiempo computacional parece ser lineal en lugar de cuadrático. En la Figura 4.6, se presenta el de rendimiento de nuestro algoritmo, diferenciando por el tipo de instancia (CAB, AP y TR). Como se observa, las instancias de tipo TR parecen ser menos desafiantes que las de tipo AP y CAB, y además, la mayoría de las instancias se resolvieron en menos de una hora.

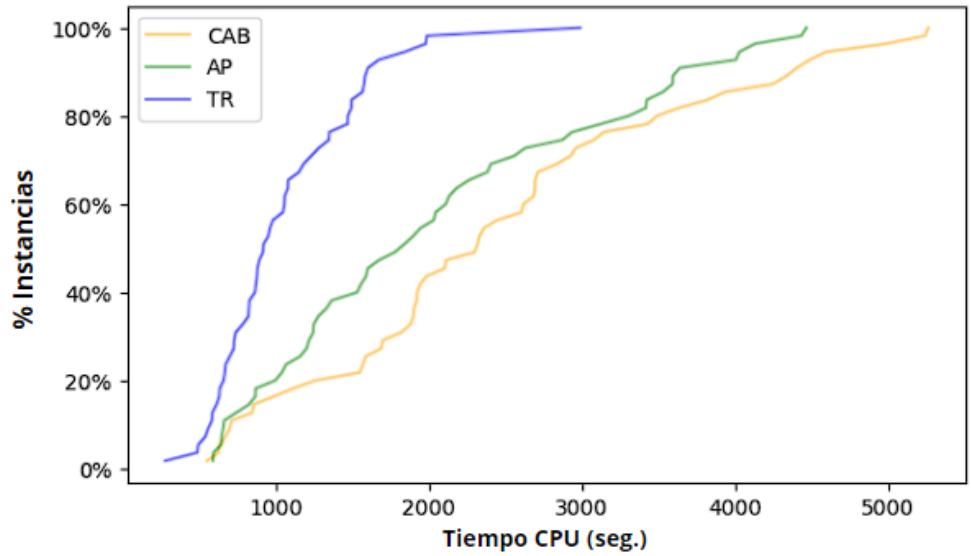


Figura 4.6: Rendimiento por tipo de instancia

En la Figura 4.7, mostramos los boxplots de los diferentes tiempos de CPU requeridos por nuestro enfoque en los tres tipos diferentes de instancias. Se puede observar que los tres conjuntos de datos de instancias requieren tiempos computacionales promedio similares. Sin embargo, nuestro algoritmo parece ser ligeramente más estable en las instancias TR.

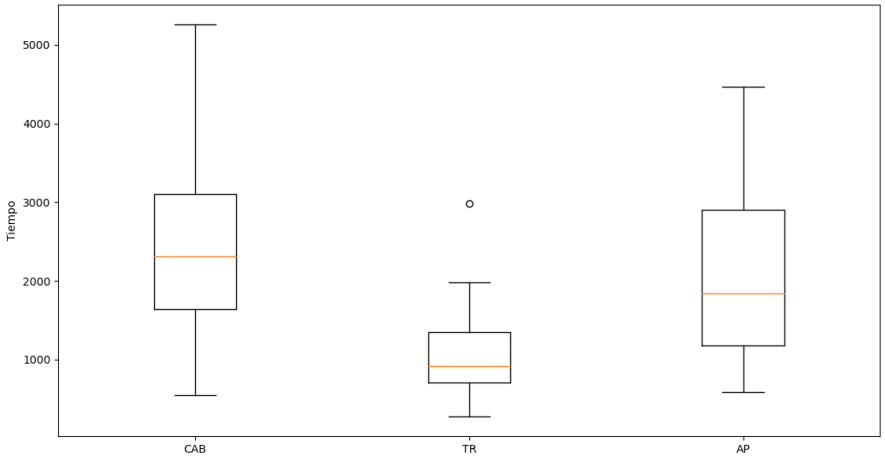


Figura 4.7: Boxplots de los tiempos computacionales requeridos por nuestro algoritmo

Por otro lado, en la Figura 4.8, mostramos los boxplots con respecto a las desviaciones porcentuales del peor valor de cada instancia con respecto a la mejor ganancia obtenida. Nuevamente, se puede concluir que el algoritmo propuesto demuestra estabilidad en los tres tipos diferentes de instancias.

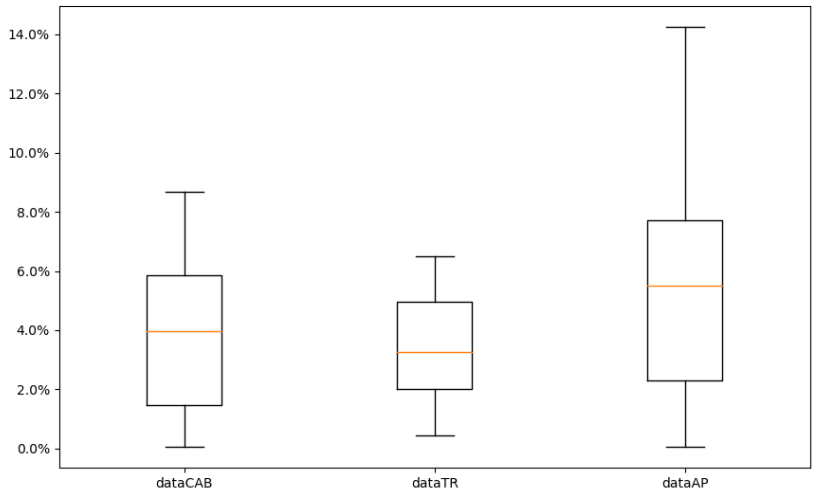


Figura 4.8: Boxplots percent deviations reported by our algorithm in the three types of instances.

Una observación interesante de nuestros experimentos es que en el BTHLPwP, no existe una relación directa entre el número de instalaciones a ubicar (p) y la ganancia

obtenida por el líder, como es habitual en los problemas clásicos de ubicación de instalaciones.

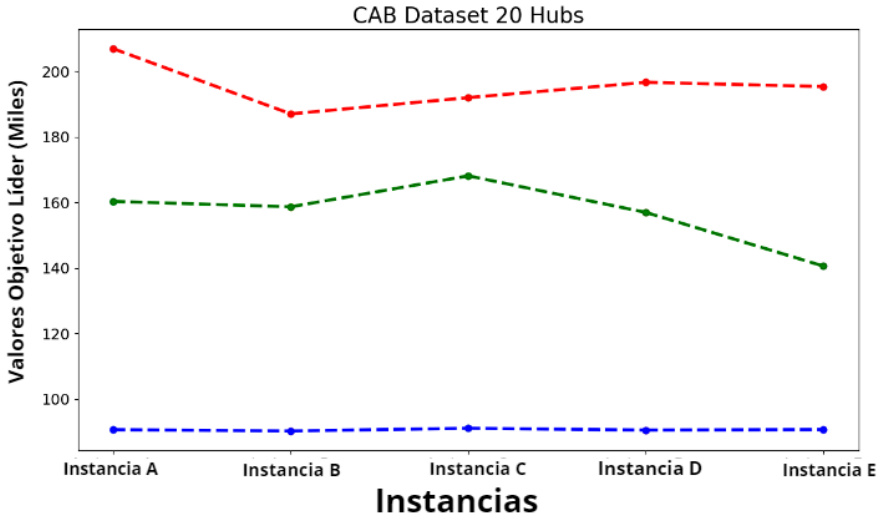


Figura 4.9: Ganancia del líder para la instancia del CAB dataset con 20 sitios potenciales, variando el número de hubs localizados

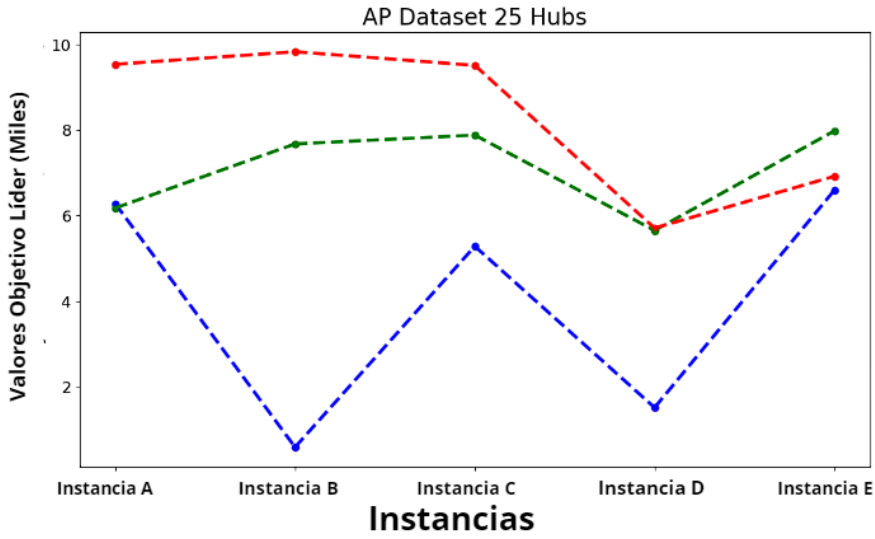


Figura 4.10: Ganancia del líder para la instancia del AP dataset con 25 sitios potenciales, variando el número de hubs localizados

Aumentar el número de hubs no siempre resulta en una ganancia mayor: mientras que los costos de configuración aumentan, las ganancias por el uso de los clientes pueden ser similares, reduciendo consecuentemente la ganancia. Por lo tanto, en algunas instancias, observamos que a mayor valor de p para la misma instancia, mayor es la ganancia,

mientras que en otras instancias, esta relación no se verifica. Por ejemplo, en la Figura 4.9, representamos las ganancias para las cinco instancias (A, B, C, D y E) con 20 sitios potenciales en el conjunto de datos CAB. Las ganancias se detallan en las tablas mostradas. Los gráficos azul, rojo y verde representan la ganancia obtenida al ubicar 3, 5 y 7 hubs, respectivamente. Es evidente que, en general, aumentar el número de hubs ubicados se correlaciona con ganancias más altas. Sin embargo, instancias específicas muestran tendencias variables. En la Figura 4.10, presentamos información similar para las instancias de AP con 25 sitios potenciales. Aquí, ubicar 3 hubs a menudo produce una ganancia mayor que ubicar 5 hubs, como se observa en la instancia A. Sin embargo, la instancia D no muestra una diferencia significativa en la ganancia entre ubicar 5 o 7 hubs. Por el contrario, la instancia E muestra que la ganancia es mayor cuando el líder ubica 5 hubs en lugar de 7 hubs.

Capítulo 5

Optimización Binivel Multi-objetivo: Múltiples enfoques y algoritmos metaheurísticos.

Como ya se ha mencionado, un problema de programación binivel es un tipo específico de problema que involucra dos niveles de decisión jerarquizados e interrelacionados, cada uno asociado con líder y un seguidor. La dificultad de esta clase de problemas yace en que para obtener soluciones factibles binivel, el seguidor debe resolver óptimamente su problema parametrizado por la decisión del líder.

Cuando el seguidor considera simultáneamente dos objetivos, no está claro cuál será su reacción óptima debido que al resolver un problema bi-objetivo, el concepto de una solución óptima se modifica por un conjunto de soluciones dominadas que forman el llamado frente de Pareto. Como resultado, el seguidor debe seleccionar una de estas soluciones como su reacción racional. Sin embargo, este problema fundamental para la programación binivel ha recibido atención limitada en la literatura existente. En este capitulo, el objetivo es demostrar el impacto significativo que puede tener ciertos supuestos de la reacción del seguidor en un problema binivel. Para ello, una nueva variante de un problema de localización de instalaciones competitivas se presenta resultando en un problema binivel binario con dos objetivos en el nivel inferior.

El resto del capitulo se conforma de la siguiente manera: En la sección 5.1 se mostrará una pequeña introducción a los problemas de programación binivel multi-objetivo y distintos enfoques que se han tomado para resolverlos. Por otro lado, en la sección 5.2 se mostrará una aplicación de un problema binivel con dos objetivos en el problema del nivel inferior a través de un problema de localización de plantas competitivas, en la sección 5.3 se introduce el modelo matemático a estudiar, luego, en la sección 5.4 se muestra (a través de un ejemplo) una manera enumerativa para ver el impacto de los distintos enfoques al resolver estos problemas además de introducir dos nuevos enfoques. Por último en

la sección **??** se resolverán múltiples instancias del problema en base a dos enfoques adaptando un algoritmo metaheurístico-exacto y un algoritmo metaheurístico-aproximativo y comparando los resultados para obtener conclusiones.

5.1 Antecedentes: Optimización Binivel Multi-objetivo

Otra manera de representar un modelo general de optimización binivel es de la siguiente forma:

$$\text{“}\max_{x\in X}\text{”} \quad F(x,y) \tag{5.1}$$

$$\text{s.t} \quad G(x,y) \leq b_1 \tag{5.2}$$

$$y \in M(x) \tag{5.3}$$

donde $F : X \times Y \to \mathbb{R}$ y $f : X \times Y \to \mathbb{R}$ además $b_1 \in \mathbb{R}^p, b_2 \in \mathbb{R}^q$. Consideremos el conjunto $M(x) = \arg\min_{y\in Y}\{f(x,y) : g(x,y) \leq b_2\}$ como el conjunto de la reacción racional del seguidor. Las comillas en la ecuación 5.1 considera la ambigüedad de la reacción del seguidor cuando múltiples soluciones óptimas existen en $M(x)$ para una x dada.

Si en el problema consideramos más de un objetivo, se convierte en un problema binivel multi-objetivo. A continuación, se presenta el problema general binivel multi-objetivo:

$$\text{“}\max_{x\in X}\text{”} \quad \mathbf{F}(x,y) = [F_1(x,y), F_2(x,y), ..., F_l(x,y)] \tag{5.4}$$

$$\text{s.t} \quad G(x,y) \leq b_1 \tag{5.5}$$

$$y \in \arg\min_{y\in Y}\{\mathbf{f}(x,y) = [f_1(x,y), f_2(x,y), ..., f_m(x,y)] : g(x,y) \leq b_2\} \tag{5.6}$$

donde $\mathbf{F} : X \times Y \to \mathbb{R}^l$, $\mathbf{f} : X \times Y \to \mathbb{R}^m$, $b_1 \in \mathbb{R}^p$, y $b_2 \in \mathbb{R}^q$.

En el problema de optimización definido por las ecuaciones (5.4)-(5.6), el nivel superior tiene l funciones objetivo y el nivel inferior tiene m funciones objetivo.

Cuando $l = 1$ y $m = 2$, es decir, el nivel superior es un problema mono-objetivo y el nivel inferior es un problema bi-objetivo, pasa a llamarse Problema de Optimización Binivel Semivectorial (SVBP por sus siglas en inglés) [47, 115]. Consideramos el modelo general SVBP presentado en [19], el cual se introduce a continuación:

$$\text{“}\max_{x \in X}\text{”} \quad F(x, y) \tag{5.7}$$

$$\text{s.t} \quad G(x, y) \leq b_1 \tag{5.8}$$

$$y \in \arg\min_{y \in Y}\{[f_1(x, y), f_2(x, y)] : g(x, y) \leq b_2\} \tag{5.9}$$

Como se menciona en [19], en un SVBP solo las soluciones eficientes del problema del nivel inferior para un x' dado del nivel superior son factibles. Así, una solución factible y' del nivel inferior es una solución débilmente eficiente si y solo si no existe otra y factible tal que $f_i(x', y) < f_i(x', y')$ para $i = 1, 2$. Además, una solución factible y' del nivel inferior es una solución eficiente si y solo si no existe otra y factible tal que $f_i(x', y) \leq f_i(x', y')$ para $i = 1, 2$, y $f_i(x', y) < f_i(x', y')$ al menos en un i. Asumimos el criterio de dominancia de Pareto (véase [109]), que se expresa como $\mathbf{f}(x', y) \prec \mathbf{f}(x', y')$ y se refiere a las soluciones eficientes o no dominadas. Entonces, el conjunto de soluciones eficientes para una solución del nivel superior x' se define como:

$$\begin{aligned} M_{Ef}(x') \;=\; & \{y' \in Y \text{ and } g(x', y') \leq b_2 : \text{ no hay } y \in Y \text{ y} \\ & g(x', y') \leq b_2 \text{ tal que } \mathbf{f}(x', y) \prec \mathbf{f}(x', y')\} \end{aligned}$$

Cuando aparece un problema multi-objetivo en el nivel inferior, la reacción del seguidor se vuelve más compleja, ya que no se puede alcanzar una única solución óptima. Por lo tanto, se vuelve crucial establecer una posición clara respecto a cuál solución no dominada elegirá el seguidor como su reacción. Para ello vamos a explorar algunos de los enfoques asumidos que se encuentran en la literatura existente.

5.1.1 Enfoques para resolver SVBP

Las suposiciones clásicas optimista y pesimista hechas para problemas binivel mono-objetivo pueden adaptarse a nuestro problema. Como se presenta en [114], el enfoque optimista implica seleccionar la reacción del seguidor que resulta en el resultado más favorable para la función objetivo del líder. Por el contrario, el enfoque pesimista contempla el peor escenario para el líder, donde el seguidor elige la solución con el impacto más adverso en la función objetivo del líder. Los enfoques optimistas y pesimistas pueden aplicarse en SVBP pero necesitan ser adaptados para seleccionar una solución del frente de Pareto obtenido. La solución optimista (x^o, y^o) se determina mediante la siguiente reacción racional del seguidor:

$$\max_{x,y}\{F(x,y) : y \in M_{Ef}(x), G(x,y) \leq b_1\}$$

Por otro lado, en el enfoque pesimista, la solución pesimista (x^p, y^p) se determina mediante:

$$\max_{x}\{\min_{y}\{F(x,y) : y \in M_{Ef}(x)\} : G(x,y) \leq b_1\}$$

En [18, 19], las ideas optimista y pesimista se extienden introduciendo otro par de soluciones de interés: la solución engañosa y la solución recompensante. Para la solución engañosa, se considera la decisión optimista del líder x^o, y la solución engañosa (x^d, y^d) se forma con $x^d = x^o$ y y^d dado por $\min_{y}\{F(x^o, y) : y \in M_{Ef}(x^o)\}$. En contraste, para la solución recompensante, el líder considera su decisión pesimista x^p, y la solución recompensante (x^r, y^r) se construye con $x^r = x^p$ y y^r dado por $\max_{y}\{F(x^p, y) : y \in M_{Ef}(x^p), G(x^p, y) \leq b_1\}$. La lógica detrás de estas dos nuevas soluciones es que cuando el líder asume un enfoque optimista (pesimista), está preparado para el mejor (peor) escenario. Sin embargo, en la práctica, esta suposición puede no ser cumplida por la reacción del seguidor. Entonces, el seguidor selecciona otra solución a pesar de que el líder espera el resultado más favorable. Esto lleva a la solución extrema engañosa (recompensante).

Otro enfoque es el llamado "solución mejor comprometida", mencionado en [2]. En este enfoque, la solución ideal basada en la frontera de Pareto obtenida se utiliza como punto de referencia para encontrar la solución no dominada que conduzca a la mejor solución de compromiso. Es evidente que la solución ideal es inadmisible para el problema

binivel, pero dado que la solución de mejor compromiso se encuentra en la frontera de Pareto, es factible para el problema binivel.

Para encontrar la solución ideal, primero se identifica el valor máximo para cada uno de los objetivos (en el caso de un problema de maximización). Luego, la solución ideal se obtiene combinando las coordenadas de estas soluciones extremas de acuerdo con la siguiente regla:

$$\text{Ideal} = \left(\max_{x \in M_{Ef}} f_1(x), \max_{x \in M_{Ef}} f_2(x) \right)$$

Después de eso, se mide el grado de similitud con las soluciones en el frente de Pareto según un criterio predefinido. Por ejemplo, la distancia euclidiana se puede usar para identificar la solución mejor comprometida. Por lo tanto, la solución no dominada que está más cerca de la solución ideal se elige como la solución mejor comprometida.

Dentro de esta capítulo presentamos dos nuevos enfoque basado en la densidad del frente de Pareto, llamado enfoque neutral y enfoque incómodo. La justificación detrás de este enfoque surge al tratar casos que involucran información incompleta sobre el comportamiento del tomador de decisiones del nivel inferior. En tales instancias, el tomador de decisiones del nivel superior estima la reacción esperada del nivel inferior. Este enfoque se puede ver como el enfoque bayesiano inducido por distribuciones uniformes, como se propone en [295]. En nuestro problema, nuestro objetivo es considerar una solución no dominada del nivel inferior dentro de una región de mayor densidad, lo que representa una estimación neutral de la reacción del nivel inferior. En términos más simples, apuntar a una solución en una región más densa aumenta la probabilidad de que el tomador de decisiones del nivel inferior seleccione una de esa región, dado el mayor subconjunto de soluciones. En contraste, el enfoque incómodo, apunta a una solución menos densa del frente de pareto. Estas alternativa se vuelven relevantes cuando los tomadores de decisiones muestran cierto grado de aversión al riesgo.

5.2 Un problema de localización de instalaciones competitivas.

La localización de instalaciones competitivas (CFL) es un proceso estratégico de toma de decisiones que implica identificar la ubicación óptima para una instalación de una empresa con el objetivo de obtener una participación significativa en el mercado. Una

revisión de los diferentes enfoques y características involucradas en los problemas de CFL se muestra en [273]. Se hacen supuestos principales respecto a las reglas de asignación de los clientes a las instalaciones ubicadas y las características de las instalaciones.

Por un lado, las reglas de asignación de los clientes son un aspecto fundamental que ha sido estudiado desde diferentes perspectivas. Por ejemplo, las reglas binarias y proporcionales se discuten en [43], una regla parcialmente proporcional con un valor radio limite en [385], una regla en forma de S en [297], y una regla bajo incertidumbre en [386]. Por otro lado, las instalaciones con capacidad se consideran en [254], y su atractivo ha sido examinado en [188]. Otros aspectos, como la fijación de precios, también se han tenido en cuenta (ver [300]).

Para nuestro estudio consideramos un problema de CFL discreto que sigue la regla binaria basada en los criterios de asignación más económica, lo cual implica asignar clientes a la instalación abierta con el menor costo de asignación. Específicamente, examinamos dos empresas competidoras que operan en el mismo mercado. Una empresa tiene una posición superior y busca ubicar estratégicamente sus instalaciones para captar clientes y maximizar su beneficio. Mientras tanto, la otra empresa, aunque también busca captar clientes y maximizar su beneficio al mismo tiempo busca maximizar la distancia mínima entre sus instalaciones. Esto se hace para evitar que la demanda capturada se divida entre sus instalaciones.

