UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

PROGRAMA DE MAESTRÍA Y DOCTORADO EN INGENIERÍA
INGENIERÍA DE SISTEMAS – INVESTIGACIÓN DE OPERACIONES

OPTIMIZACIÓN DE RUTAS DE VEHÍCULOS CON ENTREGA Y RECOLECCIÓN CON MÚLTIPLES OBJETIVOS

TESIS
QUE PARA OPTAR POR EL GRADO DE:
DOCTORA EN INGENIERÍA

PRESENTA:
MARÍA DEL CARMEN FERNÁNDEZ GARCÍA

TUTOR PRINCIPAL
DR. MIGUEL ÁNGEL GUTIÉRREZ ANDRADE, UAM - IZTAPALAPA

COMITÉ TUTOR
DRA. IDALIA FLORES DE LA MOTA, FACULTAD DE INGENIERÍA
DR. FELIPE DE JESÚS LARA ROSANO, ICAT

CIUDAD UNIVERSITARIA, CD. MX., MAYO 2019

JURADO ASIGNADO:

Presidente:	Dr. Felipe de Jesús Lara Rosano
Secretario:	Dra. Idalia Flores de la Mota
Vocal:	Dr. Miguel Ángel Gutiérrez Andrade
1 er. Suplente:	Dra. Esther Segura Pérez
2 do. Suplente:	Dr. Abel García Nájera

Ciudad Universitaria, Ciudad de México

TUTOR DE TESIS:

DR. MIGUEL ÁNGEL GUTIÉRREZ ANDRADE

FIRMA

A Héctor y
a Javier

Agradecimientos

Al Dr. Miguel Ángel Gutiérrez Andrade por ser mi mejor profesor en el Posgrado de Ingeniería, por: el tema de tesis, sus enseñanzas, su apoyo y sus observaciones.

Al Dr. Abel García Nájera por sus valiosos comentarios, por el tiempo que me dedicó y por sus preguntas.

A la Dra. Idalia Flores de la Mota por su apoyo a mi ingreso al doctorado, por siempre estar disponible para los reportes de investigación y el examen, por su continua exhortación a publicar y a participar en congresos.

A Esther Segura Pérez por sus sugerencias y apoyo.

Al Dr. Felipe de Jesús Lara Rosano por el título de tesis.

Al Posgrado de Ingeniería por su apoyo para asistir el Congreso Latino-Iberoamericano de Investigación Operativa.

A Aida Huerta por mostrarme cómo hacer el reporte de investigación semestral.

Índice general

Resumen

El problema de la composición y tamaño de la flota para el problema de rutas de vehículos (PCTF) considera una flota heterogénea de vehículos con diferentes capacidades y costos. El PCTF consiste en determinar la mejor composición de la flota así como el conjunto de rutas que minimiza el costo total.

En el problema de la composición y tamaño de la flota para el problema de rutas de vehículos con recolecciones (PCTFR) se tiene una flota de vehículos heterogéneos con diferentes costos fijos de acuerdo a sus tamaños. Hay tanto clientes de entrega como clientes de recolección, los primeros deben ser atendidos antes de los segundos. El objetivo del PCTFR es diseñar un conjunto de rutas que minimice el costo total.

Para resolver el PCTFR se han empleado métodos heurísticos y metaheurísticas. Entre las metaheurísticas, la Búsqueda Dispersa (BD) ha mostrado ser adecuada para resolverlo. La BD emplea un conjunto de soluciones que pueden representar concesiones entre objetivos múltiples.

Esta tesis presenta un algoritmo de BD que resuelve el PCTF y el PCTFR. El algoritmo se probó en casos de prueba de la literatura y encontró la solución óptima de algunos de éstos. En algunos casos de prueba del PCTFR, se encontraron soluciones que superan a las mejores soluciones conocidas. Asimismo, propone un algoritmo de BD multiobjetivo para resolver las dos variantes antes mencionadas y muestra los resultados obtenidos en casos de prueba de la literatura.