Tradicionalmente, los problemas de CFL se han estudiado como problemas de binivel, donde la jerarquía entre las empresas involucradas es un aspecto crucial para seguir un enfoque de optimización binivel. Sin embargo, la mayoría de los trabajos anteriores se han centrado en problemas de un solo objetivo en cada nivel jerárquico. El único artículo que aborda un problema de CFL biobjetivo es [39]. En ese trabajo, se consideran dos objetivos en el nivel superior: la maximización del beneficio garantizado y la probabilidad de que ocurra un escenario rentable. En [39], se mencionan los desafíos de tener múltiples soluciones óptimas en el nivel inferior, dado que el nivel superior considera dos funciones objetivo simultáneamente.

Nos encontramos con una situación similar pero distinta. Estamos abordando un problema bi-objetivo en el nivel inferior. Si tener un problema bi-objetivo en el nivel superior complica los enfoques clásicos optimistas y pesimistas, entonces considerar un problema bi-objetivo en el nivel inferior introduce aún más incertidumbre. Para ver el impacto de estos enfoques en este problema, a continuación se muestra un modelo de programación binivel multi-objetivo.

5.3 Modelo Matemático

Para modelar la aplicación que nos ayudará a analizar diferentes soluciones factibles binivel, definimos los conjuntos, parámetros y variables de decisión involucrados.

Sea I el conjunto de sitios potenciales para ubicar las instalaciones, y sea J el conjunto de clientes.Consideremos c_{ij} como el costo de asignar un cliente $j \in J$ a una instalación $i \in I$. También consideremos d_{ik} como la distancia desde una instalación $i \in I$ a un nodo k, que puede ser una instalación (es decir, $k \in I$) o un cliente (es decir, $k \in J$). Hay un valor s que indica la distancia máxima que un cliente puede estar de una instalación para ser asignado a ella. Sea w_j el ingreso asociado con satisfacer la demanda de un cliente $j \in J$. La empresa asociada con el nivel superior debe ubicar r instalaciones, mientras que la empresa asociada con el nivel inferior ubica p instalaciones. También utilizamos una constante positiva y suficientemente grande M.

Las variables de decisión se denotan por x_i y y_i, para todos $i \in I$, para el nivel superior e inferior, respectivamente. Las variables de decisión del nivel superior son:

- $x_i = \begin{cases} 1 & \text{si la instalación es localizada en el sitio potencial } i \in I \\ 0 & \text{otherwise} \end{cases}$

mientras que las variables de decisión del nivel inferior son:

- $y_i = \begin{cases} 1 & \text{si la instalación es localizada en el sitio potencial } i \in I \\ 0 & \text{de lo contrario} \end{cases}$

Se añaden conjuntos auxiliares $I(j)$ para identificar el índice de instalaciones que cubren a cada cliente $j \in J$.

$$I(j) = \{i \in I | d_{ij} \leq s\} \qquad \forall j \in J.$$

Adicionalmente, para identificar si un cliente es atendido por una instalación abierta de la empresa del nivel superior, consideramos las siguientes variables auxiliares:

- $z_{ij}^L = \begin{cases} 1 & \text{si el cliente } j \in J \text{ está asignado a la instalación } i \in I \\ 0 & \text{de lo contrario} \end{cases}$

De manera similar, se incluyen variables auxiliares asociadas a la asignación de clientes a la empresa del nivel inferior:

- $z_{ij}^F = \begin{cases} 1 & \text{si el cliente } j \in J \text{ está asignado a la instalación } i \in I \\ 0 & \text{de lo contrario} \end{cases}$

Cabe destacar que las variables auxiliares pueden identificarse aplicando la regla de asignación más económica asumiendo el radio de cobertura. Es decir, una vez que las empresas de los niveles superior e inferior han abierto sus instalaciones, la asignación de los clientes puede realizarse de manera directa.

El modelo resultante de programación binivel, en el que hay un único objetivo en el nivel superior y dos objetivos en el nivel inferior, se presenta a continuación:

$$\max_x \sum_{i\in I(j)} \sum_{j\in J} (w_j - c_{ij}) z_{ij}^L \tag{5.10}$$

$$\text{s.t} \quad \sum_{i\in I} x_i = r \tag{5.11}$$

$$x_i \in \{0,1\} \qquad \forall i \in I \tag{5.12}$$

donde para una variable x fija, las variables y, z^F y z^L deben ser la solución óptima del siguiente problema bi-objetivo:

$$\max_y \quad \sum_{i\in I(j)} \sum_{j\in J} (w_j - c_{ij}) z_{ij}^F \tag{5.13}$$

$$\max_{i,i'\in I, i<i'} \min \quad \{d_{ii'} y_i y_{i'}\} \tag{5.14}$$

$$\text{s.t.} \quad \sum_{i\in I} y_i = p \tag{5.15}$$

$$x_i + y_i \leq 1 \qquad \forall i \in I \tag{5.16}$$

$$z_{ij}^F \leq y_i \qquad \forall i \in I(j), \forall j \in J \tag{5.17}$$

$$z_{ij}^L \leq x_i \qquad \forall i \in I(j), \forall j \in J \tag{5.18}$$

$$\sum_{i\in I(j)} z_{ij}^L + z_{ij}^F \leq 1 \qquad \forall j \in J \tag{5.19}$$

$$\sum_{a\in I} c_{aj}(z_{aj}^L + z_{aj}^F) + (M - c_{ij})(x_i + y_i) \leq M \qquad \forall i \in I(j), \forall j \in J \tag{5.20}$$

$$z_{ij}^L, z_{ij}^F, y_i \in \{0,1\} \qquad \forall i \in I(j), j \in J \tag{5.21}$$

La Ecuación (5.10) representa la función objetivo del nivel superior, que busca maximizar su beneficio (ingresos menos costos de asignación) de los clientes asignados a sus instalaciones abiertas. La ecuación (5.11) asegura el requisito de ubicar exactamente r instalaciones, y la ecuación (5.12) indica la naturaleza binaria de las variables de decisión. Es importante destacar que hay una declaración que indica claramente que las variables asociadas con el nivel inferior y las variables auxiliares se obtienen resolviendo óptimamente el problema descrito por las Ecuaciones (5.13)- (5.21).

Note que el nivel inferior considera dos funciones objetivo de manera simultánea. Por un lado, la función objetivo, denotada por la ecuación (5.13) busca maximizar su propio ingreso. Por otro lado, la ecuación (5.14) busca maximizar la mínima distancia entre cada par de instalaciones abiertas. La ecuación (5.15) se asegura que solo p instalaciones sean ubicadas. Ecuación (5.16) garantiza que solo una de las instalaciones del nivel superior o del nivel inferior pueda ubicarse en cada sitio potencial, pero no ambas. Las ecuaciones (5.17) y (5.18) permiten la asignación de clientes solo a instalaciones que los cubren. La ecuación (5.19) se asegura que los clientes solo pueden ser asignados a una única instalación dentro de sus radios de cobertura. Por otro lado, la ecuación (5.20) impone la regla de asignación más económica considerando el radio de cobertura. Finalmente (5.21) indica la naturaleza binaria de las variables de decisión y auxiliares del nivel inferior.

5.4 Multiples enfoques para programación binivel multi-objetivo

En esta sección, mostraremos el impacto que conlleva asumir distintos enfoques como los ya mencionado, para ello se mostrará un ejemplo ilustrativo y se discutirán los resultados.

Para analizar soluciones binivel factibles para la aplicación discutida, debemos seguir dos pasos. En primer lugar, necesitamos definir el enfoque para obtener soluciones de nivel superior. En segundo lugar, debemos especificar la metodología utilizada para identificar el frente de Pareto para el problema de nivel inferior parametrizado.

Después de obtener una solución del nivel superior y la respuesta del nivel inferior, es necesario hacer ciertas suposiciones para seleccionar una sola solución del frente de Pareto obtenido. Es evidente que habrá un impacto potencialmente impredecible en la ganancia del nivel superior. En este caso, se consideran varias suposiciones al analizar las soluciones factibles en binivel.

5.4.1 Construyendo soluciones del nivel superior

En nuestro modelo la única restricción en el problema del nivel superior es que se debe ubicar un cierto número de instalaciones. Por lo tanto, una representación de solución directa puede identificarse mediante un vector binario de tamaño $|I|$, donde 1 representa si una instalación ha sido ubicada y 0 en el caso en que la instalación permanezca cerrada.

Para obtener soluciones del nivel superior, se considera un enfoque enumerativo. Este algoritmo implica identificar todas las posibles combinaciones de r instalaciones ubicadas entre los $|I|$ sitios potenciales. Teniendo en cuenta que este enfoque solo es adecuado para instancias de tamaño muy limitado.

5.4.2 Resolviendo el problema del nivel inferior.

Recordando que para obtener soluciones factibles para un problema en binivel, dada una solución del nivel superior, implica que el nivel inferior debe resolverse de manera óptima. Sin embargo, dado que el nivel inferior es un problema bi-objetivo, se obtiene un frente de Pareto al resolverlo. Para encontrar este frente de pareto se deberá hacer algunos pasos previos para resolverlo.

Dada una solución del nivel superior x. En este caso, el problema bi-objetivo de nivel inferior, definido por las Ecuaciones (5.13)-(5.21), está parametrizado en función de x. Se puede observar que la segunda función objetivo del nivel inferior es no lineal, como se ve en la Ecuación (5.14). No obstante, puede ser fácilmente linealizada. Para este propósito, introducimos una variable auxiliar ρ y una nueva constante positiva y suficientemente grande M'. Adicionalmente, necesitamos incluir el siguiente conjunto de restricciones de acotamiento:

$$\rho \leq M'(2 - y_i - y_k) + d_{ik} \qquad \forall i, k \in I \tag{5.22}$$

El modelo del nivel inferior linealizado se presenta a continuación:

$$\max_{y, z^L, z^F} \quad \sum_{i \in I(j)} \sum_{j \in J} (w_j - c_{ij}) z_{ij}^F \tag{5.23}$$

$$\text{máx} \quad \rho \tag{5.24}$$

$$\text{s.t.} \qquad \rho \leq M'(2 - y_i - y_k) + d_{ik} \qquad \forall i, k \in I \tag{5.25}$$

$$\sum_{i \in I} y_i = p \tag{5.26}$$

$$x_i + y_i \leq 1 \qquad \forall i \in I \tag{5.27}$$

$$z_{ij}^F \leq y_i \qquad \forall i \in I(j), \forall j \in J \tag{5.28}$$

$$z_{ij}^L \leq x_i \qquad \forall i \in I(j), \forall j \in J \tag{5.29}$$

$$\sum_{i \in I(j)} z_{ij}^L + z_{ij}^F \leq 1 \qquad \forall j \in J \tag{5.30}$$

$$\sum_{a \in I} c_{aj}(z_{aj}^L + z_{aj}^F) + (M - c_{ij})(x_i + y_i) \leq M \qquad \forall i \in I(j), \forall j \in J \tag{5.31}$$

$$z_{ij}^L, z_{ij}^F, y_i \in \{0, 1\} \qquad \forall i \in I, j \in J \tag{5.32}$$

Ahora, se puede observar que la segunda función objetivo del nivel inferior es lineal, como se ve en la Ecuación (5.24). Por lo tanto, se pueden implementar métodos exactos clásicos para encontrar el frente de Pareto del este problema de nivel inferior parametrizado en x.

5.4.3 Un método ε-restricción.

Para obtener el frente de Pareto del problema de nivel inferior parametrizado y linealizado, se desarrolla un método ε-restricción. Este método consiste en transformar un problema bi-objetivo en uno de un solo objetivo agregando una de las funciones objetivo como restricción. La restricción agregada impone un límite ε al valor de la función objetivo que se considera como restricción. El método itera cambiando el valor de ε y termina cuando ε alcanza el valor máximo permitido. En este método, es crucial tener bien definidos los tamaños de los pasos para aumentar ε. En la práctica, es común considerar como restricción la función objetivo para la cual los valores de compensación entre las funciones objetivo pueden obtenerse analíticamente.

En la aplicación bajo estudio, consideramos la segunda función objetivo, la Ecuación (5.24), como una restricción. Esta decisión se basa en el hecho de que podemos calcular las distancias entre las instalaciones. Dado que la Ecuación (5.24) tiene como objetivo maximizar la distancia mínima entre las instalaciones ubicadas, los valores de ε se seleccionan del conjunto que contiene todas las distancias entre las instalaciones potenciales que no son ubicadas por el tomador de decisión del nivel superior. Este enfoque nos permite explorar los valores de compensación entre las funciones objetivo.

El problema de un solo objetivo resultante que se resolverá iterativamente utilizando el método de ε-restricción se presenta a continuación:

$$\max_{y,z^L,z^F} \quad \sum_{i\in I(j)}\sum_{j\in J}(w_j - c_{ij})z_{ij}^F \tag{5.33}$$

$$\text{s.t.} \quad \rho \leq M'(2 - y_i - y_k) + d_{ik} \qquad \forall i,k \in I \tag{5.34}$$

$$\rho \geq \varepsilon \tag{5.35}$$

$$\sum_{i\in I} y_i = p \tag{5.36}$$

$$x_i + y_i \leq 1 \qquad \forall i \in I \tag{5.37}$$

$$z_{ij}^F \leq y_i \qquad \forall i \in I(j), \forall j \in J \tag{5.38}$$

$$z_{ij}^L \leq x_i \qquad \forall i \in I(j), \forall j \in J \tag{5.39}$$

$$\sum_{i\in I(j)} z_{ij}^L + z_{ij}^F \leq 1 \qquad \forall j \in J \tag{5.40}$$

$$\sum_{a\in I} c_{aj}(z_{aj}^L + z_{aj}^F) + (M - c_{ij})(x_i + y_i) \leq M \qquad \forall i \in I(j), \forall j \in J \tag{5.41}$$

$$z_{ij}^L, z_{ij}^F, y_i \in \{0,1\} \qquad \forall i \in I, j \in J \tag{5.42}$$

Cabe mencionar que la restricción (5.35) es la parte fundamental del método de ε-restricción. El esquema de este método se describe en el Algoritmo 5.4.1.

Algoritmo 5.4.1: Método ε-restricción propuesto

Input: x (an upper level solution)

$I^L = \{i \in I : x_i = 1\}$ `/* Set of facilities open in the upper level */`

$D = \{d_{ii'} :$ distances from location $i \in I \setminus I^L$ to location $i' \in I \setminus I^L, i \neq i'\}$

PF $= \emptyset$

for $\varepsilon \in D$ **do**

  $\varepsilon = d_{ii'}$

  if $\max_y \{(5.33) : x$ *and* $(5.34) - (5.42)\}$ *is feasible* **then**

    $(y^*, z^{L*}, z^{F*}) \leftarrow \max_y \{(5.33) : x$ and $(5.34) - (5.42)\}$

    Compute lower-level's profit $f_1(z^{F*})$ `/* According to Eq. (5.13) */`

    Compute lower-level's distance $f_2(y^*)$ `/* According to Eq. (5.14) */`

    Compute upper-level's profit $F(z^{L*})$ `/* According to Eq. (5.10) */`

  $PF \leftarrow PF \cup (f_1, f_2)$

return Pareto front

Para asegurar que se obtengan los extremos del frente de Pareto, utilizamos un método de penalización sutil en el que se incorporan las variables auxiliares z_{ij}^L y z_{ij}^F en la función objetivo. Es importante destacar que el método ε-restricción puede producir soluciones débilmente eficientes. Sin embargo, debido a la estructura del problema, estas soluciones se clasifican como soluciones débilmente eficientes desechables (ver [245]). En términos más simples, para diferentes distancias, resultan en el mismo beneficio de nivel inferior. Además, el beneficio asociado con el nivel superior permanece constante. En consecuencia, estas soluciones débilmente eficientes no son de interés en nuestro problema. Por lo tanto, al implementar el método de restricción ε descrito para una solución de nivel superior dada, obtenemos el frente de Pareto de nivel inferior, que está compuesto por soluciones eficientes.

5.4.4 Un ejemplo Ilustrativo

Para ilustrar los desafíos de clasificar y analizar las soluciones factibles en binivel, consideramos un ejemplo.

Ejemplo: *Consideremos 10 sitios potenciales, 24 clientes, y supongamos que ambos competidores tienen como objetivo abrir dos instalaciones, es decir,* $r = p = 2$. *La Figura 5.1 ilustra la distribución espacial de los sitios potenciales y los clientes.*

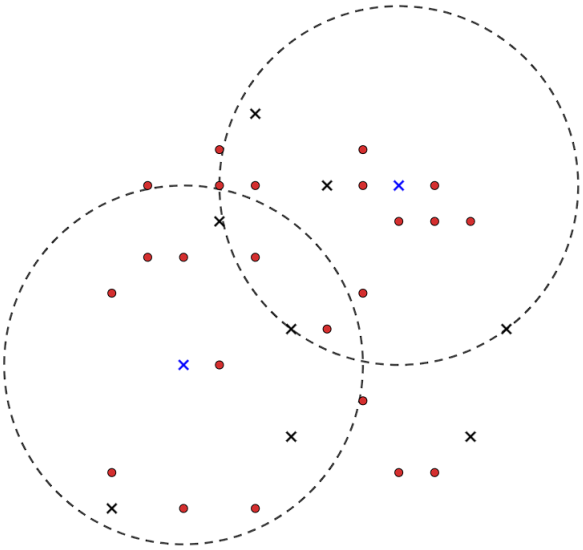


Figura 5.1: Distribución espacial del ejemplo ilustrativo

En la Figura 5.1, las decisiones específicas de nivel superior se indican con cruces azules, representando las instalaciones abiertas por el tomador de decisiones del nivel superior. Las cruces negras representan los sitios potenciales donde el tomador de decisiones del nivel inferior ubicará dos instalaciones. Los puntos rojos representan a los clientes, y se muestra el radio de cobertura de las dos instalaciones ubicadas por el tomador de decisiones del nivel superior. Es importante mencionar que todas las instalaciones tienen su propio radio de cobertura, pero por claridad visual, no todas están representadas.

Como se mencionó en la Sección 5.4.1, dado el tamaño del problema, exploramos todas las combinaciones posibles de dos instalaciones entre los 10 sitios potenciales diferentes. En este caso, se pueden considerar 45 soluciones del nivel superior. Para cada una de estas soluciones, se resuelve el nivel inferior utilizando el método de ε-restricción, como se describe en la Sección 5.4.2, para identificar el frente de Pareto correspondiente. En resumen, el problema en binivel se resuelve utilizando un algoritmo exhaustivo para el problema del nivel superior y, de manera anidada, el método de ε-restricción para el nivel inferior.

Para mostrar los resultados se identificaron cuatro de las 45 soluciones del nivel superior para identificar explícitamente sus frentes de Pareto asociados. Para cada una de estas cuatro soluciones, presentamos los valores de las funciones objetivo para las soluciones del nivel inferior no dominadas. Esto incluye las instalaciones ubicadas por el tomador de decisiones del nivel inferior y sus valores de función objetivo asociados, es

decir, la distancia máxima entre las instalaciones ubicadas por el nivel inferior y el beneficio correspondiente. Además, mostramos el beneficio del nivel superior asociado con cada solución específica de nivel inferior no dominada.

Solución A: Para el caso en que el tomador de decisiones de nivel superior ubica las instalaciones 1 y 3, los resultados obtenidos para el problema del nivel inferior se presentan en la Tabla 5.1. La Figura 5.2 muestra el frente de Pareto obtenido por el tomador de decisiones del nivel inferior para esta solución del nivel superior en azul. Además, el beneficio para el nivel superior se muestra en amarillo.

	Nivel Inferior		Nivel Superior
Instalaciones ubicadas	Distancia Max	Ganancia	Ganancia
{6,7}	6.4	3341.6	1700.6
{2,7}	7.81	3319.1	1516.5
{4,5}	8.54	2902.9	2267.4
{5,10}	9.22	2652.4	1818.1
{4,6}	10.82	2641.3	2325.4
{2,4}	12.04	2518.8	2141.29

Tabla 5.1: Resultados para la solución del nivel superior {1,3}.

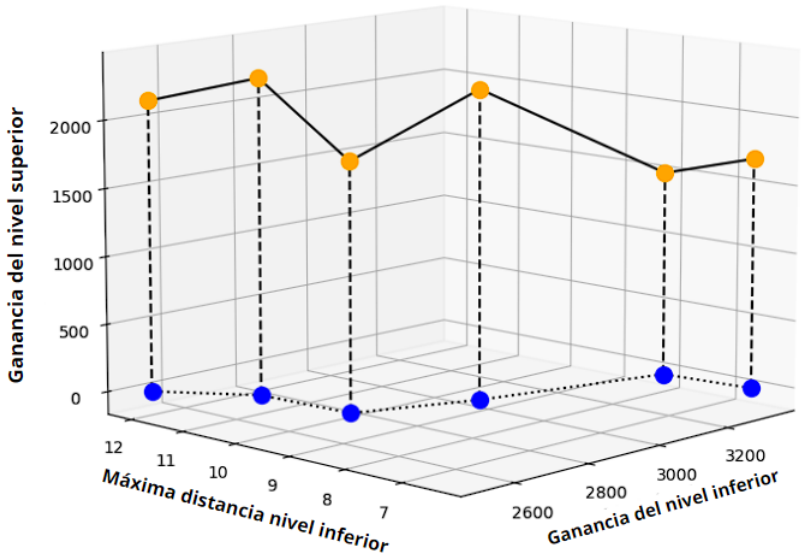


Figura 5.2: Reacción del nivel inferior y ganancia para el nivel superior para la solución A.

A partir de los resultados obtenidos para esta solución, se puede notar que, a medida que aumenta la distancia máxima, el beneficio del nivel inferior disminuye. Este comporta-

miento es esperado debido al conflicto inherente entre los dos objetivos en el nivel inferior. Sin embargo, es importante señalar que, cuando el beneficio del nivel inferior disminuye, el beneficio del nivel superior no necesariamente aumenta. En general, no hay una correlación directa entre el impacto en los beneficios de ambos tomadores de decisiones.

Solución B: Ahora considere que el tomador de decisiones de nivel superior ubica las instalaciones 2 y 9. Los resultados obtenidos se presentan en la Tabla 5.2 y se muestran en la Figura 5.3.

	Nivel Inferior		Nivel Superior
Instalaciones ubicadas	Distancia Max	Ganancia	Ganancia
{5,8}	6.32	3136.2	1938.1
{6,8}	7.07	2793.7	1989.3
{4,5}	8.54	2702.4	1938.1
{5,10}	9.22	2654.3	1938.1
{4,6}	10.82	2352.3	1989.3
{3,4}	12.08	1561.6	3402.2

Tabla 5.2: Resultados para la solución del nivel superior {2,9}.

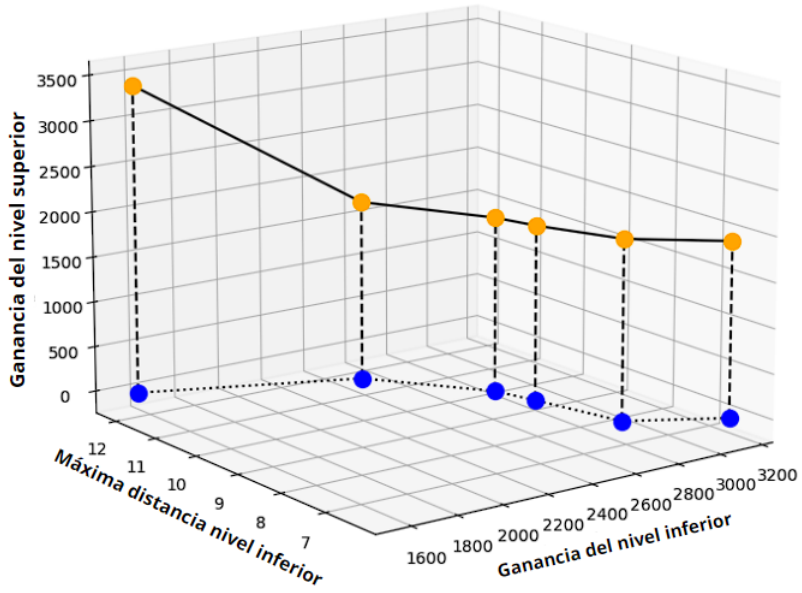


Figura 5.3: Reacción del nivel inferior y ganancia para el nivel superior para la solución B.

Para la solución B, el beneficio del nivel superior se mantiene constante para diferentes soluciones no dominadas del nivel inferior asociadas con beneficios variables, como

se observa en los casos en que se ubican las instalaciones 4,5 y 5,10. El beneficio del nivel superior permanece en 1938.1.

Solución C: Ahora, asumamos que el tomador de decisiones de nivel superior ubica las instalaciones 3 y 7. Los detalles de los resultados obtenidos se presentan en la Tabla 5.3 y se visualizan en la Figura 5.4.

	Nivel Inferior		Nivel Superior
Instalaciones ubicadas	Distancia Max	Ganancia	Ganancia
{6,8}	7.07	3630.1	1529.4
{2,8}	7.62	3555.9	1283.9
{2,4}	10.82	3210.4	1844.4
{2,4}	12.04	2966.2	1844.4

Tabla 5.3: Resultados para la solución del nivel superior {3,7}.

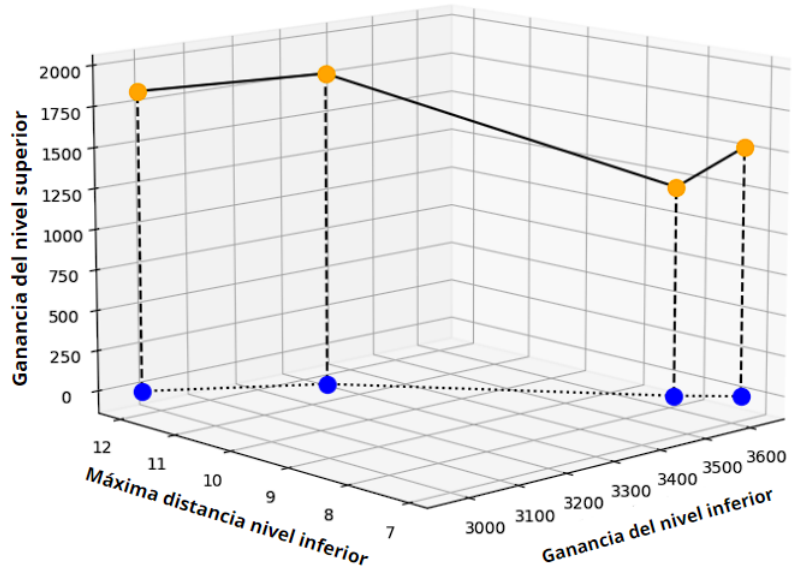


Figura 5.4: Reacción del nivel inferior y ganancia para el nivel superior para la solución C.

Se puede observar que el frente de Pareto obtenido en el nivel inferior, cuando el tomador de decisiones de nivel superior ubica las instalaciones 3 y 7, contiene menos soluciones que los ejemplos anteriores. Es racional que los respectivos frentes de Pareto para cada solución de nivel superior tengan un número diferente de soluciones no dominadas. Similar al ejemplo anterior (solución B), diferentes instalaciones ubicadas en el nivel superior pueden llevar al mismo beneficio del nivel superior. Además, a medida que el beneficio

del nivel inferior disminuye, el beneficio del nivel superior puede aumentar, disminuir o permanecer igual (ver Figura 5.4).

Solución D: Por último, consideremos el caso en el que se ubican las instalaciones 3 y 8. Los resultados obtenidos se informan en la Tabla 5.4 y se visualizan en la Figura 5.5.

	Nivel Inferior		Nivel Superior
Instalaciones ubicadas	Distancia Max	Ganancia	Ganancia
{2,5}	5.10	3534.0	1534.4
{2,7}	7.81	3492.1	1347.7
{4,6}	10.82	3010.7	1727.9
{2,4}	12.04	2912.5	1296.5

Tabla 5.4: Resultados para la solución del nivel superior {3,8}.

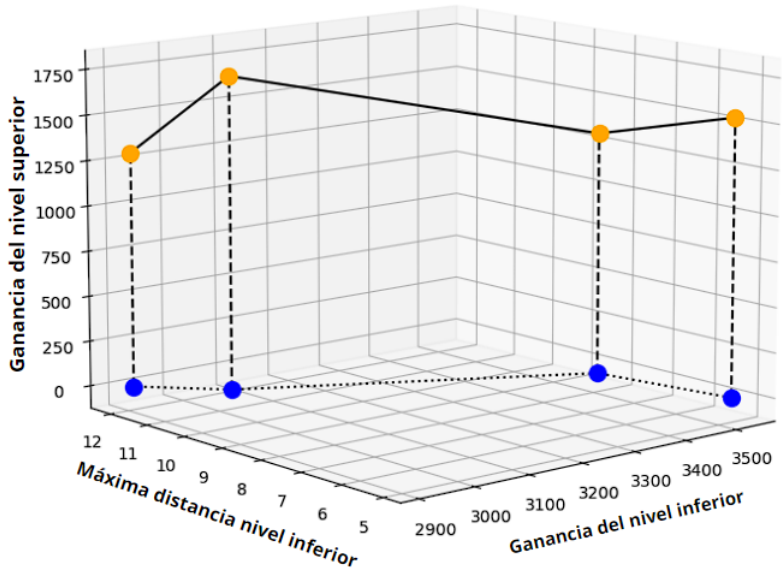


Figura 5.5: Reacción del nivel inferior y ganancia para el nivel superior para la solución D.

Similar a los casos anteriores, el comportamiento del beneficio del nivel superior no sigue una tendencia consistente cuando se ubican diferentes instalaciones en el nivel inferior. En otras palabras, si la distancia máxima de las instalaciones del nivel inferior aumenta, entonces el beneficio del nivel superior puede aumentar o permanecer sin cambios. La misma situación ocurre con el beneficio del nivel inferior. La función objetivo del nivel superior puede variar sin un patrón discernible a medida que se consideran diferentes soluciones no dominadas del nivel inferior.

Se sigue el mismo procedimiento para evaluar las otras 41 soluciones factibles del nivel superior restantes. Una ilustración del beneficio del nivel superior asociado con cada una de las soluciones no dominadas para todas las soluciones del nivel superior se muestra en la Figura 5.6. Vale la pena señalar que el número de soluciones no dominadas para cada solución del nivel superior puede variar, lo que resulta en un número diferente de soluciones en el frente de Pareto del nivel inferior.

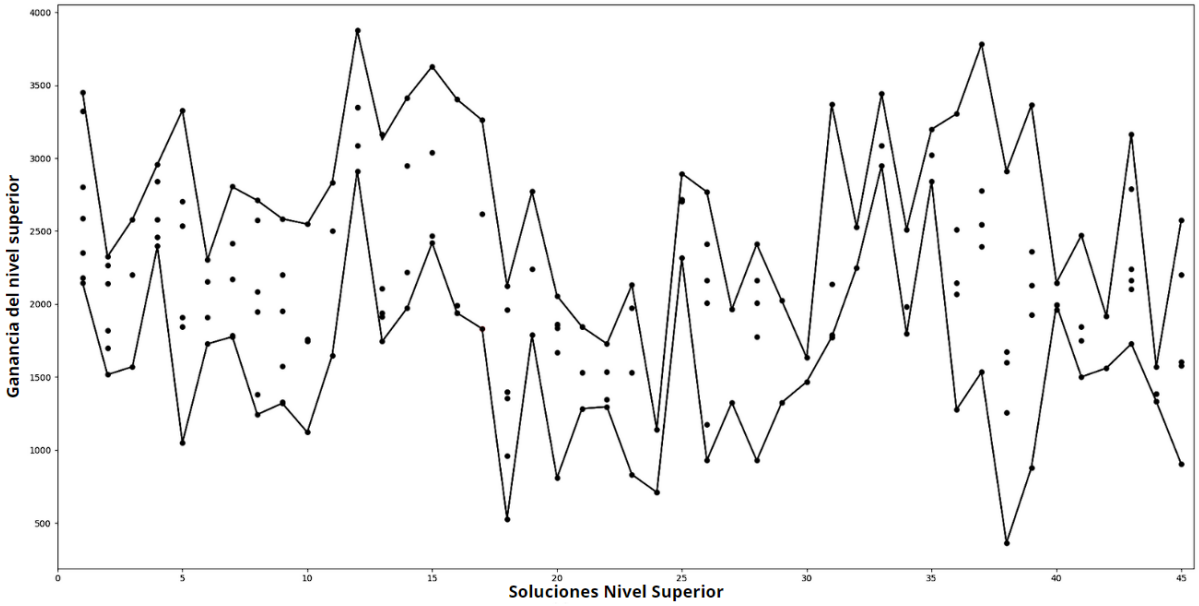


Figura 5.6: La ganancia del nivel superior varia para cada solución no dominada obtenida en el nivel inferior, dependiendo de la solución del nivel superior.

Basándonos en los conceptos mostrados en [404] y [231], adaptamos el método del k-ésimo vecino más cercano. Esta métrica implica identificar las k soluciones no dominadas más cercanas y calcular la distancia entre ellas. Los valores más pequeños indican una región del frente de Pareto con una mayor densidad, mientras que los valores más grandes implican una región más dispersa. Dado que, para el problema en estudio, hay un número relativamente pequeño de soluciones no dominadas en cada frente de Pareto, usamos $k = 2$, lo que significa que calculamos la distancia entre una solución y sus dos vecinos más cercanos. Para examinar a fondo esta métrica, evaluamos tanto la distancia mínima del k-ésimo para obtener el enfoque neutral como la distancia máxima del k-ésimo para obtener el enfoque incómodo. Es importante destacar que estas dos métricas nunca se han aplicado en problemas binivel con un problema bi-objetivo en el nivel inferior. Además, es importante tener en cuenta que, para una mejor precisión en la búsqueda de la solución óptima del nivel inferior, las funciones objetivo asociadas con cada solución del nivel inferior se estandarizan en un rango entre 0 y 1. Esta estandarización es crucial

para calcular correctamente las distancias entre soluciones.

Por lo tanto, presentamos un resumen de los resultados obtenidos aplicando los siete enfoques diferentes mencionados anteriormente (optimista, engañosa, pesimista, recompensante, la mejor comprometida, neutral e incómodo). Nuestra metodología implica una exploración exhaustiva de las soluciones del nivel superior. Para cada una de estas soluciones, empleamos el método de la ε-restricción descrito en la Sección 5.4.3 para resolver el problema del nivel inferior. Posteriormente, para cada frente de Pareto resultante, seleccionamos las soluciones no dominadas del nivel inferior correspondientes a cada uno de los enfoques asumidos como la reacción del tomador de decisiones del nivel inferior. Los resultados numéricos están disponibles en la Tabla 5.5, donde: F representa el beneficio del nivel superior, x indica las instalaciones ubicadas por el tomador de decisiones del nivel superior, f_1 y f_2 denotan las funciones objetivo del nivel inferior, es decir, el beneficio y la distancia máxima entre las instalaciones ubicadas, respectivamente, y y indica las instalaciones ubicadas en el nivel inferior.

Assumed approach	F	x	f_1	f_2	y
Optimistic	3876.6	{2,5}	1396.6	12.8	{3,4}
Deceiving	2911.7	{2,5}	2156.7	7.07	{6,8}
Pessimistic	2950.6	{5,8}	2390.8	7.81	{2,7}
Rewarding	3441.4	{5,8}	1728.0	12.08	{3,4}
Best compromise	3086.3	{5,8}	2271.2	12.04	{2,4}
Neutral	1380.1	{1,9}	3056.3	8.54	{3,5}
Awkward	3086.3	{5,8}	2271.2	12.04	{2,4}

Tabla 5.5: Resultados obtenidos al resolver el ejemplo ilustrativo bajo cada enfoque asumido

De la Tabla 5.5, se puede observar que solo dos enfoques arrojan el mismo beneficio óptimo del nivel superior: la mejor comprometida y el incómodo. Todos los demás enfoques resultan en soluciones "óptimas"diferentes para el problema. Como se esperaba, el enfoque optimista, que selecciona la solución del nivel inferior que conduce al mayor beneficio del nivel superior, logra el mayor beneficio del nivel superior. Sin embargo, de manera contraintuitiva, el enfoque asociado con el menor beneficio del nivel superior no es el pesimista, sino el neutral. La razón de esto es que el enfoque pesimista elige el beneficio máximo entre todos los mínimos, mientras que el enfoque neutral, que considera la distancia mínima de 2, devuelve la solución no dominada más densa entre todas las no dominadas como la reacción del tomador de decisiones del nivel inferior. También, notemos que el enfoque engañoso conduce a un beneficio del nivel superior peor que el

enfoque pesimista.

Para ilustrar este comportamiento, la Figura 5.7 presenta los resultados basados en los siete enfoques diferentes. Se han seleccionado tres soluciones específicas del nivel superior para ayudar en la ilustración, y se incluye una imagen ampliada en la figura.

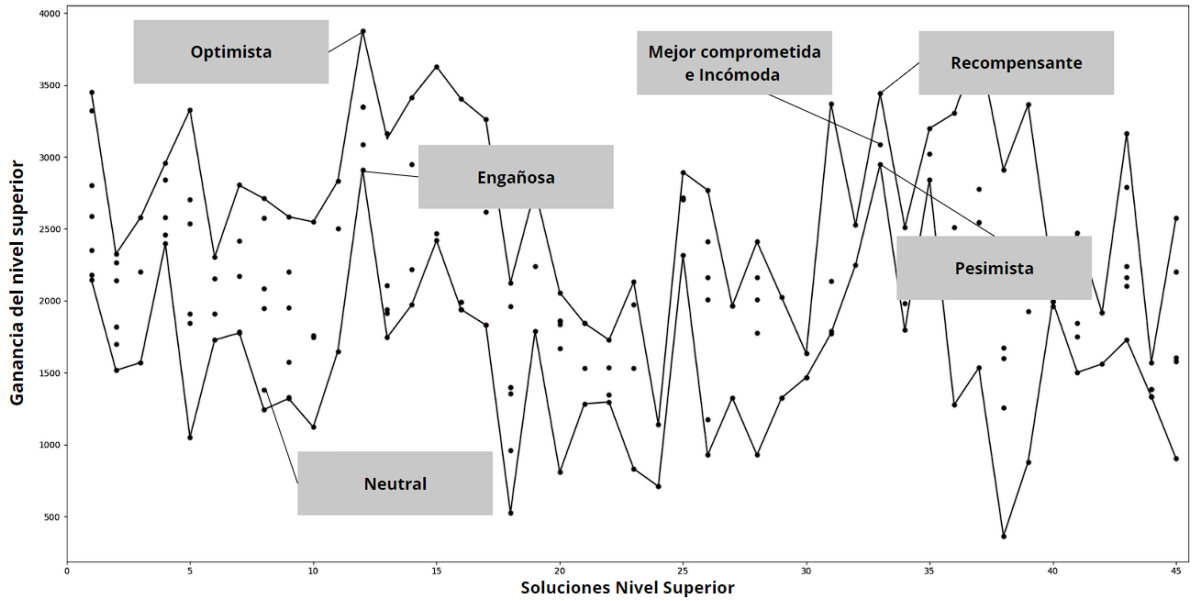


Figura 5.7: Soluciones optimas para cada enfoque asumido.

Es crucial enfatizar la importancia del enfoque elegido para seleccionar la reacción del nivel inferior. Cuando se emplea un enfoque aleatorio, los resultados se vuelven no reproducibles. Por ejemplo, considera el ejemplo específico representado en la Figura 5.6. Supongamos que se sigue un enfoque aleatorio, donde para cada solución del nivel superior, la reacción del tomador de decisiones del nivel inferior se selecciona aleatoriamente de entre las soluciones no dominadas correspondientes. Se vuelve evidente que este enfoque aleatorio carece de reproducibilidad, ya que puede arrojar resultados diferentes cada vez que se resuelve el problema. Además, la "solución óptima"bajo el enfoque aleatorio puede diferir significativamente de los resultados obtenidos utilizando otros enfoques estructurados.

5.5 Un algoritmo evolutivo anidado para el problema de localización de instalaciones competitivas: ε-restricción vs NSGA-II

Si el tamaño del problema es pequeño, un método que enumere las soluciones del nivel superior es viable, sin embargo, debido a la naturaleza combinatoria del problema, conforme el tamaño del problema aumente, el problema se vuelve más complicado para resolverlo. Para ello en esta sección mostraremos un esquema de solución para el problema a través de dos enfoques: el enfoque optimista y el enfoque pesimista.

Como se mencionó anteriormente, los algoritmos evolutivos (EAs, por sus siglas en inglés) han demostrado ser útiles al tratar con problemas binivel. El EA propuesto adopta un enfoque anidado (ver [71]), es decir, para cada una de las soluciones generadas por el líder, se resuelve el problema del seguidor.

Es por ello que proponemos dos algoritmos anidados distintos basándonos en los enfoques optimista y pesimista. El algoritmo evolutivo anidado optimista (ONEA por sus siglas en inglés) y el algoritmo evolutivo anidado pesimista (PNEA también por sus siglas en inglés) Es importante mencionar que ambos enfoques son independientes de uno del otro, es decir, tanto el ONEA como el PNEA se usarán para resolver el CFL propuesto. Un diagrama de flujo que ilustra la idea general de los algoritmos propuesto se presenta en la figura 5.8.

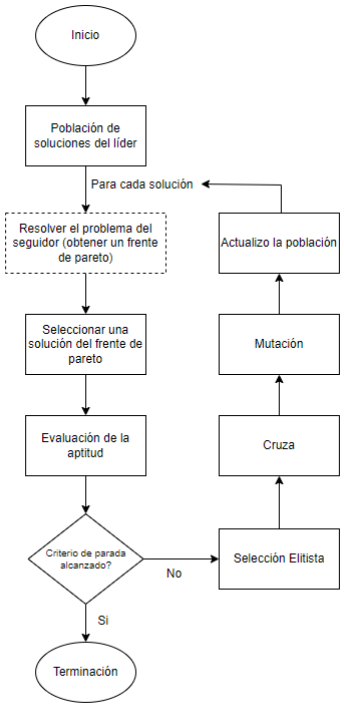


Figura 5.8: Esquema general de los algoritmos propuestos.

A continuación se muestra algunos de los componentes que conforman los algoritmos utilizados.

5.5.1 Población Inicial

Dado que el líder tiene como objetivo abrir r instalaciones, una solución puede ser representada como un vector binario de tamaño $|I|$, donde 1 indica que una instalación está abierta y 0 en caso contrario.

Para inicializar el NEA propuesto, se necesita un conjunto inicial de soluciones (población). P soluciones factibles son creadas generando aleatoriamente una permutación de los números naturales del 1 al $|I|$. Considere estos números como las posiciones de las instalaciones abiertas en el vector binario.

5.5.2 Resolviendo el problema del seguidor

Para nuestro problema, utilizamos la metodología propuesta en la sección 5.4.3, primero se linealiza el modelo estudiado y luego se utiliza un algoritmo ε-restricción para el problema bi-objetivo asociado al seguidor.

5.5.3 Reacción del seguidor

Como se mencionó anteriormente, estamos considerando tanto enfoques optimistas como pesimistas, lo que conduce a diferentes algoritmos: el ONEA y el PNEA, respectivamente. Para cada solución del líder, cada versión del NEA propuesto se comporta de la siguiente manera:

ONEA: Del frente de pareto, el seguidor selecciona la solución que resulta en mayores ingresos para el líder.

PNEA: Del frente de pareto, el seguidor identifica la solución que resulta en menores ingresos para el líder.

Es importante enfatizar que al final del ONEA, se elige la solución que mejor se adapta al líder. En el PNEA, el proceso es ligeramente diferente. Entre todas las peores soluciones del líder, se selecciona la que sea mejor para el líder [19].

5.5.4 Selección

Implementamos un proceso de selección elitista que incorpora un cierto grado de diversidad. Para emparejar soluciones, empleamos una estrategia de torneo. Para las soluciones mejor clasificadas, emparejamos aleatoriamente cada una con otra solución de la población y comparamos sus funciones objetivo del líder. La solución que rinde mejor se declara ganadora y su puntuación se incrementa en uno. En cada torneo, todas las puntuaciones se inicializan en cero. Una vez que se completa el proceso de emparejamiento, se concluye un torneo. Este procedimiento se repite durante un número predeterminado de torneos, denotado como T.

Siguiendo este esquema, permitimos que soluciones que no están en el subconjunto elitista sean consideradas para generar nuevas soluciones. Sin embargo, es importante notar que las soluciones mejor clasificadas tienen más probabilidades de ser usadas en la generación de nuevas soluciones debido a sus mayores chances de ganar partidas.

Una vez completados los torneos, se emparejan pares de soluciones para generar nuevas. Seleccionamos la solución mejor clasificada y elegimos aleatoriamente otra solución de la población. Para cada par de soluciones (en adelante los padres), se aplica un operador de cruce para generar nuevas soluciones.