Capítulo 1

Introducción

El problema de rutas de vehículos (PRV) fue propuesto por Dantzig y Ramser en 1959. Un objetivo común del PRV es diseñar un conjunto de rutas de costo mínimo que satisfagan las demandas de los clientes, sin embargo, la mayoría de los problemas encontrados en la industria son de naturaleza multiobjetivo. El PRV y sus variantes son muy importantes a causa de la significativa contribución del costo de distribución al costo total [Gol84, Goe89, Hof10].

El problema de rutas de vehículos es uno de los problemas de optimización combinatoria más ampliamente estudiados. En la práctica, las rutas de vehículos pueden ser la historia de mayor éxito en investigación de operaciones. Por ejemplo, cada día 105,267 choferes de UPS[1] (United Parcel Service) siguen rutas generadas por computadora. Los choferes visitan 9.8 millones de clientes (1.6 millones de recolecciones (16 %), 8.2 millones de entregas (84 %)) y manejan un promedio de 18 millones de paquetes [Gol08].

El problema de rutas de vehículos con recolecciones (PRVR), es una extensión del PRV, en la cual el conjunto de clientes está particionado en dos subconjuntos: clientes de entrega y clientes de recolección. Los clientes de entrega requieren bienes del almacén, mientras que los clientes de recolección tienen bienes que requieren ser recogidos y transportados al almacén. Esta partición de clientes es frecuente en situaciones prácticas, por ejemplo, en la industria de los abarrotes, los supermercados y las tiendas son los clientes de entrega, y los proveedores de abarrotes son los clientes de recolección [Tot97]. Se ha reconocido ampliamente que con el PRVR se puede lograr

[1] Información de UPS correspondiente a 2014 extraída el 3 de abril de 2015 de la página: pressroom.ups.com/Fact+Sheet/UPS+Fact+Sheet?mkname=companyinfo
Con dicha información se estimó el número de choferes. Se consideró que el número de empleados de UPS que fueron choferes en 2006 es proporcional al de 2014.

un ahorro significativo en términos de costos de transporte al aprovechar la capacidad sin usar, del vehículo vacío, que regresa al almacén [Gol85].

El PCTFR fue propuesto por Salhi et al. [Sal13]. Combina aspectos de dos variantes del PRV: el PRVR y el PCTF. En el PCTFR los clientes de entrega son atendidos antes que los clientes de recolección, no se permite que una ruta sólo contenga clientes de recolección, se tiene una flota heterogénea con diferentes costos fijos de acuerdo a sus tamaños y se consideran restricciones de capacidad en los vehículos. El problema consiste en determinar la mejor composición de la flota y rutas de costo mínimo que satisfagan la demanda, u oferta, de los clientes y las restricciones del problema.

El PCTF y el PCTFR son problemas de optimización importantes, pues la mayoría de las empresas que tienen que entregar o recoger bienes poseen una flota heterogénea de vehículos [Tai99, Hof10].

Puede mostrarse que el PCTFR es NP-duro: si se considera una flota compuesta por un único tipo de vehículo y no se realizan recolecciones, el problema se reduce al PRV, el cual se sabe es NP-duro [Len81]. Esto significa que no se conoce un algoritmo que encuentre una solución óptima en tiempo polinomial para todos los casos del problema. Esto es, desde el punto de vista práctico, es imposible resolver casos de prueba "grandes" del problema. Para resolver este problema se han empleado métodos heurísticos y metaheurísticas. La metaheurística de la BD se ha empleado para resolver varias variantes del PRV [Och98, Cor02, Rus06, Bel09, Bel13, Mel14, Zha12, Pra13, Tan10, Sil13], por lo que se considera que la BD es apropiada para resolver el PCTFR. La BD emplea conjuntos de soluciones que representan concesiones mutuas entre objetivos múltiples y requiere diversidad entre soluciones para su buen desempeño. Por lo tanto, utilizar la BD para solucionar el problema facilita considerar más de un objetivo al mismo tiempo [Coe07].

El PCTFR es un importante problema de optimización que permite al administrador encargado de la distribución determinar, tanto la composición y tamaño de la flota, como el transporte eficiente, lo cual tiene como resultado una reducción de los costos de distribución y permite a la compañía competir en el mercado [Goe89, Sal13]. El PCTF y el PRVR se han estudiado ampliamente por separado [Sal13], no combinados.