5.5.5 Operador Cruza

En este operador, dos nuevas soluciones se generan aleatoriamente a partir de dos soluciones existentes (en adelante, denominadas descendientes). La probabilidad de entrar en este operador está definida por p_c. Empleamos un cruce de dos puntos bien conocido, donde se seleccionan aleatoriamente dos posiciones de los padres. Esta división da como resultado tres partes de una solución. Los componentes del primer padre en los extremos se copian al primer descendiente, y los componentes en el medio del segundo padre se copian al mismo primer descendiente. El segundo descendiente se genera de manera similar pero opuesta. Una ilustración de este procedimiento se muestra en la Figura 5.9 para ejemplificar el proceso.

Figura 5.9: Ilustración del operador de cruza

Los descendientes pueden llevar a soluciones infactibles debido a la restricción de que exactamente r instalaciones deben estar abiertas. Para abordar este problema, se implementa una fase de reparación. Si la solución infactible excede el número de instalaciones que deben abrirse, entonces cerramos instalaciones aleatoriamente hasta que solo queden r abiertas. Por el contrario, si hay menos de r instalaciones abiertas en la solución infactible, entonces abrimos aleatoriamente las instalaciones faltantes.

Para agregar más diversidad al ONEA y PNEA y además prevenir la convergencia prematura, se incluye un operador de mutación.

5.5.6 Operador de Mutación

Con una cierta probabilidad p_m, el operador de mutación intercambia aleatoriamente un par de instalaciones abiertas y cerradas. Este esquema asegura que se mantenga la viabilidad.

5.5.7 Actualización de la Población

Después de obtener un conjunto de nuevas soluciones factibles de los operadores de cruce y mutación, es necesario actualizar la población. Dado que las soluciones recién generadas no necesariamente ofrecen una mejor aptitud que las existentes, optamos por no realizar un reemplazo completo de toda la población. En su lugar, fusionamos ambas poblaciones (la antigua y la nueva) y seleccionamos las mejores P soluciones basándonos en la aptitud. Este proceso de selección sigue un criterio puramente elitista.

Este proceso se repite durante un número predeterminado de G generaciones (iteraciones) o hasta que ocurran N iteraciones consecutivas sin mejorar la solución con el mejor valor de aptitud. Finalmente, se selecciona la solución con la mejor aptitud, dependiendo del enfoque asumido.

5.6 Resultados

En esta sección presentamos los resultados obtenidos para las instancias del problema. El objetivo es validad el rendimiento de las dos versiones del algoritmo propuesto. Adicionalmente, comparamos los resultados obtenidos por los algoritmos propuestos con los obtenidos al reemplazar el algoritmo ε-restricción con el conocido NSGA-II [112].

Consideramos un conjunto de 10 instancias de dos tamaños diferentes. Las instancias se generan de la siguiente manera: para la ubicación de cada sitio potencial y cada cliente, se genera un punto aleatorio dentro del cuadrado unitario. No se permite la co-ubicación, por lo que cada punto generado debe ser diferente de los demás. El número de instalaciones a abrir se fija, teniendo en cuenta el número de sitios potenciales. Además, sin pérdida de generalidad, asumimos que $p < r$. El tamaño de las instancias se indica por el número de instalaciones potenciales y el número de clientes. Consideramos instancias de tamaño 10×24 y 15×40, etiquetadas como instancias de tamaño pequeño y mediano, respectivamente. En las instancias pequeñas, el líder y el seguidor van a abrir exactamente tres instalaciones cada uno, y cinco en las medianas.

5.6.1 Parámetros del ONEA y PNEA

Basándonos en experimentación preliminar, se seleccionaron los parámetros para ambos algoritmos, ONEA y PNEA. Vale la pena señalar que ambos algoritmos utilizaron la misma configuración de parámetros. El tamaño de la población se fija en 60 soluciones. Realizamos cinco torneos para seleccionar las soluciones que se usarán como padres. Las probabilidades de entrar en los operadores de cruce y mutación son 0.75 y 0.15, respectivamente. El proceso evolutivo se repite hasta 50 generaciones y se detiene si, durante cinco generaciones consecutivas, no se observa ninguna mejora. Por lo tanto, tenemos $P = 60$, $T = 5$, $p_c = 0{,}75$, $p_m = 0{,}15$, $G = 50$, y $N = 5$. Cabe mencionar que de nuevo los valores de la probabilidad de cruza y de mutación están dentro de los valores pertinentes señalados en [319]

5.6.2 Epsilon Constraint vs NSGA

Gracias a la experimentación previa, se observó que el tiempo computacional que llevaba utilizar el algoritmo ε-restricción era muy grande, llegando a consumir un 99 % del tiempo total del algoritmo.

Esto se debe al enfoque anidado que se utiliza, es decir, para cada solución del líder, resolvemos el problema del seguidor, el cual debido a su naturaleza combinatoria, explotaba conforme el tamaño del problema se hacía más grande.

Es por ello que, con el objetivo de disminuir el esfuerzo computacional, se implementa una variante del ONEA y PNEA propuesto, donde el método de ε-restricción se reemplaza por conocido algoritmo metaheurístico NSGA-II [112].

NSGA-II (Non-Dominated Sorting Genetic Algorithm II) es un técnica de optimización multi-objetivo el cual esta equipado con operadores para obtener soluciones diversas. Funciona de manera aproximativa, es decir, no garantiza encontrar la solución óptima, pero sí busca un conjunto de soluciones que representen el equilibrio entre los diferentes objetivos.

Es importante mencionar que aunque el tiempo computacional se ve mejorado utilizando este algoritmo, puede llevar a soluciones semi-factibles binivel, esto se discutirá más en la sección 5.6.4.

5.6.3 Resultados Computacionales

Todos los experimentos computacionales se realizaron en una computadora personal con un procesador Intel(R) Core i5-10400F con 16 GB de RAM bajo Windows 10 Professional. ONEA y PNEA se implementaron en Python 3.9 utilizando Gurobi 11.0 para resolver el problema escalarizado del seguidor. Además, el NSGA-II se adaptó de la biblioteca DEAP para Python, que proporciona marcos de algoritmos evolutivos implementados para resolver problemas de alta complejidad. Debido a la estocasticidad involucrada en ambas versiones de los algoritmos probados, se realizaron 10 ejecuciones para cada instancia.

Los resultados del ONEA se resumen en la Tabla 5.6, que está dividida en dos secciones: una dedicada al NEA propuesto, con el método de ε-restricción y otra dedicada al NEA con NSGA-II para fines de comparación. La primera columna indica la etiqueta de la instancia que se está resolviendo. Posteriormente, para cada sección, en las 3 primeras columnas de la tabla se informa los mejores, el promedio y los peores valores de la función objetivo del líder obtenidos en las 10 ejecuciones respectivamente. La cuarta columna muestra el tiempo promedio para resolver cada instancia, medido en segundos. Finalmente, la última columna, muestra el número de ejecuciones en las que se alcanzó el mejor valor obtenido. La Tabla 5.7 es similar a la Tabla 5.6 pero resume los resultados obtenidos por el PNEA.

5.6.4 Discusión

En esta sección se discutirán los resultados obtenidos a través de la experimentación computacional.

En la Tabla 5.6, es evidente que el ONEA con el método de ε-restricción consistentemente alcanzó la mejor solución conocida en las 10 ejecuciones para las instancias pequeñas, reflejado por valores idénticos en las columnas Mejor, Promedio y Peor. Notablemente, esta versión del ONEA completó la optimización en menos de 4.11 minutos para este conjunto de instancias. Sin embargo, es crucial destacar que una parte significativa (más del 99 %) del tiempo total fue consumida por la rutina para obtener la reacción racional del seguidor, es decir, mediante la implementación del método de ε-restricción. Pasando a las instancias de tamaño mediano, el ONEA alcanzó la mejor solución conocida en 7 u 8 de las diez ejecuciones, con los valores promedio y peor alineándose estrechamente con el mejor resultado. El tiempo computacional aumentó a 67.1 minutos debido a los esfuerzos intensificados para encontrar los frentes de pareto para cada solución del líder.

	ε - restricción					NSGA-II				
Instancia	Mejor	Promedio	Peor	Tiempo	#Hits	Mejor	Promedio	Peor	Tiempo	#Hits
Small.a	515.4	515.4	515.4	197.31	10	**527.1***	513.4	499.4	13.21	7
Small.b	531.2.	531.2	531.2	183	10	531.2	531.2	531.2	12.1	10
Small.c	510.1	510.1	510.1	247.54	10	**512.1***	509.4	507.6	10.54	6
Small.d	559.7	559.7	559.7	168.51	10	**610.8***	559.7	559.7	14.95	6
Small.e	535.1	535.1	535.1	201.3	10	535.1	530.96	514.4	12.36	8
Medium.a	1045.9	1030.2	1003.4	>3200	5	1045.9	1014.79	1029.8	183.57	1
Medium.b	923.9	919.24	888.5	>3200	6	923.9	916.7	908.3	114.9	2
Medium.c	1034	1030.06	1015.7	>3200	7	987	978.15	950.3	123.71	1
Medium.d	951.9	950.34	940.6	>3200	8	949.2	944.68	937.6	113.15	1
Medium.e	973.7	971.31	966.3	>3200	6	972.2	970.31	967.5	115.21	2

Tabla 5.6: Resumen de los resultados obtenidos por el ONEA

El PNEA con el método de ε-restricción arrojó resultados comparables como se puede observar en la tabla 5.7, reflejando el desempeño del ONEA para las instancias pequeñas, con la mejor función objetivo del líder conocida consistentemente alcanzada en las 10 ejecuciones. El tiempo requerido es similar al del ONEA, incrementándose ligeramente a 4.21 minutos. Sin embargo, para las instancias de tamaño mediano, esta versión del PNEA aseguró la mejor solución conocida en 6 de las 10 ejecuciones para cuatro de las cinco instancias probadas, con un correspondiente aumento en el tiempo computacional a 67.1 minutos. Notablemente, el enfoque pesimista parece ser más intrincado que el optimista. De manera análoga al ONEA, el tiempo computacional para el método de ε-restricción representa más del 99.9 % del tiempo total consumido por el PNEA.

	ε - restricción					NSGA-II				
Instancia	Mejor	Promedio	Peor	Tiempo	#Hits	Mejor	Promedio	Peor	Tiempo	#Hits
Small.a	435.7	435.7	435.7	196.21	10	**454.6***	435.7	435.7	11.51	9
Small.b	458.6	458.6	458.6	182.46	10	458.6	458.6	458.6	10.17	10
Small.c	398.2	398.2	398.2	253.19	10	**413.1***	398.2	398.2	12.37	8
Small.d	488.9	488.9	488.9	167.56	10	**492.2***	488.9	488.9	11.77	9
Small.e	405	405	405	215.23	10	**429.4***	405	405	13.14	9
Medium.a	857.2	852.15	839.3	>3200	7	857.2	843.14	832.9	98.6	1
Medium.b	795.3	789.82	795.3	>3200	6	795.1	782.09	757.2	117.34	1
Medium.c	825.3	821.56	811.4	>3200	7	823.4	811.1	799.3	162.21	1
Medium.d	843.7	831.1	783.5	>3200	6	842.7	842.7	817.3	141.74	1
Medium.e	815.6	801.34	757.3	>3200	4	815.3	804.16	787.4	105.96	2

Tabla 5.7: Resumen de los resultados obtenidos por el PNEA

Para resaltar visualmente las diferencias en la ganancia del líder al adoptar el enfoque optimista o pesimista, las Figuras 5.10 y 5.11 muestran las ganancias obtenidas para instancias pequeñas y medianas, respectivamente. La ganancia del líder bajo el enfoque optimista se representa en azul (línea punteada), mientras que el enfoque pesimista se representa en rojo (línea continua). Notablemente, para las instancias de tamaño pequeño,

small.e se destaca, obteniendo la mayor ganancia para el líder bajo el enfoque optimista y mostrando la diferencia más sustancial en comparación con el enfoque pesimista. Por el contrario, dentro del enfoque pesimista, la instancia small.c refleja la ganancia menos favorable para el líder.

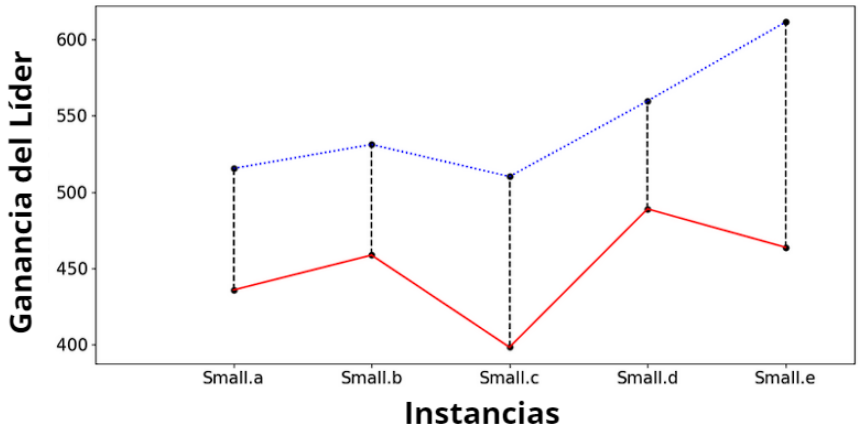


Figura 5.10: Comparación de los enfoques asumidos para las instancias pequeñas

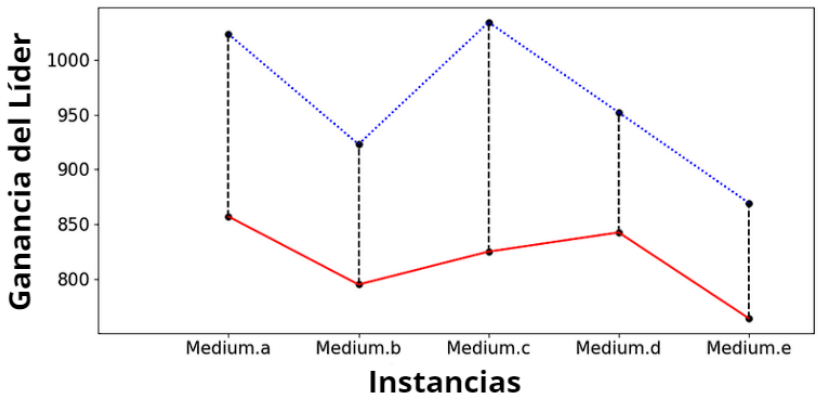


Figura 5.11: Comparación de los enfoques asumidos para las istancias medianas

En el contexto de las instancias de tamaño mediano, el enfoque pesimista parece exhibir una mayor estabilidad, como lo evidencia una menor variación en comparación con el enfoque optimista. Notablemente, la instancia medium.e se destaca al ofrecer la ganancia menos favorable para el líder tanto en los escenarios optimista como pesimista. Las tres primeras instancias medianas muestran notablemente una disparidad sustancial entre la ganancia del líder bajo los enfoques optimista y pesimista.

Cabe resaltar el impacto que tiene aproximar la reacción del seguidor. Primero, es importante recordar que las soluciones factibles en la optimización binivel deben considerar la reacción óptima del seguidor. Esta característica importante del NEA se aborda cuando se utiliza el método de ε-restricción para obtener el frente de Pareto del seguidor. Cuando se usa NSGA-II para obtener la reacción del seguidor, no se puede asegurar su reacción óptima. En este caso, se obtiene una aproximación del frente de Pareto.

Podemos observar, a partir de los resultados presentados en las Tablas 5.6 y 5.7 que, en los resultados donde aparece un $*$, el frente de Pareto óptimo no es obtenido por NSGA-II. Por lo tanto, se está considerando una aproximación que es ligeramente diferente de la óptima, desviándose típicamente por una o dos soluciones no dominadas. No obstante, estas soluciones pueden llevar a una sobre estimación de la función objetivo del líder, resultando en una mayor ganancia para el líder, correspondiente a una solución semi-factible del problema binivel.

A pesar de la importancia de reducir el tiempo computacional requerido por los métodos exactos para obtener la reacción del seguidor, el uso de algoritmos de aproximación debe manejarse con cuidado al resolver el problema del seguidor.

Capítulo 6

Conclusiones

Finalmente, en este capitulo se presentará un resumen de los hallazgos obtenidos a lo largo de esta tesis, destacando las conclusiones generales, la relevancia y la exploración de posibles líneas de trabajo que podría derivarse, con el fin de orientar investigaciones posteriores.

6.1 Conclusiones Generales

En esta tesis se ha estudiado a través de distintos problemas, metaheuristicas para problemas de programación binivel. El objetivo era mostrar esta área que ha tomado gran relevancia en los últimos años y mostrar las ventajas que conlleva utilizar esta clase de algoritmos. A través de los capítulos hemos presentado no solo diversos contextos donde los problemas de programación binivel resultan en una buena herramienta para modelar situaciones donde existe una toma de decisiones jerarquizada y relacionadas si no también, esquemas de solución a través de estos algoritmos.

Primero, se mostró un análisis completo tanto bibliográfico como bibliométrico de metaheuristicas para problemas de programación binivel. A través de la revisión de más de 250 artículos enfocados en el tema a investigar se respondieron algunas preguntas de investigación entre las que se encuentran, artículos por año, journals, autores, principales componentes de las metaheurísticas, enfoques asumidos al resolver un problema de programación binivel a través de metaheuristicas, etc. A través de este análisis, algunas practicas buenas y no tan buenas se han evidenciado cuando se trata con metaheurísticas por lo que distintas áreas prometedoras para futuras lineas de investigación se presentan, como la importancia de asumir enfoques cuando se tiene múltiples soluciones óptimas para el problema del seguidor o cuando el problema del nivel inferior se tiene un problema multi-objetivo.

En base a la investigación realizada es claro que el seguir desarrollando metaheurísticas es una tarea importante, es por ello que en el tercer capitulo un problema de localiza-

ción de hubs en un contexto de redes de telecomunicaciones 5G se presenta. El objetivo de esta investigación, además de presentar un contexto novedoso en las telecomunicaciones, en donde usuarios buscan conectarse a hubs 5G para realizar sus tareas diarias garantizando la conectividad es mostrar una de las primeras iteraciones de un algoritmo metaheuristico que se se realizo en el doctorado, el cual es un algoritmo evolutivo anidado. Esta clase de algoritmos, son importantes, ante la falta de software especializado, muchas veces el tiempo es crucial durante la etapa de planeación, ya que la información puede cambiar de un día al otro y obtener una solución, la cual, aunque es aproximada, puede ayudar a satisfacer a los directores o inversores en las empresas.

Continuando con las conclusiones, en el cuarto capitulo un problema de localización de hubs en un contexto de redes de transporte y un novedoso esquema de solución se expone. En este problema, una situación donde las decisiones de creación y uso de redes de trasporte se modela utilizando programación binivel. En este problema, un tomador de decisiones debe decidir la estructura de la red en forma de árbol, decidiendo los hubs a abrir, los arcos para conectarla en forma de un grafo árbol y los precios que se deben pagar para usarla, por otro lado, un tomador de decisiones con menor jerarquía decide como hacer uso de esta red de transporte. Debido a la amplia relación que tiene las decisiones del nivel superior y debido al número exponencial de restricciones que se generan al formular el problema, se optó por el desarrollo de un algoritmo co-evolutivo que integra las decisiones del nivel superior para luego resolver el problema del nivel inferior de manera exacta. Es importante mencionar que el uso de algoritmos co-evolutivos para resolver el problema del nivel superior no se le había prestado la atención debida como se pudo observar en la revisión de metaheurísticas para programación binivel. Para la experimentación se adaptaron instancias conocidas y los resultados muestran una amplia estabilidad dentro del algoritmo.

Como ya se mencionó, gracias al review completo de metaheurísticas, en muchos de los trabajos existe una falta clara de información cuando se trata con múltiples soluciones óptimas al resolver el problema del nivel inferior, como ya se ha mencionado, uno de los aspectos claves en los problemas de programación binivel, es garantizar la reacción óptima del seguidor, si la solución es única, entonces podemos garantizar soluciones binivel factibles, sin embargo, cuando se tiene múltiples soluciones óptimas o cuando se obtiene soluciones que matemáticamente son iguales de "buenas"se debe asumir un enfoque, de esta manera, la reproducibilidad de los resultados se garantiza para futuras investigaciones. Esto se explica en el sexto y último capitulo, donde se presenta un problema binivel multi-objetivo de localización de plantas competitivas. En este problema, dos empresas compiten para establecer una posición en el mercado, la primera busca maximizar su ganancia, mientras que la segunda busca lo mismo y además que las plantas estén alejadas

entre si, esto se puede modelar utilizando programación binivel multi-objetivo. Como ya se ha mencionado, asumir un enfoque cuando se resuelve el problema del nivel inferior es importante, es por ello que a través de un ejemplo, mostramos como asumir distintos enfoques puede llevar a distintas soluciones, para esto, se resuelve el problema considerando algunos enfoques clásicos y se proponen dos nuevos en base a la densidad de las soluciones obtenidas del nivel inferior. Para resolver instancias del problema, se adaptaron dos algoritmos basados en los enfoques optimista y pesimista, para ello se combinó un algoritmo evolutivo anidado con un algoritmo ε-restricción para el nivel inferior, de esta manera se garantiza que el algoritmo encontrará el frente de pareto exacto, garantizando poder utilizar algunos de los enfoques optimista o pesimista, la experimentación muestra que aunque el algoritmo funciona de manera correcta, el tiempo juega una desventaja enorme conforme el tamaño del problema aumenta debido al enfoque anidado y el algoritmo ε-restricción, es por ello que, para comparar resultados se decidió sustituir el algoritmo ε-restricción por un NSGA-II, sin embargo, como se puede observar en los resultados, el aproximar el frente de pareto o dicho de otra manera, aproximar la reacción del seguidor aunque se ve una reducción sustancial de tiempo, puede llevar a soluciones binivel semi-factibles, por lo que se debe tener cuidado.

6.2 Trabajo a futuro

Es claro que los problemas de programación binivel y las metaheurísticas para problemas de programación binivel son muy importantes y al ser un área de continuo desarrollo, también es claro que aún hay mucho trabajo por hacer, no solo en modelar situaciones utilizando programación binivel sino también la mejora continua de las metaheurísticas, haciendo operadores más ajustados y reduciendo el GAP entre las soluciones aproximadas obtenidas por una meta-heurística y aquellas obtenidas de manera exacta.

Un área que ha tomado importancia de manera relativo es el uso de métodos subrogados para aproximar la reacción del seguidor. Los métodos subrogados son técnicas utilizadas en la optimización para aproximar funciones objetivo o que son difíciles de manejar directamente. Dentro de la programación binivel ha tenido un auge importante, ya que como se sabe, obtener la reacción del seguidor puede ser muy complicado, por lo que aproximar estas funciones objetivo y luego que el aprendizaje automático ayude a mejorar el método subrogado para reducir la aproximación y obtener un valor objetivo cerca del óptimo.

Otra área muy importante dentro de las metaheurísticas para programación binivel son las metaheuristicas de una solución. Dentro del review de metaheurísticas se observó

que las metaheurísticas favoritas para estos problemas son las poblacionales, esto aunque ayuda a la exploración de más soluciones puede llevar a tiempos computacionales elevados debido al enfoque anidado utilizado para resolver el problema de nivel inferior. Por lo que explorar metaheurísticas con una sola solución como una búsqueda tabú o un recocido simulado vale la pena explorarlo.

Y muchas más áreas de oportunidad, por mencionar algunas puede ser la programación binivel con machine learning para ajuste de hiper-parámetros, metaheurísticas híbridas, mate-heurísticas, así como metaheurísticas con métodos multicriterio es solo la punta del iceberg que conforma todo esta área que son metaheurísticas para programación binivel.

Bibliografía

[1] Convergence analysis of canonical genetic algorithms. *IEEE transactions on neural networks*, 5(1):96–101, 1994.

[2] M. Abbassi, A. Chaabani, L. B. Said, and N. Absi. Bi-level multi-objective combinatorial optimization using reference approximation of the lower level reaction. *Procedia Computer Science*, 176:2098–2107, 2020.

[3] M. Abbassi, A. Chaabani, L. B. Said, and N. Absi. An approximation-based chemical reaction algorithm for combinatorial multi-objective bi-level optimization problems. In *2021 IEEE Congress on Evolutionary Computation (CEC)*, pages 1627–1634. IEEE, 2021.

[4] M. Abbassi, A. Chaabani, N. Absi, and L. Ben Said. An elitist cooperative evolutionary bi-level multi-objective decomposition-based algorithm for sustainable supply chain. *International Journal of Production Research*, 60(23):7013–7032, 2022.

[5] M. Abbassi, A. Chaabani, and L. B. Said. An efficient chemical reaction algorithm for multi-objective combinatorial bi-level optimization. *Engineering Optimization*, 54(4):665–686, 2022.

[6] M. A. Adibi. Dcgael: An optimized ensemble learning using a discrete-continuous bi-level genetic algorithm. *Journal of Information Science and Engineering*, 38(4): 761–774, 2022.

[7] J. Agor and O. Y. Özaltın. Feature selection for classification models via bilevel optimization. *Computers & Operations Research*, 106:156–168, 2019.

[8] M. Ait Laamim, A. Makrizi, and E. Essoufi. Application of genetic algorithm for solving bilevel linear programming problems. In *Bioinspired Heuristics for Optimization*, pages 123–136. Springer, 2019.

[9] E. Aiyoshi and K. Shimizu. A solution method for the static constrained stackelberg problem via penalty method. *IEEE Transactions on Automatic Control*, 29(12): 1111–1114, 1984.

[10] D. Aksen and N. Aras. A bilevel fixed charge location model for facilities under imminent attack. *Computers & Operations Research*, 39(7):1364–1381, 2012.

[11] D. Aksen and N. Aras. A matheuristic for leader-follower games involving facility location-protection-interdiction decisions. In *Metaheuristics for Bi-level Optimization*, pages 115–151. Springer, 2013.

[12] L. Alanís-López, M.-S. Casas-Ramírez, and J.-F. Camacho-Vallejo. Solving binary programming problems using homotopy theory ideas. *Engineering Computations*, 39(5):1642–1668, 2022.

[13] V. M. Albornoz and P. I. Vera. Coordinating harvest planning and scheduling in an agricultural supply chain through a stochastic bilevel programming. *International Transactions in Operational Research*, 30(4):1819–1842, 2023.

[14] E. Alekseeva and Y. Kochetov. Matheuristics and exact methods for the discrete (r—p)-centroid problem. *Metaheuristics for Bi-level Optimization*, pages 189–219, 2013.

[15] E. Alekseeva, N. Kochetova, Y. Kochetov, and A. Plyasunov. A hybrid memetic algorithm for the competitive p-median problem. *IFAC Proceedings Volumes*, 42 (4):1533–1537, 2009.

[16] E. Alekseeva, Y. Kochetov, and E.-G. Talbi. A matheuristic for the discrete bilevel problem with multiple objectives at the lower level. *International Transactions in Operational Research*, 24(5):959–981, 2017.

[17] N. Aliakbarian, F. Dehghanian, and M. Salari. A bi-level programming model for protection of hierarchical facilities under imminent attacks. *Computers & Operations Research*, 64:210–224, 2015.

[18] M. J. Alves and C. H. Antunes. An illustration of different concepts of solutions in semivectorial bilevel programming. In *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–7. IEEE, 2016.

[19] M. J. Alves and C. H. Antunes. A semivectorial bilevel programming approach to optimize electricity dynamic time-of-use retail pricing. *Computers & Operations Research*, 92:130–144, 2018.

[20] M. J. Alves and J. P. Costa. An algorithm based on particle swarm optimization for multiobjective bilevel linear problems. *Applied Mathematics and Computation*, 247:547–561, 2014.

[21] O. Amirtaheri, M. Zandieh, and B. Dorri. A bi-level programming model for decentralized manufacturer-distributer supply chain considering cooperative advertising. *Scientia Iranica*, 25(2):891–910, 2018.

[22] G. Anandalingam and D. White. A solution method for the linear static stackelberg problem using penalty functions. *IEEE Transactions on Automatic Control*, 35(10): 1170–1173, 1990.

[23] J. S. Angelo and H. J. Barbosa. A study on the use of heuristics to solve a bilevel programming problem. *International Transactions in Operational Research*, 22(5): 861–882, 2015.

[24] J. S. Angelo, E. Krempser, and H. J. Barbosa. Differential evolution for bilevel programming. In *2013 IEEE Congress on Evolutionary Computation*, pages 470–477. IEEE, 2013.

[25] J. S. Angelo, E. Krempser, and H. J. Barbosa. Differential evolution assisted by a surrogate model for bilevel programming problems. In *2014 IEEE Congress on Evolutionary Computation (CEC)*, pages 1784–1791. IEEE, 2014.

[26] Z. Ankhili and A. Mansouri. An exact penalty on bilevel programs with linear vector optimization lower level. *European Journal of Operational Research*, 197(1):36–41, 2009.

[27] C. Aranha, C. L. Camacho Villalón, F. Campelo, M. Dorigo, R. Ruiz, M. Sevaux, K. Sörensen, and T. Stützle. Metaphor-based metaheuristics, a call for action: the elephant in the room. *Swarm Intelligence*, 16(1):1–6, 2022.

[28] N. Aras and H. Küçükaydın. Bilevel models on the competitive facility location problem. In *Spatial interaction models: facility location using game theory*, pages 1–19. Springer, 2017.

[29] J. M. Arroyo and F. J. Fernández. A genetic algorithm approach for the analysis of electric grid interdiction with line switching. In *2009 15th International Conference on Intelligent System Applications to Power Systems*, pages 1–6. IEEE, 2009.

[30] J. M. Arroyo and F. J. Fernández. A genetic algorithm for power system vulnerability analysis under multiple contingencies. *Metaheuristics for Bi-level Optimization*, pages 41–68, 2013.

[31] A. P. Barbosa-Póvoa, C. da Silva, and A. Carvalho. Opportunities and challenges in sustainable supply chain: An operations research perspective. *European Journal of Operational Research*, 268(2):399–431, 2018.

[32] J. F. Bard. An investigation of the linear three level programming problem. *IEEE Transactions on Systems, Man, and Cybernetics*, (5):711–717, 1984.

[33] J. F. Bard. *Practical bilevel optimization: algorithms and applications*, volume 30. Springer Science & Business Media, 2013.

[34] J. F. Bard and J. T. Moore. A branch and bound algorithm for the bilevel programming problem. *SIAM Journal on Scientific and Statistical Computing*, 11(2): 281–292, 1990.

[35] O. Baskan and H. Ceylan. Modified differential evolution algorithm for the continuous network design problem. *Procedia-Social and Behavioral Sciences*, 111:48–57, 2014.

[36] J. E. Beasley. Population heuristics. *Handbook of applied optimization*, 138:156, 2002.

[37] B. Behnia, I. Mahdavi, B. Shirazi, and M. M. Paydar. A bi-level bi-objective mathematical model for cellular manufacturing system applying evolutionary algorithms. *Scientia Iranica*, 26(4):2541–2560, 2019.

[38] B. Behnia, B. Shirazi, I. Mahdavi, and M. M. Paydar. Nested bi-level metaheuristic algorithms for cellular manufacturing systems considering workersâ[[[ERROR FOR PACKAGE inputenc]]][[[ERROR FOR PACKAGE inputenc]]] interest. *RAIRO-Operations Research*, 55:S167–S194, 2021.

[39] V. Beresnev and A. Melnikov. ε-constraint method for bi-objective competitive facility location problem with uncertain demand scenario. *EURO Journal on Computational Optimization*, 8(1):33–59, 2020.

[40] W. F. Bialas and M. H. Karwan. Two-level linear programming. *Management Science*, 30(8):1004–1020, 1984.

[41] B. Biesinger, B. Hu, and G. Raidl. An evolutionary algorithm for the leader-follower facility location problem with proportional customer behavior. In *International Conference on Learning and Intelligent Optimization*, pages 203–217. Springer, 2014.

[42] B. Biesinger, B. Hu, and G. Raidl. A hybrid genetic algorithm with solution archive for the discrete (r— p)(r— p)-centroid problem. *Journal of Heuristics*, 21:391–431, 2015.

[43] B. Biesinger, B. Hu, and G. Raidl. Models and algorithms for competitive facility location problems with different customer behavior. *Annals of Mathematics and Artificial Intelligence*, 76(1-2):93–119, 2016.

[44] B. Biesinger, B. Hu, and G. Raidl. *A Memetic Algorithm for Competitive Facility Location Problems*, pages 637–660. 05 2019. ISBN 978-3-030-06221-7. doi: 10.1007/978-3-030-06222-4_15.

[45] V. Blanco and A. Marín. Upgrading nodes in tree-shaped hub location. *Computers and Operations Research*, 102, 02 2018.

[46] N. Boland, M. Krishnamoorthy, A. T. Ernst, and J. Ebery. Preprocessing and cutting for multiple allocation hub location problems. *European Journal of Operational Research*, 155(3):638–653, 2004.

[47] H. Bonnel and J. Morgan. Semivectorial bilevel optimization problem: penalty approach. *Journal of Optimization Theory and Applications*, 131:365–382, 2006.

[48] M. Bostian, G. Whittaker, B. Barnhart, R. Färe, and S. Grosskopf. Valuing water quality tradeoffs at different spatial scales: An integrated approach using bilevel optimization. *Water Resources and Economics*, 11:1–12, 2015.

[49] M. Bostian, G. Whittaker, A. Sinha, and B. Barnhart. Incorporating data envelopment analysis solution methods into bilevel multi-objective optimization. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pages 1667–1674. IEEE, 2015.

[50] J. Bracken and J. T. McGill. Mathematical programs with optimization problems in the constraints. *Operations Research*, 21(1):37–44, 1973.

[51] L. Brotcorne, M. Labbé, P. Marcotte, and G. Savard. A bilevel model for toll optimization on a multicommodity transportation network. *Transportation Science*, 35(4):345–358, 2001.

[52] L. Brotcorne, F. Cirinei, P. Marcotte, and G. Savard. A tabu search algorithm for the network pricing problem. *Computers & Operations Research*, 39(11):2603–2611, 2012.

[53] E. K. Burke, E. K. Burke, G. Kendall, and G. Kendall. *Search methodologies: introductory tutorials in optimization and decision support techniques*. Springer, 2014.

[54] E. K. Burke, M. R. Hyde, G. Kendall, G. Ochoa, E. Özcan, and J. R. Woodward. A classification of hyper-heuristic approaches: revisited. *Handbook of metaheuristics*, pages 453–477, 2019.

[55] H. I. Calvete and C. Galé. On linear bilevel problems with multiple objectives at the lower level. *Omega*, 39(1):33–40, 2011.

[56] H. I. Calvete, C. Gale, and P. M. Mateo. A new approach for solving linear bilevel problems using genetic algorithms. *European Journal of Operational Research*, 188 (1):14–28, 2008.

[57] H. I. Calvete, C. Galé, and P. M. Mateo. A genetic algorithm for solving linear fractional bilevel problems. *Annals of Operations Research*, 166(1):39–56, 2009.

[58] H. I. Calvete, C. Galé, and M.-J. Oliveros. Bilevel model for production–distribution planning solved by using ant colony optimization. *Computers & Operations Research*, 38(1):320–327, 2011.

[59] H. I. Calvete, C. Galé, and J. A. Iranzo. An efficient evolutionary algorithm for the ring star problem. *European Journal of Operational Research*, 231(1):22–33, 2013.

[60] H. I. Calvete, C. Galé, and M.-J. Oliveros. A hybrid algorithm for solving a bilevel production-distribution planning problem. In *International Conference on Modeling and Simulation in Engineering, Economics and Management*, pages 138–144. Springer, 2013.

[61] H. I. Calvete, C. Galé, J. A. Iranzo, J.-F. Camacho-Vallejo, and M.-S. Casas-Ramírez. A matheuristic for solving the bilevel approach of the facility location problem with cardinality constraints and preferences. *Computers & Operations Research*, 124:105066, 2020.

[62] H. I. Calvete, C. Galé, and J. A. Iranzo. An evolutionary algorithm for a bilevel biobjective location-routing-allocation problem. In *Advances in Evolutionary and Deterministic Methods for Design, Optimization and Control in Engineering and Sciences*, pages 17–33. Springer, 2021.

[63] H. I. Calvete, C. Gale, J. A. Iranzo, and P. Toth. The school bus routing problem with student choice: a bilevel approach and a simple and effective metaheuristic. *International Transactions in Operational Research*, 30(2):1092–1119, 2023.

[64] J.-F. Camacho-Vallejo and C. Garcia-Reyes. Co-evolutionary algorithms to solve hierarchized steiner tree problems in telecommunication networks. *Applied Soft Computing*, 84:105718, 08 2019.

[65] J.-F. Camacho-Vallejo and C. Garcia-Reyes. Co-evolutionary algorithms to solve hierarchized steiner tree problems in telecommunication networks. *Applied Soft Computing*, 84:105718, 2019.

[66] J.-F. Camacho-Vallejo, Á. E. Cordero-Franco, and R. G. González-Ramírez. Solving the bilevel facility location problem under preferences by a stackelberg-evolutionary algorithm. *Mathematical Problems in Engineering*, 2014, 2014.

[67] J.-F. Camacho-Vallejo, J. Mar-Ortiz, F. López-Ramos, and R. P. Rodríguez. A genetic algorithm for the bi-level topological design of local area networks. *PloS one*, 10(6):e0128067, 2015.

[68] J.-F. Camacho-Vallejo, R. Muñoz-Sánchez, and J. L. González-Velarde. A heuristic algorithm for a supply chain production distribution planning. *Computers & Operations Research*, 61:110–121, 2015.

[69] J.-F. Camacho-Vallejo, J.-C. García-Vélez, and C. Corpus. A 5g hubs location hierarchized problem that balances the connection of the users. *Mobile Networks and Applications*, pages 1–11, 2022.

[70] J.-F. Camacho-Vallejo, L. López-Vera, A. E. Smith, and J.-L. González-Velarde. A tabu search algorithm to solve a green logistics bi-objective bi-level problem. *Annals of Operations Research*, 316(2):927–953, 2022.

[71] J.-F. Camacho-Vallejo, C. Corpus, and J. G. Villegas. Metaheuristics for bilevel optimization: A comprehensive review. *Computers & Operations Research*, page 106410, 2023.

[72] J.-F. Camacho-Vallejo, D. Dávila, and S. Nucamendi-Guillén. A hierarchized green supply chain with customer selection, routing, and nearshoring. *Computers & Industrial Engineering*, 178:109151, 2023.

[73] C. L. Camacho-Villalón, M. Dorigo, and T. Stützle. Exposing the grey wolf, moth-flame, whale, firefly, bat, and antlion algorithms: six misleading optimization techniques inspired by bestial metaphors. *International Transactions in Operational Research*, 2022.

[74] C. Campos-Rodríguez, J. A. Moreno-Pérez, and D. R. Santos-Peñate. Particle swarm optimization with two swarms for the discrete (r— p)-centroid problem. In *International Conference on Computer Aided Systems Theory*, pages 432–439. Springer, 2011.

[75] W. Candler and R. Norton. *Multi-level programming and development policy*. The World Bank, 1977.

[76] W. Candler and R. Townsley. A linear two-level programming problem. *Computers & Operations Research*, 9(1):59–76, 1982.

[77] P. Carrasqueira, M. J. Alves, and C. H. Antunes. Bi-level particle swarm optimization and evolutionary algorithm approaches for residential demand response with different user profiles. *Information Sciences*, 418(C):405–420, 2017.

[78] M.-S. Casas-Ramírez and J.-F. Camacho-Vallejo. Solving the p-median bilevel problem with order through a hybrid heuristic. *Applied Soft Computing*, 60(C):73–86, 2017.

[79] M.-S. Casas-Ramírez, J.-F. Camacho-Vallejo, and I.-A. Martínez-Salazar. Approximating solutions to a bilevel capacitated facility location problem with customer's patronization toward a list of preferences. *Applied Mathematics and Computation*, 319:369–386, 2018.

[80] M.-S. Casas-Ramírez, J.-F. Camacho-Vallejo, J. A. Díaz, and D. E. Luna. A bi-level maximal covering location problem. *Operational Research*, 20(2):827–855, 2020.

[81] M. Castelli, L. Manzoni, L. Mariot, M. S. Nobile, and A. Tangherloni. Salp swarm optimization: a critical review. *Expert Systems with Applications*, 189:116029, 2022.

[82] H. Ceylan and M. G. Bell. Traffic signal timing optimisation based on genetic algorithm approach, including driversâ[[[ERROR FOR PACKAGE inputenc]]][[[ERROR FOR PACKAGE inputenc]]] routing. *Transportation Research Part B: Methodological*, 38(4):329–342, 2004.

[83] A. Chaabani, S. Bechikh, and L. B. Said. A co-evolutionary decomposition-based algorithm for bi-level combinatorial optimization. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pages 1659–1666. IEEE, 2015.

[84] A. Chaabani, S. Bechikh, and L. B. Said. A co-evolutionary decomposition-based chemical reaction algorithm for bi-level combinatorial optimization problems. *Procedia Computer Science*, 112(C):780–789, 2017.

[85] A. Chaabani, S. Bechikh, and L. B. Said. A new co-evolutionary decomposition-based algorithm for bi-level combinatorial optimization. *Applied Intelligence*, 48: 2847–2872, 2018.

[86] A. Chaabani, S. Bechikh, and L. Ben Said. A co-evolutionary hybrid decomposition-based algorithm for bi-level combinatorial optimization problems. *Soft Computing*, 24(10):7211–7229, 2020.

[87] M. K. Chalmardi and J.-F. Camacho-Vallejo. A bi-level programming model for sustainable supply chain network design that considers incentives for using cleaner technologies. *Journal of Cleaner Production*, 213:1035–1050, 2019.

[88] A. Chen, J. Kim, S. Lee, and Y. Kim. Stochastic multi-objective models for network design problem. *Expert Systems with Applications*, 37(2):1608–1619, 2010.

[89] D. Chen and Z. Yang. Systematic optimization of port clusters along the maritime silk road in the context of industry transfer and production capacity constraints. *Transportation Research Part E: Logistics and Transportation Review*, 109:174–189, 2018.

[90] H. Chen, X. Luo, Z. Zhang, and Q. Zhou. Stochastic bi-level programming model for home healthcare scheduling problems considering the degree of satisfaction with visit time. *Journal of Systems Science and Systems Engineering*, 30:572–599, 2021.

[91] X. Chen, T. Zuo, M. Lang, S. Li, and S. Li. Integrated optimization of transfer station selection and train timetables for road–rail intermodal transport network. *Computers & Industrial Engineering*, 165:107929, 2022.

[92] Y. Chen, A. Bouferguene, Y. Shen, X. Yin, and M. Al-Hussein. Bilevel decision-support model for bus-route optimization and accessibility improvement for seniors. *Journal of Computing in Civil Engineering*, 34(2):04019057, 2020.

[93] A. Cheraghalipour and E. Roghanian. A bi-level model for a closed-loop agricultural supply chain considering biogas and compost. *Environment, Development and Sustainability*, pages 1–47, 2022.

[94] A. Cheraghalipour, M. M. Paydar, and M. Hajiaghaei-Keshteli. Designing and solving a bi-level model for rice supply chain using the evolutionary algorithms. *Computers and Electronics in Agriculture*, 162:651–668, 2019.

[95] P. Clark and A. Westerberg. A note on the optimality conditions for the bilevel programming problem. *Naval Research Logistics*, 35(5):413–418, 1988.

[96] B. Colson, P. Marcotte, and G. Savard. An overview of bilevel optimization. *Annals of Operations Research*, 153:235–256, 2007.

[97] I. Contreras, E. Fernández, and A. Martín. Tight bounds from a path based formulation for the tree of hub location problem. *Computers and Operations Research*, 36(12):3117–3127, 2009.

[98] I. Contreras, E. Fernández, and A. MartÃn. The tree of hubs location problem. *European Journal of Operational Research*, 202(2):390–400, 2010.

[99] I. Contreras, J.-F. Cordeau, and G. Laporte. The dynamic uncapacitated hub location problem. *Transportation Science*, 45(1):18–32, 2011.

[100] I. Correia, S. Nickel, and F. Saldanha-da Gama. A stochastic multi-period capacitated multiple allocation hub location problem: Formulation and inequalities. *Omega*, 74:122–134, 2018.

[101] T. Crainic. Parallel metaheuristics and cooperative search. In *Handbook of Metaheuristics*, pages 419–451. Springer, 2019.

[102] M. da Graça Costa, M. E. Captivo, and J. Clímaco. Capacitated single allocation hub location problem a bicriteria approach. *Computers & operations research*, 35 (11):3671–3695, 2008.

[103] L. D. Davis, K. De Jong, M. D. Vose, and L. D. Whitley. *Evolutionary algorithms*, volume 111. Springer Science & Business Media, 2012.