El PCTFR un problema real, tangible y existen muchas empresas interesadas en nuevas aproximaciones [Hof10, Par12, Par14]. Por ejemplo, debido al incremento en el número de cajeros automáticos, la industria bancaria pidió a un grupo de investigadores encontrar una solución biobjetivo (minimizar: el costo total de las rutas y la duración del viaje) al problema de reposición del dinero en los cajeros automáticos [Anb12].

En la industria una flota de vehículos raramente es homogénea. Frecuentemente, la flota de vehículos se adquiere durante un largo periodo de tiempo, y los vehículos tendrán diferentes características debido al desarrollo tecnológico. Asimismo, las características de la demanda de transporte en volumen, tiempo y geografía pueden motivar el uso de vehículos con diferentes tamaños. Una flota heterogénea de vehículos generalmente es más flexible y rentable para las variaciones de la demanda. Además, puede haber ubicaciones que limiten su acceso a vehículos con algunas características particulares [Hof10]. Por lo que se determinó desarrollar un algoritmo para resolver el PCTFR. El PRVR puede considerarse un caso particular del PCTFR en el que la flota de vehículos es fija y tiene un único tipo de vehículo.

Esta tesis está organizada de la siguiente manera. El Capítulo 2 contiene las definiciones del PRVR, el PCTF y el PCTFR, los casos de prueba propuestos para cada una de estas variantes, y la revisión de la literatura respectiva. La última sección de este capítulo presenta la revisión de la literatura correspondiente a los problemas multiobjetivo de rutas de vehículos. El Capítulo 3 contiene los antecedentes necesarios de la optimización multiobjetivo, así como tres métricas de desempeño multiobjetivo. El Capítulo 4 contiene la metodología propuesta para el algoritmo de BD para el PCTFR y los resultados experimentales del mismo. El Capítulo 5 contiene la metodología del algoritmo multiobjetivo propuesto y los resultados experimentales correspondientes. Las conclusiones de esta tesis se presentan en el Capítulo 6.

Hipótesis

Es posible proponer un algoritmo, capaz de resolver el PCTFR multiobjetivo y obtener las mejores soluciones conocidas (o lo más cercano posible) en un tiempo razonable y abarcar las regiones enteras de los frentes de Pareto.

Objetivo principal

Desarrollar e implementar un algoritmo de BD que resuelva el PCTFR multiobjetivo de manera eficiente en cuanto a tiempo y calidad de la solución y obtener frentes de Pareto amplios para dichas soluciones.

Objetivos específicos

1. Desarrollar e implementar un algoritmo de BD multiobjetivo, que abarque las regiones enteras de los frentes de Pareto, para resolver el PCTFR.

2. Estudiar métricas de desempeño para comparar conjuntos de soluciones.

3. Mejorar la calidad de las soluciones y/o el tiempo de ejecución.

Aportación

La aportación principal de esta investigación es un algoritmo de BD que resuelva el PCTFR multiobjetivo (minimizar: el número de rutas y el costo total de la solución) de manera eficiente en cuanto a tiempo y calidad de la solución y obtener la región entera de los frentes de Pareto.

Se sometió el resumen titulado "A Scatter Search Algorithm for the Heterogeneous Vehicle Routing Problem with Backhauls" al Congreso Latino-Iberoamericano de Investigación Operativa 2016, el cual fue aceptado para su presentación oral.

Durante el desarrollo de esta investigación se elaboró el artículo "A Scatter Search Algorithm for the Fleet Size and Mix Vehicle Routing Problem with Backhauls", el cual se envió a la revista IEEE Latin America Transactions (factor de impacto 0.631, Clarivate Analytics, 2017 Journal Citation Reports), paso una primera revisión, fue aceptado con cambios que ya se atendieron, fue reenviado y se encuentra actualmente en revisión.