[104] I. Davydov, Y. Kochetov, and E. Carrizosa. Vns heuristic for the (r— p)–centroid problem on the plane. *Electronic Notes in Discrete Mathematics*, 39:5–12, 2012.

[105] I. A. Davydov, Y. A. Kochetov, N. Mladenovic, and D. Urosevic. Fast metaheuristics for the discrete (r— p)-centroid problem. *Automation and Remote Control*, 75(4): 677–687, 2014.

[106] J. de Armas, E. Lalla-Ruiz, S. L. Tilahun, and S. Voß. Similarity in metaheuristics: a gentle step towards a comparison methodology. *Natural Computing*, 21(2):265–287, 2022.

[107] P.-T. De Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein. A tutorial on the cross-entropy method. *Annals of operations research*, 134:19–67, 2005.

[108] K. Deb. An efficient constraint handling method for genetic algorithms. *Computer methods in applied mechanics and engineering*, 186(2-4):311–338, 2000.

[109] K. Deb. Multi-objective optimisation using evolutionary algorithms: an introduction. In *Multi-objective evolutionary optimisation for product design and manufacturing*, pages 3–34. 2011.

[110] K. Deb and A. Sinha. An evolutionary approach for bilevel multi-objective problems. In *International Conference on Multiple Criteria Decision Making*, pages 17–24. Springer, 2009.

[111] K. Deb and A. Sinha. An efficient and accurate solution methodology for bilevel multi-objective programming problems using a hybrid evolutionary-local-search algorithm. *Evolutionary Computation*, 18(3):403–449, 2010.

[112] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2): 182–197, 2002.

[113] K. Deb, S. Gupta, J. Dutta, and B. Ranjan. Solving dual problems using a coevolutionary optimization algorithm. *Journal of Global Optimization*, 57(3):891–933, 2013.

[114] S. Dempe. *Foundations of bilevel programming*. Springer Science & Business Media, 2002.

[115] S. Dempe, N. Gadhi, and A. B. Zemkoho. New optimality conditions for the semivectorial bilevel optimization problem. *Journal of Optimization Theory and Applications*, 157:54–74, 2013.

[116] S. Dempe et al. Bilevel programming: Implicit function approach. *Encyclopedia of Optimization*, 2:260–266, 2009.

[117] J. A. Díaz, D. E. Luna, J.-F. Camacho-Vallejo, and M.-S. Casas-Ramírez. Grasp and hybrid grasp-tabu heuristics to solve a maximal covering location problem with customer preference ordering. *Expert Systems with Applications*, 82(C):67–76, 2017.

[118] J. Du, X. Li, L. Yu, R. Dan, and J. Zhou. Multi-depot vehicle routing problem for hazardous materials transportation: A fuzzy bilevel programming. *Information Sciences*, 399(C):201–218, 2017.

[119] J. Ebery, M. Krishnamoorthy, A. Ernst, and N. Boland. The capacitated multiple allocation hub location problem: Formulations and algorithms. *European journal of operational research*, 120(3):614–631, 2000.

[120] G. Erdoğan, M. Battarra, and A. M. Rodríguez-Chía. The hub location and pricing problem. *European Journal of Operational Research*, 301(3):1035–1047, 2022.

[121] H. N. Esfahani, Z. Liu, and Z. Song. Optimal pricing for bidirectional wireless charging lanes in coupled transportation and power networks. *Transportation Research Part C: Emerging Technologies*, 135:103419, 2022.

[122] W. Fan and R. B. Machemehl. Bi-level optimization model for public transportation network redesign problem: Accounting for equity issues. *Transportation Research Record*, 2263(1):151–162, 2011.

[123] A. M. Fathollahi-Fard, M. Hajiaghaei-Keshteli, R. Tavakkoli-Moghaddam, and N. R. Smith. Bi-level programming for home health care supply chain considering outsourcing. *Journal of Industrial Information Integration*, 25:100246, 2022.

[124] X. Feng, L. Jiang, Y. Zhang, and W. Wang. Optimization of capacity of ports within a regional port system. *Transportation research record*, 2222(1):10–16, 2011.

[125] M. Fischetti, I. Ljubić, M. Monaci, and M. Sinnl. A new general-purpose algorithm for mixed-integer bilevel linear programs. *Operations Research*, 65(6):1615–1637, 2017.

[126] J. Fortuny-Amat and B. McCarl. A representation and economic interpretation of a two-level programming problem. *Journal of the Operational Research Society*, 32 (9):783–792, 1981.

[127] A. Frantsev, A. Sinha, and P. Malo. Finding optimal strategies in multi-period stackelberg games using an evolutionary framework. *IFAC Proceedings Volumes*, 45 (25):33–38, 2012.

[128] M. Gallo, L. D Acierno, and B. Montella. A meta-heuristic approach for solving the urban network design problem. *European Journal of Operational Research*, 201(1): 144–157, 2010.

[129] T. Ganesan, P. Vasant, and I. Litvinchev. Chaotic simulator for bilevel optimization of virtual machine placements in cloud computing. *Journal of the Operations Research Society of China*, 10(4):703–723, 2022.

[130] R. Gao and Y. Ma. Uncertain random bilevel programming models and their application to shared capacity routing problem. *Journal of Computational and Applied Mathematics*, 423:114965, 2023.

[131] S. Gao and N. Liu. Improving the resilience of port–hinterland container logistics transportation systems: A bi-level programming approach. *Sustainability*, 14(1): 180, 2021.

[132] S. Gao, X. Xin, C. Li, Y. Liu, and K. Chen. Container ocean shipping network design considering carbon tax and choice inertia of cargo owners. *Ocean & Coastal Management*, 216:105986, 2022.

[133] X. Gao. A bi-level stochastic optimization model for multi-commodity rebalancing under uncertainty in disaster response. *Annals of Operations Research*, 319(1):115–148, 2022.

[134] Y. Gao, G. Zhang, J. Lu, and H.-M. Wee. Particle swarm optimization for bi-level pricing problems in supply chains. *Journal of Global Optimization*, 51(2):245–254, 2011.

[135] M. Gendreau and J.-Y. Potvin. Metaheuristics in combinatorial optimization. *Annals of Operations Research*, 140(1):189–213, 2005.

[136] M. Gendreau, P. Marcotte, and G. Savard. A hybrid tabu-ascent algorithm for the linear bilevel programming problem. *Journal of Global Optimization*, 8(3):217–233, 1996.

[137] N. Ghaffarinasab and A. Motallebzadeh. Hub interdiction problem variants: Models and metaheuristic solution algorithms. *European Journal of Operational Research*, 267(2):496–512, 2018.

[138] R. Ghanbari and N. Mahdavi-Amiri. Solving bus terminal location problems using evolutionary algorithms. *Applied soft computing*, 11(1):991–999, 2011.

[139] F. Glover and J.-K. Hao. The case for strategic oscillation. *Annals of Operations Research*, 183:163–173, 2011.

[140] J. F. Gonçalves and M. G. Resende. Biased random-key genetic algorithms for combinatorial optimization. *Journal of Heuristics*, 17(5):487–525, 2011.

[141] J. L. González Velarde, J.-F. Camacho-Vallejo, and G. Pinto Serrano. A scatter search algorithm for solving a bilevel optimization model for determining highway tolls. *Computación y Sistemas*, 19(1):05–16, 2015.

[142] Z. Guo. A bilevel multi-parent genetic optimization model for flexible assembly line balancing with work-sharing and workstation revisiting. *Intelligent Decision-making Models for Production and Retail Operations*, pages 87–113, 2016.

[143] Z. Guo, D. Zhang, S. Y.-S. Leung, and L. Shi. A bi-level evolutionary optimization approach for integrated production and transportation scheduling. *Applied Soft Computing*, 42:215–228, 2016.

[144] A. Gupta and Y.-S. Ong. An evolutionary algorithm with adaptive scalarization for multiobjective bilevel programs. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pages 1636–1642. IEEE, 2015.

[145] W. Halter and S. Mostaghim. Bilevel optimization of multi-component chemical systems using particle swarm optimization. In *2006 IEEE International Conference on Evolutionary Computation*, pages 1240–1247. IEEE, 2006.

[146] M. Hammami, S. Bechikh, C.-C. Hung, and L. B. Said. Weighted-features construction as a bi-level problem. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pages 1604–1611. IEEE, 2019.

[147] S. D. Handoko, L. H. Chuin, A. Gupta, O. Y. Soon, H. C. Kim, and T. P. Siew. Solving multi-vehicle profitable tour problem via knowledge adoption in evolutionary bi-level programming. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pages 2713–2720. IEEE, 2015.

[148] A. Haurie, G. Savard, and D. White. A note on: an efficient point algorithm for a linear two-stage optimization problem. *Operations Research*, 38(3):553–555, 1990.

[149] R. Hayashi, H. Takano, W. M. Nyabuto, H. Asano, and T. Nguyen-Duc. Bilevel optimization model for sizing of battery energy storage systems in a microgrid considering their economical operation. *Energy Reports*, 9:728–737, 2023.

[150] L. Hecheng and W. Yuping. Exponential distribution-based genetic algorithm for solving mixed-integer bilevel programming problems. *Journal of Systems Engineering and Electronics*, 19(6):1157–1164, 2008.

[151] S. R. Hejazi, A. Memariani, G. Jahanshahloo, and M. M. Sepehri. Linear bilevel programming solution by genetic algorithm. *Computers & Operations Research*, 29 (13):1913–1925, 2002.

[152] M. S. Hossain, R. K. Chakrabortty, S. El Sawah, and M. J. Ryan. Sustainable modular product architecture design by bi-level leader-follower joint optimization with switching-based meta-heuristic algorithm. *Journal of Cleaner Production*, 306: 127108, 2021.

[153] E. Hosseini, A. S. Sadiq, K. Z. Ghafoor, D. B. Rawat, M. Saif, and X. Yang. Volcano eruption algorithm for solving optimization problems. *Neural Computing and Applications*, 33(7):2321–2337, 2021.

[154] E. Hosseinia. Solving linear-quadratic bi-level programming and linear-fractional bi-level programming problems using genetic algorithm. *Applied Mathematics and Computational Intelligence*, 2(2):169–182, 2013.

[155] B. Huang and N. Liu. Bilevel programming approach to optimizing a logistic distribution network with balancing requirements. *Transportation Research Record*, 1894 (1):188–197, 2004.

[156] Z. Huang, C. Yang, X. Zhou, and W. Gui. A novel cognitively inspired state transition algorithm for solving the linear bi-level programming problem. *Cognitive Computation*, 10(5):816–826, 2018.

[157] Z. Huang, H. Niu, R. Gao, H. Fan, and C. Liu. Optimizing train timetable based on departure time preference of passengers for high-speed rails. *Journal of Advanced Transportation*, 2021:1–20, 2021.

[158] M. M. Islam, H. K. Singh, and T. Ray. A memetic algorithm for solving single objective bilevel optimization problems. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pages 1643–1650. IEEE, 2015.

[159] M. M. Islam, H. K. Singh, T. Ray, and A. Sinha. An enhanced memetic algorithm for single-objective bilevel optimization problems. *Evolutionary Computation*, 25 (4):607–642, 2017.

[160] M. Jerbi, Z. C. Dagdia, S. Bechikh, and L. B. Said. Android malware detection as a bi-level problem. *Computers & Security*, 121:102825, 2022.

[161] R. G. Jeroslow. The polynomial hierarchy and a simple model for competitive analysis. *Mathematical Programming*, 32(2):146–164, 1985.

[162] X. Jia, R. He, C. Zhang, and H. Chai. A bi-level programming model of liquefied petroleum gas transportation operation for urban road network by period-security. *Sustainability*, 10(12):4714, 2018.

[163] Y.-H. Jia, Y. Mei, and M. Zhang. A bilevel ant colony optimization algorithm for capacitated electric vehicle routing problem. *IEEE Transactions on Cybernetics*, 52 (10):10855–10868, 2021.

[164] L. Jiang, X. Wang, K. Yang, and Y. Gao. Bilevel optimization for the reorganization of inland river ports: A niche perspective. *Socio-Economic Planning Sciences*, 86: 101466, 2023.

[165] Y. Jiang, X. Li, C. Huang, and X. Wu. Application of particle swarm optimization based on chks smoothing function for solving nonlinear bilevel programming problem. *Applied Mathematics and Computation*, 219(9):4332–4339, 2013.

[166] A. A. Juan, J. Faulin, S. E. Grasman, M. Rabe, and G. Figueira. A review of simheuristics: Extending metaheuristics to deal with stochastic combinatorial optimization problems. *Operations Research Perspectives*, 2(C):62–72, 2015.

[167] J. J. Júdice and A. Faustino. The solution of the linear bilevel programming problem by using the linear complementarity problem. *Investigação Operacional*, 8(1):77–95, 1988.

[168] V. V. Kalashnikov, S. Dempe, G. A. Pérez-Valdés, N. I. Kalashnykova, and J.-F. Camacho-Vallejo. Bilevel programming and applications. *Mathematical Problems in Engineering*, 2015, 2015.

[169] M. Karaja, A. Chaabani, A. Azzouz, and L. Ben Said. Efficient bi-level multi objective approach for budget-constrained dynamic bag-of-tasks scheduling problem in heterogeneous multi-cloud environment. *Applied Intelligence*, pages 1–29, 2022.

[170] M. Karimi-Mamaghan, M. Mohammadi, P. Meyer, A. M. Karimi-Mamaghan, and E.-G. Talbi. Machine learning at the service of meta-heuristics for solving combinatorial optimization problems: A state-of-the-art. *European Journal of Operational Research*, 296(2):393–422, 2022.

[171] A. Karoonsoontawong and S. T. Waller. Dynamic continuous network design problem: linear bilevel programming and metaheuristic approaches. *Transportation Research Record*, 1964(1):104–117, 2006.

[172] B. Kayışoğlu and İ. Akgün. Multiple allocation tree of hubs location problem for non-complete networks. *Computers & Operations Research*, 136:105478, 2021.

[173] R. Khanduzi and A. Rastegar. An efficient and robust hybrid metaheuristic method to solve a hierarchical bi-level protection-interdiction problem on real healthcare system. *Transactions on Emerging Telecommunications Technologies*, 33(10):e4396, 2022.

[174] A. Kheirkhah, H. Navidi, and M. M. Bidgoli. Dynamic facility layout problem: a new bilevel formulation and some metaheuristic solution methods. *IEEE Transactions on Engineering Management*, 62(3):396–410, 2015.

[175] E. Kieffer, G. Danoy, P. Bouvry, and A. Nagih. A competitive approach for bi-level co-evolution. In *2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pages 609–618. IEEE, 2018.

[176] T. Kleinert and M. Schmidt. Why there is no need to use a big-m in linear bilevel optimization: A computational study of two ready-to-use approaches. *Computational Management Science*, 20(1):3, 2023.

[177] T. Kleinert, M. Labbé, F. a. Plein, and M. Schmidt. Thereâ[[[ERROR FOR PACKAGE inputenc]]][[[ERROR FOR PACKAGE inputenc]]]s no free lunch: on the hardness of choosing a correct big-m in bilevel optimization. *Operations Research*, 68(6):1716–1721, 2020.

[178] T. Kleinert, M. Labbé, I. Ljubić, and M. Schmidt. A survey on mixed-integer programming techniques in bilevel optimization. *EURO Journal on Computational Optimization*, 9:100007, 2021.

[179] A. Koh. Solving transportation bi-level programs with differential evolution. In *2007 IEEE Congress on Evolutionary Computation*, pages 2243–2250. IEEE, 2007.

[180] A. Koh. A metaheuristic framework for bi-level programming problems with multi-disciplinary applications. *Metaheuristics for Bi-level Optimization*, pages 153–187, 2013.

[181] H. Küçükaydın, N. Aras, and I. K. Altınel. A hybrid tabu search heuristic for a bilevel competitive facility location model. In *International Workshop on Hybrid Metaheuristics*, pages 31–45. Springer, 2010.

[182] R. Kuo and Y. Han. A hybrid of genetic algorithm and particle swarm optimization for solving bi-level linear programming problem–a case study on supply chain model. *Applied Mathematical Modelling*, 35(8):3905–3917, 2011.

[183] R. Kuo and C. Huang. Application of particle swarm optimization algorithm for solving bi-level linear programming problem. *Computers & Mathematics with Applications*, 58(4):678–685, 2009.

[184] M. Labbé and A. Violin. Bilevel programming and price setting problems. *Annals of Operations Research*, 11:1–30, 2013.

[185] M. Labbé, P. Marcotte, and G. Savard. A bilevel model of taxation and its application to optimal highway pricing. *Management Science*, 44(12-part-1):1608–1622, 1998.

[186] K.-M. Lan, U.-P. Wen, H.-S. Shih, and E. S. Lee. A hybrid neural network approach to bilevel programming problems. *Applied Mathematics Letters*, 20(8):880–884, 2007.

[187] P. Larrañaga and J. A. Lozano. *Estimation of distribution algorithms: A new tool for evolutionary computation*, volume 2. Springer Science & Business Media, 2001.

[188] S. E. Latifi, R. Tavakkoli-Moghaddam, E. Fazeli, and H. Arefkhani. Competitive facility location problem with foresight considering discrete-nature attractiveness for facilities: Model and solution. *Computers & Operations Research*, 146:105900, 2022.

[189] D. Lee, L. Song, and H. Wang. A genetic algorithm for a bi-level programming model of berth allocation and quay crane scheduling. In *Proceedings of the 85th Annual Meeting of the Transportation Research Board*, 2006.

[190] F. Legillon, A. Liefooghe, and E.-G. Talbi. Cobra: A coevolutionary metaheuristic for bi-level optimization. In *Metaheuristics for Bi-level Optimization*, pages 95–114. Springer, 2013.

[191] D. Leiber and G. Reinhart. A bi-level optimisation approach for assembly line design using a nested genetic algorithm. *International Journal of Production Research*, 59 (24):7560–7575, 2021.

[192] H. Li. A genetic algorithm using a finite search space for solving nonlinear/linear fractional bilevel programming problems. *Annals of Operations Research*, 235(1): 543–558, 2015.

[193] H. Li and Y. Wang. A genetic algorithm for solving a special class of nonlinear bilevel programming problems. In *International Conference on Computational Science*, pages 1159–1162. Springer, 2007.

[194] H. Li and Y. Wang. A hybrid genetic algorithm for solving nonlinear bilevel programming problems based on the simplex method. In *Third International Conference on Natural Computation (ICNC 2007)*, volume 4, pages 91–95. IEEE, 2007.

[195] H. Li and Y. Wang. An evolutionary algorithm based on a new decomposition scheme for nonlinear bilevel programming problems. *International Journal of Communications, Network and System Sciences*, 3(1):87–93, 2010.

[196] H. Li and Y. Wang. A real-binary coded genetic algorithm for solving nonlinear bilevel programming with nonconvex objective functions. In *2011 IEEE Congress of Evolutionary Computation (CEC)*, pages 2496–2500. IEEE, 2011.

[197] H. Li and L. Zhang. A hybrid heuristic approach with adaptive scalarization for linear semivectorial bilevel programming and its application. *Memetic Computing*, 14(4):433–449, 2022.

[198] H. Li, Y. Jiao, and L. Zhang. Orthogonal genetic algorithm for solving quadratic bilevel programming problems. *Journal of Systems Engineering and Electronics*, 21 (5):763–770, 2010.

[199] H. Li, Q. Zhang, Q. Chen, L. Zhang, and Y.-C. Jiao. Multiobjective differential evolution algorithm based on decomposition for a type of multiobjective bilevel programming problems. *Knowledge-Based Systems*, 107:271–288, 2016.

[200] M. Li, D. Lin, and S. Wang. Solving a type of biobjective bilevel programming problem using nsga-ii. *Computers & Mathematics with Applications*, 59(2):706–715, 2010.

[201] Q. Li, M. Li, R. Zhang, and J. Gan. A stochastic bilevel model for facility location-protection problem with the most likely interdiction strategy. *Reliability Engineering & System Safety*, 216:108005, 2021.

[202] X. Li, P. Tian, and X. Min. A hierarchical particle swarm optimization for solving bilevel programming problems. In *International Conference on Artificial Intelligence and Soft Computing*, pages 1169–1178. Springer, 2006.

[203] J. Z. Liang and R. Miikkulainen. Evolutionary bilevel optimization for complex control tasks. In *Proceedings of the 2015 annual conference on genetic and evolutionary computation*, pages 871–878, 2015.

[204] S. Lin, F. Luan, Z. Han, X. Lü, X. Zhou, and W. Liu. Genetic algorithm based on duality principle for bilevel programming problem in steel-making production. *Chinese Journal of Chemical Engineering*, 22(7):742–747, 2014.

[205] X. Lin, Q.-s. Li, and W.-s. Tang. Hybrid intelligent algorithm for solving the bilevel programming models with fuzzy variables. *Systems Engineering-Theory & Practice*, 28(7):100–104, 2008.

[206] B. Liu. Stackelberg-nash equilibrium for multilevel programming with multiple followers using genetic algorithms. *Computers & Mathematics with Applications*, 36 (7):79–89, 1998.

[207] X. Liu. Bi-level planning method of urban electric vehicle charging station considering multiple demand scenarios and multi-type charging piles. *Journal of Energy Storage*, 48:104012, 2022.

[208] M. A. Lones. Mitigating metaphors: A comprehensible guide to recent nature-inspired algorithms. *SN Computer Science*, 1(1):49, 2020.

[209] H. Louati, S. Bechikh, A. Louati, C.-C. Hung, and L. B. Said. Deep convolutional neural network architecture design as a bi-level optimization problem. *Neurocomputing*, 439:44–62, 2021.

[210] H. Louati, S. Bechikh, A. Louati, A. Aldaej, and L. B. Said. Joint design and compression of convolutional neural networks as a bi-level optimization problem. *Neural Computing and Applications*, 34(17):15007–15029, 2022.

[211] F. Lu, T. Yan, H. Bi, M. Feng, S. Wang, and M. Huang. A bilevel whale optimization algorithm for risk management scheduling of information technology projects considering outsourcing. *Knowledge-Based Systems*, 235:107600, 2022.

[212] Z. Lu, K. Deb, and A. Sinha. Handling decision variable uncertainty in bilevel optimization problems. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pages 1683–1690. IEEE, 2015.

[213] A. Luer-Villagra and V. Marianov. A competitive hub location and pricing problem. *European journal of operational research*, 231(3):734–744, 2013.

[214] A. Lüer-Villagra, V. Marianov, H. Eiselt, and G. Méndez-Vogel. The leader multipurpose shopping location problem. *European Journal of Operational Research*, 302 (2):470–481, 2022.

[215] Z. Lukač, K. Šorić, and V. V. Rosenzweig. Production planning problem with sequence dependent setups as a bilevel programming problem. *European Journal of Operational Research*, 187(3):1504–1512, 2008.

[216] M. Ma, H. Huang, X. Song, F. Peña-Mora, Z. Zhang, and J. Chen. Optimal sizing and operations of shared energy storage systems in distribution networks: A bi-level programming approach. *Applied Energy*, 307(C):118170, 2022.

[217] M. Ma, W. Zhao, H. Fan, and Y. Gong. Collaborative optimization of yard crane deployment and inbound truck arrivals with vessel-dependent time windows. *Journal of Marine Science and Engineering*, 10(11):1650, 2022.

[218] T. T. Magalhães and H. J. C. Barbosa. Differential evolution algorithms for solving bilevel optimization problems using computational clusters. In *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 1511–1516. IEEE, 2019.

[219] S. Maldonado-Pinto, M.-S. Casas-Ramírez, and J.-F. Camacho-Vallejo. Analyzing the performance of a hybrid heuristic for solving a bilevel location problem under different approaches to tackle the lower level. *Mathematical Problems in Engineering*, 2016:1–10, 2016.

[220] Z. Maleki, H. R. Maleki, and R. Akbari. A bi-level mathematical model to protect gateways in underwater wireless sensor networks. *International Journal of Sensor Networks*, 40(2):85–93, 2022.

[221] M. M. Mamun, H. K. Singh, and T. Ray. A multifidelity approach for bilevel optimization with limited computing budget. *IEEE Transactions on Evolutionary Computation*, 26(2):392–399, 2021.

[222] V. Maniezzo, M. Boschetti, and T. Stützle. Matheuristics: Algorithms and implementations (euro advanced tutorials on operational research), 2021.

[223] P. Manoharan, R. Walia, C. Iwendi, T. A. Ahanger, S. Suganthi, M. Kamruzzaman, S. Bourouis, W. Alhakami, and M. Hamdi. Svm-based generative adverserial networks for federated learning and edge computing attack model and outpoising. *Expert Systems*, 40(5):e13072, 2023.

[224] S. T. W. Mara, R. Norcahyo, P. Jodiawan, L. Lusiantoro, and A. P. Rifai. A survey of adaptive large neighborhood search algorithms and applications. *Computers & Operations Research*, page 105903, 2022.

[225] M. Marić, Z. Stanimirović, and N. Milenković. Metaheuristic methods for solving the bilevel uncapacitated facility location problem with clients' preferences. *Electronic Notes in Discrete Mathematics*, 39:43–50, 2012.

[226] M. Marić, Z. Stanimirović, N. Milenković, and A. Đenić. Metaheuristic approaches to solving large-scale bilevel uncapacitated facility location problem with clients' preferences. *Yugoslav Journal of Operations Research*, 25(3):361–378, 2015.

[227] Y. Marinakis and M. Marinaki. A bilevel genetic algorithm for a real life location routing problem. *International Journal of Logistics: Research and Applications*, 11 (1):49–65, 2008.

[228] Y. Marinakis and M. Marinaki. A bilevel particle swarm optimization algorithm for supply chain management problems. In *Metaheuristics for Bi-level Optimization*, pages 69–93. Springer, 2013.

[229] Y. Marinakis, A. Migdalas, and P. M. Pardalos. A new bilevel formulation for the vehicle routing problem and a solution method using a genetic algorithm. *Journal of Global Optimization*, 38(4):555–580, 2007.

[230] R. Martí. Multi-start methods. *Handbook of metaheuristics*, pages 355–368, 2003.

[231] R. Marti, J. L. G. Velarde, and A. Duarte. Heuristics for the bi-objective path dissimilarity problem. *Computers & Operations Research*, 36(11):2905–2912, 2009.

[232] R. Martí, P. Pardalos, and M. G. Resende. Handbook of heuristics, 2017.

[233] R. E. Martínez, E. C. Bravo, W. A. Morales, and J. D. Garcia-Racines. A bi-level multi-objective optimization model for the planning, design and operation of smart grid projects. case study: an islanded microgrid. *International Journal of Energy Economics and Policy*, 10(4):325, 2020.

[234] E. Martins de Sá, R. Camargo, and G. Miranda. An improved benders decomposition algorithm for the tree of hubs location problem. *European Journal of Operational Research*, 226:185-202, 04 2013.

[235] R. Mathieu, L. Pittard, and G. Anandalingam. Genetic algorithm based approach to bi-level linear programming. *RAIRO-Operations Research-Recherche Opérationnelle*, 28(1):1–21, 1994.

[236] A. L. Medaglia, J. G. Villegas, and D. M. Rodríguez-Coca. Hybrid biobjective evolutionary algorithms for the design of a hospital waste management network. *Journal of Heuristics*, 15(2):153–176, 2009.

[237] J.-A. Mejía-de Dios and E. Mezura-Montes. A metaheuristic for bilevel optimization using tykhonov regularization and the quasi-newton method. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pages 3134–3141. IEEE, 2019.

[238] J.-A. Mejía-de Dios and E. Mezura-Montes. A surrogate-assisted metaheuristic for bilevel optimization. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, pages 629–635, 2020.

[239] J.-A. Mejía-de Dios, E. Mezura-Montes, and P. Toledo-Hernández. Pseudo-feasible solutions in evolutionary bilevel optimization: Test problems and performance assessment. *Applied Mathematics and Computation*, 412:126577, 2022.

[240] M. Memarpour, A. Hafezalkotob, M. Khalilzadeh, A. Saghaei, and R. Soltani. Determining the optimum investment portfolios in the iranian banking network base on bi-level game using the markowitz optimization model by firefly algorithm. *Journal of Industrial Strategic Management*, 6(3):1–26, 2021.

[241] R. Menasri, A. Nakib, H. Oulhadj, B. Daachi, P. Siarry, and G. Hains. Path planning for redundant manipulators using metaheuristic for bilevel optimization and maximum of manipulability. In *2013 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 145–150. IEEE, 2013.

[242] R. Menasri, A. Nakib, B. Daachi, H. Oulhadj, and P. Siarry. A trajectory planning of redundant manipulators based on bilevel optimization. *Applied Mathematics and Computation*, 250:934–947, 2015.

[243] M. Meraklı and H. Yaman. A capacitated hub location problem under hose demand uncertainty. *Computers & Operations Research*, 88:58–70, 2017.

[244] M. Mesbah, M. Sarvi, and G. Currie. Optimization of transit priority in the transportation network using a genetic algorithm. *IEEE Transactions on Intelligent Transportation Systems*, 12(3):908–919, 2011.

[245] M. Mesquita-Cunha, J. R. Figueira, and A. P. Barbosa-Póvoa. New - constraint methods for multi-objective integer linear programming: A pareto front representation approach. *European Journal of Operational Research*, 306(1):286–307, 2023.

[246] E. Miandoabchi and R. Z. Farahani. Optimizing reserve capacity of urban road networks in a discrete network design problem. *Advances in Engineering Software*, 42(12):1041–1050, 2011.

[247] H. Min and Z. Guo. Developing bi-level equilibrium models for the global container transportation network from the perspectives of multiple stakeholders. *International Journal of Logistics Systems and Management*, 6(4):362–379, 2010.

[248] A. Mohamadi, S. Ebrahimnejad, R. Soltani, and M. Khalilzadeh. An integrated approach based on a bi-level genetic algorithm and a combined zonelp for the facility layout problem. *South African Journal of Industrial Engineering*, 30(4):87–101, 2019.

[249] M. Mohammadi, P. Jula, and R. Tavakkoli-Moghaddam. Reliable single-allocation hub location problem with disruptions. *Transportation Research Part E: Logistics and Transportation Review*, 123:90–120, 2019.

[250] M. Mokhlesian and S. H. Zegordi. Application of multidivisional bi-level programming to coordinate pricing and inventory decisions in a multiproduct competitive supply chain. *The International Journal of Advanced Manufacturing Technology*, 71(9):1975–1989, 2014.

[251] D. Molina, J. Poyatos, J. D. Ser, S. García, A. Hussain, and F. Herrera. Comprehensive taxonomies of nature-and bio-inspired optimization: Inspiration versus algorithmic behavior, critical analysis recommendations. *Cognitive Computation*, 12(5):897–939, 2020.

[252] M. Momenitabar, Z. Dehdari Ebrahimi, and J. Mattson. Robust electric transit route network design problem considering energy storage technology and demand charges: Model and application. *Available at SSRN 4232774*.

[253] J. T. Moore and J. F. Bard. The mixed integer linear bilevel programming problem. *Operations Research*, 38(5):911–921, 1990.

[254] M. M. Nasiri, V. Mahmoodian, A. Rahbari, and S. Farahmand. A modified genetic algorithm for the capacitated competitive facility location problem with the partial demand satisfaction. *Computers & Industrial Engineering*, 124:435–448, 2018.

[255] S. Nesmachnow. An overview of metaheuristics: accurate and efficient methods for optimisation. *International Journal of Metaheuristics*, 3(4):320–347, 2014.

[256] S. Nickel, A. Schöbel, and T. Sonneborn. Hub location problems in urban traffic networks. In *Mathematical methods on optimization in transportation systems*, pages 95–107. Springer, 2001.

[257] I. Nishizaki and M. Sakawa. Computational methods through genetic algorithms for obtaining stackelberg solutions to two-level integer programming problems. *Cybernetics and Systems: An International Journal*, 36(6):565–579, 2005.

[258] I. Nishizaki, M. Sakawa, K. Niwa, and Y. Kitaguchi. A computational method using genetic algorithms for obtaining stackelberg solutions to two-level linear programming problems. *Electronics and Communications in Japan (Part III: Fundamental Electronic Science)*, 85(6):55–62, 2002.

[259] S. Nucamendi-Guillén, D. Dávila, J.-F. Camacho-Vallejo, and R. G. González-Ramírez. A discrete bilevel brain storm algorithm for solving a sales territory design problem: a case study. *Memetic Computing*, 10(4):441–458, 2018.

[260] T. Num, F. Legillon, A. Liefooghe, and E.-G. Talbi. Cobra: A coevolutionary metaheuristic for bi-level optimization. *Studies in Computational Intelligence*, 482, 09 2011.

[261] V. Oduguwa and R. Roy. Bi-level optimisation using genetic algorithm. In *Proceedings 2002 IEEE International Conference on Artificial Intelligence Systems (ICAIS 2002)*, pages 322–327. IEEE, 2002.

[262] M. O'Kelly. The location of interacting hub facilities. *Transportation Science*, 20: 92–106, 05 1986.

[263] M. O'Kelly. A quadratic integer program for the location of interacting hub facilities. *European Journal of Operational Research*, 32:393–404, 12 1987.

[264] M. Osman, W. El-Wahed, M. M. Shafei, and H. B. Wahab. A solution methodology of bi-level linear programming based on genetic algorithm. *Journal of Mathematics and Statistics*, 5(4):352, 2009.

[265] A. A. Panin, M. G. Pashchenko, and A. V. Plyasunov. Bilevel competitive facility location and pricing problems. *Automation and Remote Control*, 75(4):715–727, 2014.

[266] S. P. Parvasi, M. Mahmoodjanloo, and M. Setak. A bi-level school bus routing problem with bus stops selection and possibility of demand outsourcing. *Applied Soft Computing*, 61:222–238, 2017.

[267] S. Paul and Z. H. Rather. A new bi-level planning approach to find economic and reliable layout for large-scale wind farm. *IEEE Systems Journal*, 13(3):3080–3090, 2019.

[268] M. Peker and B. Y. Kara. The p-hub maximal covering problem and extensions for gradual decay functions. *Omega*, 54:158–172, 2015.

[269] Z. Peng, C. Wang, W. Xu, and J. Zhang. Research on location-routing problem of maritime emergency materials distribution based on bi-level programming. *Mathematics*, 10(8):1243, 2022.

[270] A. F. Pérez Posada, J. G. Villegas, and J. M. López-Lezama. A scatter search heuristic for the optimal location, sizing and contract pricing of distributed generation in electric distribution systems. *Energies*, 10(10):1449, 2017.

[271] L. Pessoa, A. Santos, and M. Resende. A biased random-key genetic algorithm for the tree of hubs location problem. *Optimization Letters*, 11:1371-1384, 10 2017.

[272] S. Pineda and J. M. Morales. Solving linear bilevel problems using big-ms: Not all that glitters is gold. *IEEE Transactions on Power Systems*, 34(3):2469–2471, 2019.

[273] F. Plastria. Static competitive facility location: an overview of optimisation approaches. *European Journal of Operational Research*, 129(3):461–470, 2001.

[274] J.-Y. Potvin and M. Gendreau. *Handbook of Metaheuristics*. Springer, 2018.

[275] M. Pozo, J. Puerto, and A. Chía. The ordered median tree of hubs location problem. *TOP*, 06 2020.

[276] Z. Qing-cheng and Y. Zhong-zhen. A bi-level programming model and its algorithm for scheduling at a container terminal. In *2006 International Conference on Management Science and Engineering*, pages 402–406. IEEE, 2006.

[277] G. R. Raidl. A unified view on hybrid metaheuristics. In *International Workshop on Hybrid Metaheuristics*, pages 1–12. Springer, 2006.

[278] J. Rajesh, K. Gupta, H. S. Kusumakar, V. K. Jayaraman, and B. D. Kulkarni. A tabu search based approach for solving a class of bilevel programming problems in chemical engineering. *Journal of Heuristics*, 9(4):307–319, 2003.

[279] G. Ren, Z. Huang, Y. Cheng, X. Zhao, and Y. Zhang. An integrated model for evacuation routing and traffic signal optimization with background demand uncertainty. *Journal of Advanced Transportation*, 47(1):4–27, 2013.

[280] M. G. Resende and C. C. Ribeiro. Greedy randomized adaptive search procedures: Advances and applications. *Handbook of Metaheuristics*, 146:281–317, 2010.

[281] M. G. Resende, C. C. Ribeiro, F. Glover, and R. Martí. Scatter search and path-relinking: Fundamentals, advances, and applications. *Handbook of Metaheuristics*, pages 87–107, 2010.

[282] J. P. Ribeiro and A. Barbosa-Povoa. Supply chain resilience: Definitions and quantitative modelling approaches–a literature review. *Computers & Industrial Engineering*, 115:109–122, 2018.

[283] N. Rikhtegar, M. Keshtgari, O. Bushehrian, and G. Pujolle. Bite: a dynamic bi-level traffic engineering model for load balancing and energy efficiency in data center networks. *Applied Intelligence*, 51(7):4623–4648, 2021.

[284] R. M. Rizk-Allah and M. A. Abo-Sinna. A comparative study of two optimization approaches for solving bi-level multi-objective linear fractional programming problem. *OPSEARCH*, 58(2):374–402, 2021.

[285] I. Rodriguez-Martin and J. J. Salazar-Gonzalez. Solving a capacitated hub location problem. *European Journal of Operational Research*, 184(2):468–479, 2008.

[286] E. Ruano-Daza, C. Cobos, J. Torres-Jimenez, M. Mendoza, and A. Paz. A multiobjective bilevel approach based on global-best harmony search for defining optimal routes and frequencies for bus rapid transit systems. *Applied Soft Computing*, 67: 567–583, 2018.

[287] Z. Saeidi-Mobarakeh, R. Tavakkoli-Moghaddam, M. Navabakhsh, and H. Amoozad-Khalili. A bi-level meta-heuristic approach for a hazardous waste management problem. *International Journal of Engineering*, 33(7):1304–1310, 2020.

[288] E. Saghehei, A. Memariani, and A. Bozorgi-Amiri. Implementing solution algorithms for a bi-level optimization to the emergency warehouse location-allocation problem. *International Journal of Supply and Operations Management*, 2022.

[289] G. Saharidis, M. Golias, M. Boile, S. Theofanis, and M. Ierapetritou. The berth scheduling problem with customer differentiation: a new methodological approach based on hierarchical optimization. *The International Journal of Advanced Manufacturing Technology*, 46:377–393, 2010.

[290] G. K. Saharidis, A. J. Conejo, and G. Kozanidis. Exact solution methodologies for linear and (mixed) integer bilevel programming. *Metaheuristics for Bi-level Optimization*, pages 221–245, 2013.

[291] K. H. Sahin and A. R. Ciric. A dual temperature simulated annealing approach for solving bilevel programming problems. *Computers & Chemical Engineering*, 23(1): 11–25, 1998.

[292] R. Said, M. Elarbi, S. Bechikh, and L. Ben Said. Solving combinatorial bi-level optimization problems using multiple populations and migration schemes. *Operational Research*, pages 1–39, 2021.

[293] R. Said, M. Elarbi, S. Bechikh, C. A. C. Coello, and L. B. Said. Discretization-based feature selection as a bi-level optimization problem. *IEEE Transactions on Evolutionary Computation*, 2022.

[294] M. Sakawa, Ichiro Nishizaki. Computational methods through genetic algorithms for obtaining stackelberg solutions to two-level mixed zero-one programming problems. *Cybernetics & Systems*, 31(2):203–221, 2000.

[295] D. Salas and A. Svensson. Existence of solutions for deterministic bilevel games under a general bayesian approach. *SIAM Journal on Optimization*, 33(3):2311–2340, 2023.

[296] E. Samaniego Mena and P. Novoa-Hernández. A hybrid approach for solving dynamic bi-level optimization problems. *Computación y Sistemas*, 22, 07 2018.

[297] D. R. Santos-Peñate, C. M. Campos-Rodríguez, and J. A. Moreno-Pérez. A kernel search matheuristic to solve the discrete leader-follower location problem. *Networks and Spatial Economics*, 20:73–98, 2020.

[298] M. Sarhani, S. Voß, and R. Jovanovic. Initialization of metaheuristics: comprehensive review, critical analysis, and research directions. *International Transactions in Operational Research*, 2022.

[299] G. A. Segundo, R. A. Krohling, and R. C. Cosme. A differential evolution approach for solving constrained min–max optimization problems. *Expert Systems with Applications*, 39(18):13440–13450, 2012.

[300] W. Shan, Q. Yan, C. Chen, M. Zhang, B. Yao, and X. Fu. Optimization of competitive facility location for chain stores. *Annals of Operations Research*, 273:187–205, 2019.

[301] C. Shi. *Linear bilevel programming technology: models and algorithms*. PhD thesis, 2005.

[302] J. Shouwen, L. Di, C. Zhengrong, and G. Dong. Integrated scheduling in automated container terminals considering agv conflict-free routing. *Transportation Letters*, 13 (7):501–513, 2021.

[303] E. A. Silver, R. Victor, V. Vidal, and D. de Werra. A tutorial on heuristic methods. *European Journal of Operational Research*, 5(3):153–162, 1980.

[304] H. K. Singh, M. M. Islam, T. Ray, and M. Ryan. Nested evolutionary algorithms for computationally expensive bilevel optimization problems: Variants and their systematic analysis. *Swarm and Evolutionary Computation*, 48:329–344, 2019.

[305] A. Sinha. Bilevel multi-objective optimization problem solving using progressively interactive emo. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 269–284. Springer, 2011.

[306] A. Sinha and K. Deb. Towards understanding evolutionary bilevel multi-objective optimization algorithm. *IFAC Proceedings Volumes*, 42(2):338–343, 2009.

[307] A. Sinha, P. Malo, and K. Deb. Efficient evolutionary algorithm for single-objective bilevel optimization. *arXiv preprint arXiv:1303.3901*, 2013.

[308] A. Sinha, P. Malo, A. Frantsev, and K. Deb. Multi-objective stackelberg game between a regulating authority and a mining company: A case study in environmental economics. In *2013 IEEE congress on Evolutionary Computation*, pages 478–485. IEEE, 2013.

[309] A. Sinha, P. Malo, and K. Deb. An improved bilevel evolutionary algorithm based on quadratic approximations. In *2014 IEEE Congress on Evolutionary Computation (CEC)*, pages 1870–1877. IEEE, 2014.

[310] A. Sinha, P. Malo, A. Frantsev, and K. Deb. Finding optimal strategies in a multi-period multi-leader–follower stackelberg game using an evolutionary algorithm. *Computers & Operations Research*, 41:374–385, 2014.

[311] A. Sinha, P. Malo, P. Xu, and K. Deb. A bilevel optimization approach to automated parameter tuning. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*, pages 847–854, 2014.

[312] A. Sinha, P. Malo, and K. Deb. Towards understanding bilevel multi-objective optimization with deterministic lower level decisions. In *EMO (1)*, pages 426–443, 2015.

[313] A. Sinha, P. Malo, and K. Deb. Transportation policy formulation as a multi-objective bilevel optimization problem. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pages 1651–1658. IEEE, 2015.

[314] A. Sinha, P. Malo, K. Deb, P. Korhonen, and J. Wallenius. Solving bilevel multicriterion optimization problems with lower level decision uncertainty. *IEEE Transactions on Evolutionary Computation*, 20(2):199–217, 2015.

[315] A. Sinha, P. Malo, and K. Deb. Solving optimistic bilevel programs by iteratively approximating lower level optimal value function. In *2016 IEEE Congress on Evolutionary Computation (CEC)*, pages 1877–1884. IEEE, 2016.

[316] A. Sinha, P. Malo, and K. Deb. Evolutionary bilevel optimization: An introduction and recent advances. *Recent Advances in Evolutionary Multi-Objective Optimization*, pages 71–103, 2017.

[317] A. Sinha, P. Malo, and K. Deb. A review on bilevel optimization: From classical to evolutionary approaches and applications. *IEEE Transactions on Evolutionary Computation*, 22(2):276–295, 2017.

[318] A. Sinha, Z. Lu, K. Deb, and P. Malo. Bilevel optimization based on iterative approximation of multiple mappings. *Journal of Heuristics*, 26:151–185, 2020.

[319] M. Sipper, W. Fu, K. Ahuja, and J. H. Moore. Investigating the parameter space of evolutionary algorithms. *BioData mining*, 11(1):1–14, 2018.

[320] A. N. Sloss and S. Gustafson. *2019 Evolutionary Algorithms Review*, pages 307–344. Springer International Publishing, Cham, 2020.

[321] I. Soares, M. J. Alves, and C. H. Antunes. A bi-level optimization approach to define dynamic tariffs with variable prices and periods in the electricity retail market. In *Advances in Evolutionary and Deterministic Methods for Design, Optimization and Control in Engineering and Sciences*, pages 1–16. Springer, 2021.