Capítulo 2

PRV

El problema de rutas de vehículos fue propuesto por primera vez por Dantzig y Ramser en 1959. Consiste en encontrar una ruta óptima para una flota de camiones de reparto de gasolina entre una terminal de carga y un gran número de estaciones de servicio abastecidas por ésta [Dan59]. Se desea asignar las estaciones a los camiones de manera que las demandas de las estaciones sean satisfechas y la distancia total viajada por la flota sea mínima. Con un procedimiento basado en una formulación de programación lineal los autores obtienen una solución cercana a la óptima con cuatro rutas para un problema con doce estaciones de servicio [Gol08, ver Prefacio].

Está disponible en Internet un sitio[1] dedicado al estudio del PRV, en él se pueden encontrar, por ejemplo: casos de prueba, bibliografía, variantes del problema y reportes técnicos.

Las tres primeras secciones de este capítulo se refieren al PRVR, al PCTF y al PCTFR, respectivamente. La cuarta sección a los problemas multiobjetivo de rutas de vehículos.

2.1. PRVR

El PRVR se ha resuelto, principalmente, considerando una flota homogénea de vehículos (Toth y Vigo [Tot01]), y en menor manera considerando una flota heterogénea (Salhi, Wassan y Hajarat [Sal13]).

En relación con el PRVR, hay un sitio[2] en el que están disponibles, entre

[1] El nombre del sitio es *The VRP Web* y su dirección es: http://www.bernabe.dorronsoro.es/vrp/. Fecha de la última consulta 13/abr/2015.

[2] Denominado *Linehaul - Backhaul,* cuya dirección es: http://www2.isye.gatech.edu/~mgoetsch/lineback.html. Fecha de la última consulta

otras cosas, las publicaciones de Goetschalckx y Jacobs-Blecha, los casos de prueba propuestos y la solución a los mismos obtenida por los autores.

El PRVR puede formalmente definirse como sigue [Tot99, Gar12].

Sea $G = (V, A)$ una gráfica completa dirigida, con $V = \{0\} \cup L \cup B$ un conjunto de vértices, donde el vértice 0 corresponde al almacén y $L = \{1, \ldots, n\}$ y $B = \{n+1, \ldots, n+m\}$ son los subconjuntos de vértices correspondientes a los clientes de entrega y recolección, respectivamente, y $A = \{(i,\ j) : i,\ j \in V, i \neq j\}$ es un conjunto de arcos.

Hay K vehículos idénticos de capacidad Q estacionados en el almacén, para atender a los clientes. Cada vehículo comienza y termina su ruta en el almacén.

El problema consiste en encontrar K rutas de costo mínimo, tales que:

(i) cada vehículo atiende una sola ruta,

(ii) cada cliente es visitado exactamente una vez por únicamente un vehículo.

(iii) los clientes de recolección, si los hay, son atendidos después de los clientes de entrega,

(iv) no se permite que una ruta sólo contenga clientes de recolección,

(v) la carga total asociada tanto a los clientes de entrega, como a los clientes de recolección no debe exceder la capacidad del vehículo que la atiende.

El costo c_{ij} asociado con el arco $(i,\ j) \in A$ se define como la distancia euclidiana entre los vértices i y j, con $c_{ij} = c_{ji}$ para cada $i,\ j \in V$ tal que $i \neq j$, y $c_{ii} = +\infty$ para cada $i \in V$.

El objetivo es minimizar el costo total de las rutas definido como la suma de los costos de los arcos que pertenecen a las rutas.

2.1.1. Casos de prueba

La primera clase está compuesta por 68 casos de prueba de dominio público[3] en los que el número total de vértices está entre 25 y 200. Fueron propuestos por Goetschalckx y Jacobs-Blecha para el PRVR [Goe89, Jac92]. Todos estos casos de prueba fueron generados aleatoriamente, los puntos de entrega y recolección se distribuyen uniformemente con x en el intervalo $(0, 24000)$ y y en el intervalo $(0, 32000)$. Para todos los casos de prueba el almacén se colocó en posición central en $(12000, 16000)$. La demanda y oferta se generaron de una distribución normal con media de 500 unidades y desviación estándar de 200 unidades. Se generaron 20, 30, 45, 75 y 100

13/abr/2015.

[3] Disponible en http://www2.isye.gatech.edu/∼mgoetsch/lineback.html

clientes de entrega. El número de clientes de recolección corresponde al 25, 50 y 100 % del número de clientes de entrega. Para cada caso, los valores de K y Q fueron propuestos por los autores.