[322] L. Song, T. Cherrett, and W. Guan. Study on berth planning problem in a container seaport: Using an integrated programming approach. *Computers & Industrial Engineering*, 62(1):119–128, 2012.

[323] X. Song, H. Lin, G. De, H. Li, X. Fu, and Z. Tan. An energy optimal dispatching model of an integrated energy system based on uncertain bilevel programming. *Energies*, 13(2):477, 2020.

[324] K. Sörensen. Metaheuristicsâ[[[ERROR FOR PACKAGE inputenc]]][[[ERROR FOR PACKAGE inputenc]]]the metaphor exposed. *International Transactions in Operational Research*, 22(1):3–18, 2015.

[325] K. Sörensen and F. Glover. Metaheuristics. *Encyclopedia of Operations Research and Management Science*, 62:960–970, 2013.

[326] K. Sörensen and M. Sevaux. Ma— pm: memetic algorithms with population management. *Computers & Operations Research*, 33(5):1214–1225, 2006.

[327] B. Soylu and H. Katip. A multiobjective hub-airport location problem for an airline network design. *European Journal of Operational Research*, 277(2):412–425, 2019.

[328] H. v. Stackelberg et al. Theory of the market economy. 1952.

[329] H. Stegherr, M. Heider, and J. Hähner. Classifying metaheuristics: Towards a unified multi-level classification system. *Natural Computing*, 21(2):155–171, 2022.

[330] T. Stützle and R. Ruiz. Iterated greedy. *Handbook of Heuristics*, pages 547–577, 2018.

[331] D. Sun, R. F. Benekohal, and S. T. Waller. Bi-level programming formulation and heuristic solution approach for dynamic traffic signal optimization. *Computer-Aided Civil and Infrastructure Engineering*, 21(5):321–333, 2006.

[332] S. Sun, B. Song, P. Wang, H. Dong, and X. Chen. An adaptive bi-level task planning strategy for multi-usvs target visitation. *Applied Soft Computing*, 115:108086, 2022.

[333] V. Suryan, A. Sinha, P. Malo, and K. Deb. Handling inverse optimal control problems using evolutionary bilevel optimization. In *2016 IEEE Congress on Evolutionary Computation (CEC)*, pages 1893–1900. IEEE, 2016.

[334] J. Swan, S. Adriaensen, A. E. Brownlee, K. Hammond, C. G. Johnson, A. Kheiri, F. Krawiec, J. J. Merelo, L. L. Minku, E. Özcan, et al. Metaheuristics "in the large". *European Journal of Operational Research*, 297(2):393–406, 2022.

[335] S. Tabrizi, S. H. Ghodsypour, and A. Ahmadi. Modelling three-echelon warm-water fish supply chain: A bi-level optimization approach under nash–cournot equilibrium. *Applied Soft Computing*, 71:1035–1053, 2018.

[336] H. Tagawa, T. Kawasaki, and S. Hanaoka. Exploring the factors influencing the cost-effective design of hub-and-spoke and point-to-point networks in maritime transport using a bi-level optimization model. *The Asian Journal of Shipping and Logistics*, 37(2):192–203, 2021.

[337] S. Tahernejad, T. K. Ralphs, and S. T. DeNegre. A branch-and-cut algorithm for mixed integer bilevel linear optimization problems and its implementation. *Mathematical Programming Computation*, 12:529–568, 2020.

[338] M. Takebayashi and S. Hanaoka. Efficient inter-port cooperation considering port congestion and port charge. *Maritime Transport Research*, 2:100011, 2021.

[339] E.-G. Talbi. *A taxonomy of metaheuristics for bi-level optimization*. Springer, 2013.

[340] E.-G. Talbi. A unified taxonomy of hybrid metaheuristics with mathematical programming, constraint programming and machine learning. In *Hybrid Metaheuristics*, pages 3–76. Springer, 2013.

[341] Q. Tang, Z. Fu, and M. Qiu. A bilevel programming model and algorithm for the static bike repositioning problem. *Journal of Advanced Transportation*, 2019, 2019.

[342] M. A. Tawhid and G. Paluck. Solving linear bilevel programming via particle swarm algorithm with heuristic pattern search. *Information Sciences Letters*, 6(1):1, 2021.

[343] S. L. Tilahun, S. M. Kassa, and H. C. Ong. A new algorithm for multilevel optimization problems using evolutionary strategy, inspired by natural adaptation. In *Pacific Rim International Conference on Artificial Intelligence*, pages 577–588. Springer, 2012.

[344] Z. Tingfa, Z. Liangzhi, Z. Licai, and Z. Qingzhen. Container transport network optimization model under container port competition. In *2008 IEEE International Conference on Automation and Logistics*, pages 2224–2228. IEEE, 2008.

[345] H. Topcuoglu, F. Corut, M. Ermis, and G. Yilmaz. Solving the uncapacitated hub location problem using genetic algorithms. *Computers & Operations Research*, 32 (4):967–984, 2005.

[346] A. Tsoukalas, W. Wiesemann, B. Rustem, et al. Global optimisation of pessimistic bi-level problems. *Lectures on Global Optimization*, 55:215–243, 2009.

[347] G. Ünlü. A linear bilevel programming algorithm based on bicriteria programming. *Computers & Operations Research*, 14(2):173–179, 1987.

[348] N. Van Eck and L. Waltman. Software survey: Vosviewer, a computer program for bibliometric mapping. *Scientometrics*, 84(2):523–538, 2010.

[349] A. D. Vasconcelos, C. D. Nassi, and L. A. Lopes. The uncapacitated hub location problem in networks under decentralized management. *Computers & Operations Research*, 38(12):1656–1666, 2011.

[350] J. G. Villegas, F. Palacios, and A. L. Medaglia. Solution methods for the bi-objective (cost-coverage) unconstrained facility location problem with an illustrative example. *Annals of Operations Research*, 147(1):109–141, 2006.

[351] Z. Wan, G. Wang, and B. Sun. A hybrid intelligent algorithm by combining particle swarm optimization with chaos searching technique for solving nonlinear bilevel programming problems. *Swarm and Evolutionary Computation*, 8:26–32, 2013.

[352] S. Wandelt, W. Dai, J. Zhang, and X. Sun. Toward a reference experimental benchmark for solving hub location problems. *Transportation Science*, 56(2):543–564, 2022.

[353] C. Wang, C. Jin, and Z. Li. Bilevel programming model of low energy consumption agv scheduling problem at automated container terminal. In *2019 IEEE International Conference on Smart Manufacturing, Industrial & Logistics Engineering (SMILE)*, pages 195–199. IEEE, 2019.

[354] D. Wang, G. Du, et al. A self adaptive penalty function based genetic algorithm for value-bilevel programming problem. *International Journal on Computer Science and Engineering*, 3(2):136–146, 2014.

[355] G. Wang, Z. Wan, X. Wang, and D. Fang. Genetic algorithm for solving quadratic bilevel programming problem. *Wuhan University Journal of Natural Sciences*, 12 (3):421–425, 2007.

[356] G. Wang, Z. Wan, X. Wang, and Y. Lv. Genetic algorithm based on simplex method for solving linear-quadratic bilevel programming problem. *Computers & Mathematics with Applications*, 56(10):2550–2555, 2008.

[357] G. Wang, L. Ma, and J. Chen. A bilevel improved fruit fly optimization algorithm for the nonlinear bilevel programming problem. *Knowledge-Based Systems*, 138(C): 113–123, 2017.

[358] G.-m. Wang, X.-j. Wang, Z.-p. Wan, and S.-h. Jia. An adaptive genetic algorithm for solving bilevel linear programming problem. *Applied Mathematics and Mechanics*, 28(12):1605–1612, 2007.

[359] J. Y. Wang, M. Ehrgott, K. N. Dirks, and A. Gupta. A bilevel multi-objective road pricing model for economic, environmental and health sustainability. *Transportation Research Procedia*, 3:393–402, 2014.

[360] K. Wang, J. Li, X. Yan, L. Huang, X. Jiang, Y. Yuan, R. Ma, and R. R. Negenborn. A novel bi-level distributed dynamic optimization method of ship fleets energy consumption. *Ocean Engineering*, 197:106802, 2020.

[361] Y. Wang, Y.-C. Jiao, and H. Li. An evolutionary algorithm for solving nonlinear bilevel programming based on a new constraint-handling scheme. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 35(2): 221–232, 2005.

[362] Y. Wang, H. Li, and C. Dang. A new evolutionary algorithm for a class of nonlinear bilevel programming problems and its global convergence. *INFORMS Journal on Computing*, 23(4):618–629, 2011.

[363] J. Watada, A. Roy, B. Wang, S. C. Tan, and B. Xu. An artificial bee colony-based double layered neural network approach for solving quadratic bi-level programming problems. *IEEE Access*, 8:21549–21564, 2020.

[364] U. Wen and A. Huang. A simple tabu search method to solve the mixed-integer linear bilevel programming problem. *European Journal of Operational Research*, 88 (3):563–571, 1996.

[365] D. Weyland. A critical analysis of the harmony search algorithm. how not to solve sudoku. *Operations Research Perspectives*, 2:97–105, 2015.

[366] G. Whittaker, R. Färe, S. Grosskopf, B. Barnhart, M. Bostian, G. Mueller-Warrant, and S. Griffith. Spatial targeting of agri-environmental policy using bilevel evolutionary optimization. *Omega*, 66:15–27, 2017.

[367] W. Wiesemann, A. Tsoukalas, P.-M. Kleniati, and B. Rustem. Pessimistic bilevel optimization. *SIAM Journal on Optimization*, 23(1):353–380, 2013.

[368] C. Wohlin, M. Kalinowski, K. R. Felizardo, and E. Mendes. Successful combination of database search and snowballing for identification of primary studies in systematic literature studies. *Information and Software Technology*, 147:106908, 2022.

[369] C.-P. Wu. A hybrid technique for global optimization of hierarchical systems. In *1996 IEEE International Conference on Systems, Man and Cybernetics. Information Intelligence and Systems*, volume 3, pages 1706–1711. IEEE, 1996.

[370] W. Xiao, G. Du, Y. Zhang, and X. Liu. Coordinated optimization of low-carbon product family and its manufacturing process design by a bilevel game-theoretic model. *Journal of Cleaner Production*, 184:754–773, 2018.

[371] J. Xu, Y. Tu, and Z. Zeng. A nonlinear multiobjective bilevel model for minimum cost network flow problem in a large-scale construction project. *Mathematical Problems in Engineering*, 2012, 2012.

[372] T. Xu, H. Wei, and G. Hu. Study on continuous network design problem using simulated annealing and genetic algorithm. *Expert Systems with Applications*, 36 (2):1322–1328, 2009.

[373] X.-D. Xu, W.-H. Chang, and J. Liu. Resource allocation optimization model of collaborative logistics network based on bilevel programming. *Scientific Programming*, 2017, 2017.

[374] T. Yamada, B. F. Russ, J. Castro, and E. Taniguchi. Designing multimodal freight transport networks: A heuristic approach and applications. *Transportation Science*, 43(2):129–143, 2009.

[375] G. Yang, Y. Huang, Y. Fu, B. Huang, S. Sheng, L. Mao, S. Huang, Y. Xu, J. Le, Y. Ouyang, et al. Parcel locker location based on a bilevel programming model. *Mathematical Problems in Engineering*, 2020:1–12, 2020.

[376] R.-L. Yang and C.-P. Wu. Global solution of nonlinear bilevel programming problems based on simulated annealing and neural network. *IFAC Proceedings Volumes*, 28(10):623–628, 1995.

[377] X.-S. Yang. *Nature-inspired optimization algorithms*. Academic Press, 2020.

[378] Y. Yang, M. Zhong, Y. Dessouky, and O. Postolache. An integrated scheduling method for agv routing in automated container terminals. *Computers & Industrial Engineering*, 126:482–493, 2018.

[379] Z. Yang and K. Chen. Optimization of shipping network of trunk and feeder lines for inter-regional and intra-regional container transport. *Journal of the Eastern Asia Society for Transportation Studies*, 8:694–705, 2010.

[380] J. Ye, Y. Jiang, J. Chen, Z. Liu, and R. Guo. Joint optimisation of transfer location and capacity for a capacitated multimodal transport network with elastic demand: a bi-level programming model and paradoxes. *Transportation Research Part E: Logistics and Transportation Review*, 156(C):102540, 2021.

[381] Y. Yin. Genetic-algorithms-based approach for bilevel programming models. *Journal of Transportation Engineering*, 126(2):115–120, 2000.

[382] Y. Yin. Multiobjective bilevel optimization for transportation planning and management problems. *Journal of Advanced Transportation*, 36(1):93–105, 2002.

[383] F. Yu, F. Teng, Q. Shan, T. Li, and Y. Xiao. Continuous berth allocation considering carbon emission and uncertainty. In *2022 4th International Conference on Data-driven Optimization of Complex Systems (DOCS)*, pages 1–6. IEEE, 2022.

[384] J. Yu, G. Tang, and X. Song. Collaboration of vessel speed optimization with berth allocation and quay crane assignment considering vessel service differentiation. *Transportation Research Part E: Logistics and Transportation Review*, 160:102651, 2022.

[385] W. Yu. A leader-follower model for discrete competitive facility location problem under the partially proportional rule with a threshold. *PloS one*, 14(12):e0225693, 2019.

[386] W. Yu. Robust model for discrete competitive facility location problem with the uncertainty of customer behaviors. *Optimization Letters*, 14(8):2107–2125, 2020.

[387] X. Yu and M. Gen. *Introduction to evolutionary algorithms*. Springer Science & Business Media, 2010.

[388] G. Yuan, Y. Gao, B. Ye, and R. Huang. Real-time pricing for smart grid with multi-energy microgrids and uncertain loads: a bilevel programming method. *International Journal of Electrical Power & Energy Systems*, 123:106206, 2020.

[389] P. Yulong and C. Yusheng. Study on bi-level planning model & algorithm optimizing highway network layout. In *Proceedings of the Eastern Asia Society for Transportation Studies*, volume 5, pages 750–761. Citeseer, 2005.

[390] G. Zang, M. Xu, and Z. Gao. High-occupancy vehicle lanes and tradable credits scheme for traffic congestion management: A bilevel programming approach. *Promet-Traffic&Transportation*, 30(1):1–10, 2018.

[391] H. Zhang, Q. Zhang, and W. Chen. Bi-level programming model of truck congestion pricing at container terminals. *Journal of Ambient Intelligence and Humanized Computing*, 10:385–394, 2019.

[392] M. Zhang, B. Wiegmans, and L. Tavasszy. Optimization of multimodal networks including environmental costs: a model and findings for transport policy. *Computers in industry*, 64(2):136–145, 2013.

[393] T. Zhang and L. Liu. Container transportation network bilevel model analysis. In *Logistics: The Emerging Frontiers of Transportation and Development in China*, pages 293–298. 2009.

[394] T. Zhang, T. Hu, Y. Zheng, and X. Guo. An improved particle swarm optimization for solving bilevel multiobjective programming problem. *Journal of Applied Mathematics*, 2012, 2012.

[395] T. Zhang, T. Hu, X. Guo, Z. Chen, and Y. Zheng. Solving high dimensional bilevel multiobjective programming problem using a hybrid particle swarm optimization algorithm with crossover operator. *Knowledge-Based Systems*, 53:13–19, 2013.

[396] T. Zhang, Z. Chen, Y. Zheng, and J. Chen. An improved simulated annealing algorithm for bilevel multiobjective programming problems with application. *Journal of Nonlinear Sciences and Applications*, 9(6):3672–3685, 2016.

[397] Y. Zhang, W. H. Lam, and A. Sumalee. Transit schedule design in dynamic transit network with demand and supply uncertainties. In *Proceedings of the Eastern Asia Society for Transportation Studies Vol. 7 (The 8th International Conference of Eastern Asia Society for Transportation Studies, 2009)*, pages 250–250. Eastern Asia Society for Transportation Studies, 2009.

[398] Z. Zhao and X. Gu. Particle swarm optimization based algorithm for bilevel programming problems. In *Sixth International Conference on Intelligent Systems Design and Applications*, volume 2, pages 951–956. IEEE, 2006.

[399] F. Zhong, B. Yuan, and B. Li. A hybrid evolutionary algorithm for multiobjective variation tolerant logic mapping on nanoscale crossbar architectures. *Applied Soft Computing*, 38(C):955–966, 2016.

[400] M. Zhong and L. Lin. Location of container port investment based on bi-level programming model. *Contemporary Logistics*, (2):113, 2011.

[401] Y. Zhou, Y. Kou, and M. Zhou. Bilevel memetic search approach to the soft-clustered vehicle routing problem. *Transportation Science*, 2022.

[402] X. Zhu, Q. Yu, and X. Wang. A hybrid differential evolution algorithm for solving nonlinear bilevel programming with linear constraints. In *2006 5th IEEE International Conference on Cognitive Informatics*, volume 1, pages 126–131. IEEE, 2006.

[403] E. Ziar, M. Seifbarghy, M. Bashiri, and B. Tjahjono. An efficient environmentally friendly transportation network design via dry ports: a bi-level programming approach. *Annals of Operations Research*, 322(2):1143–1166, 2023.

[404] E. Zitzler, M. Laumanns, and L. Thiele. Spea2: Improving the strength pareto evolutionary algorithm. *TIK report*, 103, 2001.

[405] A. M. Zobaa, S. H. A. Aleem, and H. K. Youssef. Bi-level damped double-tuned harmonic passive filters design: Multi-criteria decision-making analysis. *Ain Shams Engineering Journal*, page 102082, 2022.

Apéndice A

Acrónimos Metaheurísticas

Este Apéndice muestra los acrónimos de las metaheurísticas representadas en la Figura . Los acrónimos siguen la nomenclatura más común utilizada en los artículos revisados.

Acrónimo	Nombre
ABC	Artificial Bee Colony
ALNS	Adaptive large neighborhood search
ACO	Ant Colony Optimization
BSA	Brainstorm Algorithm
BSBM	Bilevel School Bus Metaheuristic
CE	Cross Entropy
COEA	CoEvolutionary Algorithm
CRA	Chemical Reaction Algorithm
DE	Differential evolution
DFA	Drangonfly Algorithm
EA	Evolutionary Algorithm
ES	Evolutionary Strategy
FFA	Firefly Algorithm
FOA	Fruit Fly Optimization Algorithm
GA	Genetic Algorithm
GRASP	Greedy Randomized Adaptive Search Procedure
GWO	Grey Wolf Optimization
HS	Harmony Search
ICA	Imperialist Competitive Algorithm
IGA	Iterated greedy algorithm
ILS	Iterated Local Search
JA	Jaya Algorithm
MA	Memetic Algorithm
MAPM	Memetic Algorithm with Population Management
MEA	Metaevolutionary Algorithm
NSGAII	Non-dominated sorting genetic algorithm II
PHA	Path Homotopy Algorithm
PSO	Particle Swarm Algorithm
QBCA	Quasi Newton Bilevel Centers Algorithm
RS	Random Search
SFS	Stochastic Fractal Search
SA	Simulated Annealing
SS	Scatter Search
SSA	Salp Swarm Algorithm
STA	State Transition Algorithm
TS	Tabu Search
VDNS	Variable Descent Neighborhood Search
VNS	Variable Neighborhood Search
VEA	Volcanic Eruption Algorithm
WHO	Wild Horse Optimization
WOA	Whale Optimization Algorithm
WWO	Water Wave Optimization

Tabla A.1: Acrónimos utilizados para las metaheurísticas mostradas

Apéndice B

Abreviación de los Journals y Proceedings con sus nombres completos.

La lista de los nombres completos de las revistas y procedings abreviados en la Figura se presenta a continuación.

Acrónimo	Nombre
EJOR	European Journal of Operational Research
IEEE TEVC	IEEE Transactions on Evolutionary Computation
EVCO	Evolutionary Computation
C&OR	Computers & Operations Research
JOGO	Journal of Global Optimization
MONET	Mobile Networks and Applications
ENGCOMP	Engineering Computation
ANOR	Annals of Operations Research
JCLEP	Journal of Cleaner Production
ASOC	Applied Soft Computing Journal
MEMEC	Memetic Computing
AMC	Applied Mathematics and Computation
ESWA	Expert Systems with Applications
ORIJ	Operational Research International Journal
MPE	Mathematical Problems in Engineering
CyS	Computación y Sistemas
PONE	PLOS ONE
CYB&SYS	Cybernetics and Systems
IJPR	International Journal of Production Research
APPINT	Applied Intelligence
NEURO	Neurocomputing
ENGOPT	Engineering Optimization
SOCO	Soft Computing
ITOR	International Transactions in Operational Research
OMEGA	Omega
IJOC	INFORMS Journal on Computing
IJCNSS	International Journal Communications, Network and Systems Sciences
JSEE	Journal of Systems Engineering and Electronics
JHEUR	Journal of Heuristics
IEEE TSMC	IEEE Transactions on Systems, Man, and Cybernetics

Acrónimo	Nombre
Proc ICMCDM'09	Proceedings of International Conference on Multiple Criteria Decision Making
Proc IFAC'09	Proceedings of the 13th IFAC Symposium on Information Control Problems in Manufacturing
Proc EMO'11	Proceedings of International Conference on Evolutionary Multi-Criterion Optimization
Proc IFAC'12	Proceedings of the 15th IFAC Workshop on Control Applications of Optimization
Proc CEC'13	Proceedings of IEEE International Conference on Evolutionary Computation
Proc CEC'14	Proceedings of IEEE International Conference on Evolutionary Computation
Proc GECCO'14	Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation
Proc CEC'15	Proceedings of IEEE International Conference on Evolutionary Computation
Proc EMO'15	Proceedings of International Conference on Evolutionary Multi-Criterion Optimization
Proc CEC'16	Proceedings of IEEE International Conference on Evolutionary Computation
Proc KES'17	21st International Conference on Knowledge-Based and Intelligent Information & Engineering Systems
Proc CEC'21	Proceedings of IEEE International Conference on Evolutionary Computation
Proc MS'13	Proceedings of Modeling and Simulation in Engineering, Economics, and Management
Proc CEC'11	Proceedings of IEEE International Conference on Evolutionary Computation
Proc ICCS'07	Proceedings of International Conference on Computational Science
Proc ICNC'07	Proceedings of International Conference on Natural Computation
Proc SEAL'07	Proceedings of Asia-Pacific Conference on Simulated Evolution and Learning
Proc CEC'21	Proceedings of IEEE International Conference on Evolutionary Computation
Proc KBIIES'21	Proceedings of International Conference on Knowledge-Based and Intelligent Information & Engineering Systems