La segunda clase consta de 33 casos de prueba, que se generaron a partir de 11 casos de prueba de dominio público[4] del problema de rutas de vehículos con capacidad limitada (PRVC), en los que el número de vértices está entre 21 y 100. De cada caso de prueba del PRVC, Toth y Vigo generaron tres casos de prueba para el PRVR, cada uno correspondiente a un porcentaje de clientes de entrega de 50 %, 66 % y 80 % [Tot97]. El conjunto de vértices se dividió definiendo como cliente de recolección al primero de cada dos, tres o cinco vértices, respectivamente. En estos casos el número de vehículos disponibles es $K = \text{máx}\{K_L,\ K_B\}$, donde K_L y K_B denotan el número mínimo de vehículos necesarios para servir a todos los clientes de entrega y recolección, respectivamente. Para los casos donde $K_L < K_B$, los conjuntos de clientes de entrega y recolección se intercambian.

2.1.2. Revisión de la literatura

El estudio de Laporte [Lap09] da un panorama general del estado del arte del problema de rutas de vehículos, presenta algoritmos exactos, heurísticos y metaheurísticas que se han empleado para resolver diversas variantes de este problema.

En el capítulo 8, *VRP with backhauls* [Tot01], con claridad describen: los algoritmos exacto y heurístico propuestos por Toth y Vigo en los artículos [Tot97] y [Tot99], respectivamente, el algoritmo exacto desarrollado por Mingozzi, Giorgi y Baldacci [Min99]. Asimismo resumen el algoritmo heurístico de Deif y Bodin [Dei84] y los algoritmos heurísticos de Goetschalckx y Jacobs-Blecha [Goe89, Jac92].

La Sección 9.2 del capítulo *Four Variants of the Vehicle Routing Problem* [Tot14] presenta un estudio de los artículos publicados entre 2002 y principios de 2014 relativos al PRVR.

Metaheurísticas

Algoritmo de Osman y Wassan Emplearon Búsqueda Tabú (BT) para resolver el PRVR. Para construir una solución inicial emplearon dos procedimientos heurísticos:

[4] Vea sitio *TSPLIB, Ruprecht-Karls-Universitat Heidelberg* cuya dirección es: http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/. Fecha de la última consulta 14/abr/2015.

La heurística de inserción de ahorro (saving-insertion heuristic, SIH) encuentra un conjunto de rutas de vehículos para un conjunto de clientes de entrega mediante una función de ahorro modificada. Después, los clientes de recolección son insertados en las rutas generadas de acuerdo al criterio de inserción al mejor costo siempre que las restricciones de clientes y vehículos no sean violadas. La solución inicial obtenida es mejorada aplicando por separado a los clientes de entrega, de los clientes de recolección las heurísticas 2-opt y 3-opt [Lin65].

La heurística de asignación de ahorro (saving-assignment heuristic, SAH) comienza generando, con el procedimiento heurístico antes descrito, dos conjuntos de rutas de vehículos para los clientes de entrega y recolección respectivamente, después aplica a cada ruta las heurísticas 2-opt y 3-opt. Luego, para encontrar una solución del PRVR se resuelve un problema de asignación a costo mínimo de los dos conjuntos de rutas de entrega y recolección, usando la rutina descrita en [Car88].

Posteriormente, la metaheurística de BT se aplica sobre la solución inicial generada [Osm02].

Resultados computacionales. Implementaron y ejecutaron su algoritmo en una Sun SPARC 1000 a 50 MHz (10 Mflop/s según [Don14]).

Consideraron dos modos de calcular la distancia entre clientes. Primero, determinar la matriz de distancias con los valores reales de las distancias euclidianas entre clientes y dar la solución final redondeada al entero más cercano. Segundo, determinar la matriz de distancias con las distancias euclidianas entre clientes multiplicadas por 10 y redondeadas al entero más cercano. La solución final se divide entre 10 y se redondea al entero más cercano.

Resolvieron los primeros 62 casos de la primera clase (25 a 150 clientes) usando la matriz de distancias con valores reales, redondearon a dos decimales sus resultados finales. Si se comparan los resultados obtenidos por Osman y Wassan (OW2002) con los de Toth y Vigo [Tot97] (TV97), Mingozzi, Giorgi y Baldacci [Min99] (MGB), y Toth y Vigo [Tot99] (TV99) mejoraron la solución de 27 de 62 casos, en promedio sus soluciones son 1.01 % mejores. El tiempo promedio de ejecución de los casos de esta clase fue de 2859.93 y 4025.69 segundos para los algoritmos SIH y SAH, respectivamente.

También resolvieron los primeros 62 casos de la primera clase usando la matriz de distancias con valores enteros, compararon sus resultados con los de TV99, a pesar de que TV99 emplearon la matriz de distancias con valores reales. Si se comparan los resultados obtenidos por Osman y Wassan con los de MGB, TV97 y TV99 mejoraron la solución de 23 casos y llegaron a la mejor solución conocida en 20 casos. En promedio, sus soluciones son

1.01 % mejores que las soluciones conocidas a la fecha en que publicaron su artículo.

Si se comparan los resultados de los primeros 62 casos de la primera clase de Osman y Wassan con la matriz de distancias con valores reales y con valores enteros, en promedio, los resultados con la matriz con valores enteros son 0.00035 % mejores que con la matriz con valores reales.

Osman y Wassan también resolvieron los 33 casos de la segunda clase (21 a 100 clientes). Comparando sus resultados con los de TV97, MGB y TV99 mejoraron la solución en 4 casos, pero no llegaron a la mejor solución conocida en 12 casos, globalmente no mejoraron las soluciones conocidas. El tiempo promedio de ejecución de los casos de esta clase fue de 1513.81 y 1591.78 segundos para los algoritmos SIH y SAH, respectivamente.

Generalmente es difícil comparar el tiempo de ejecución de los diferentes algoritmos, ya que éste se determina en computadoras distintas. Se pueden comparar los tiempos de ejecución por medio de los valores en Mflop/s (millones de operaciones en punto flotante por segundo) de computadoras diferentes dados en la referencia Linpack [Don14]. La Tabla 2.1 presenta especificaciones de computadoras y sus valores en Mflops, empleadas para resolver el PRVR con algoritmos heurísticos. Con los valores en Mflop/s se determina un factor del tiempo de ejecución de un algoritmo en relación a la computadora Sun SPARC 1000 a 50 MHz. El factor se emplea para obtener una estimación del tiempo de ejecución en dicha computadora, si ésta fuese usada para correr un algoritmo en particular [Gaj09]. En la Tabla 2.2 se muestra la comparación de los algoritmos heurísticos de TV99 y OW2002.

Tabla 2.1
Computadoras empleadas para probar algoritmos heurísticos

Autores	Computadora	Mflop/s	Factor
TV99	IBM 486/33	0.94	.094
OW2002	Sun SPARC 1000 a 50 MHz	10	1

Algoritmo de Ropke y Pisinger Estos investigadores desarrollaron un modelo unificado al que denominaron Problema de Recolección y Entrega con Ventanas de Tiempo, con el cual resuelven las siguientes seis variantes del problema: PRVR, Problema de Rutas de Vehículos con Recolecciones Mixto[5] (PRVRM), Problema de Rutas de Vehículos con Recolecciones Mix-

[5] En éste se pueden mezclar libremente los clientes de entrega y recolección dentro de una ruta. También hay libertad para emplear tantos vehículos como se desee.

to con Múltiples Depósitos[6], Problema de Rutas de Vehículos con Recolecciones y Ventanas de Tiempo[7], Problema de Rutas de Vehículos Mixto con Recolecciones y Ventanas de Tiempo[8], Problema de Rutas de Vehículos con Entrega y Recolección Simultánea[9] (PRVERS).

Tabla 2.2
Comparación de dos heurísticas para el PRVR

	TV99	OW2002
62 primeros casos de prueba de la primera clase		
Mejor solución promedio	295,045.92	291,261.78
Tiempo promedio de ejecución**	4.56	4,025.69
33 casos de prueba de la segunda clase		
Mejor solución promedio	714.12	708.42
Tiempo promedio de ejecución**	10.77	1591.78

**Aproximado (en segundos), respecto a la computadora Sun SPARC 1000 a 50 MHz.

Cada problema se resuelve transformándolo en un problema del modelo unificado.

Este algoritmo contiene seis heurísticas de eliminación y cinco de inserción [Rop06], las cuales son aplicadas con una frecuencia variable controlada por un estrato de aprendizaje. Algunas heurísticas se usan para intensificar la búsqueda, otras para diversificarla. El estrato de aprendizaje no sólo distribuye el tiempo del CPU entre las heurísticas involucradas, sino que también controla la intensificación y diversificación de la búsqueda basado en información empírica.

[6] Este problema es una generalización del anterior. En él hay varios depósitos presentes. En cada depósito hay un número limitado de vehículos disponibles y cada vehículo debe comenzar y terminar su servicio en el mismo depósito.

[7] El PRVR se extiende mediante la asignación de una ventana de tiempo a cada cliente. Las visitas a un cliente deben comenzar dentro de la ventana de tiempo. Si el vehículo llega demasiado temprano con un cliente, tiene que esperar hasta el comienzo de la ventana de tiempo. Si el vehículo llega demasiado tarde, la ruta no es válida. En este problema pueden haber rutas que sólo contengan clientes de recolección, además, el número de vehículos a emplear no es fijo.

[8] Este problema se deriva del anterior, en él se pueden mezclar los clientes de recolección y entrega.

[9] En este problema un subconjunto de clientes simultáneamente demanda bienes y provee bienes al depósito, por tanto, ambas, la entrega y la recolección, deben ocurrir en esos clientes, considerando que la entrega se realiza antes de la recolección.

El estrato de aprendizaje observa con qué frecuencia una heurística de eliminación o inserción dada contribuye con una nueva solución aceptada, e incrementa la probabilidad de elegir la heurística dada de acuerdo a su éxito. Para asegurar la recolección de información estadística de todas las heurísticas a través de la búsqueda, cada heurística se emplea al menos un límite inferior dado.

Resultados computacionales. Implementaron y ejecutaron su algoritmo en una computadora personal con un procesador de 1.5 GHz Pentium IV (326 Mflop/s respecto a la referencia LINPACK y 1311 Mflop/s respecto al TPP[10], ver [Don14]).

Probaron tres configuraciones de su algoritmo heurístico:

- La configuración estándar emplea el estrato de aprendizaje y tres heurísticas de eliminación.

- La configuración 6R-no aprendizaje desactiva el mecanismo de aprendizaje y usa las seis heurísticas de eliminación propuestas.

- La configuración 6R-aprendizaje normal utiliza el estrato de aprendizaje y las seis heurísticas de eliminación.

Probaron su algoritmo con los 62 primeros casos de la primera clase (25 a 150 clientes). Calcularon la distancia euclidiana entre clientes con números reales y dieron la solución final con dos decimales. La configuración con la que obtuvieron mejores resultados fue la 6R-no aprendizaje, después la 6R-aprendizaje normal y por último la estándar. Mejoraron el costo conocido a la fecha de su estudio de 19 casos de prueba. En promedio sus soluciones son 0.04 % mejores. El tiempo promedio de ejecución de los casos de esta clase fue de 69, 71 y 73 segundos para los algoritmos 6R-aprendizaje normal, 6R-no aprendizaje y estándar, respectivamente.

De acuerdo a la referencia [Don14] el TPP de la computadora que usaron Ropke y Pisinger es de 1311 MFlop/s, mientras que el de la computadora empleada por Osman y Wassan es de 25 MFlop/s, por lo que estiman que su computadora es ($1311/25 \approx$) 53 veces más rápida.

Con base en los tiempos promedio de ejecución que se presentan en la Tabla 2.3, Ropke y Pisinger consideran que el tiempo de ejecución de su algoritmo y del de Osman y Wassan es comparable.

Probaron su algoritmo con los 33 casos de prueba de la segunda clase (21 a 100 clientes), y mejoraron la solución conocida de 5 casos. En promedio sus soluciones son 0.18 % mejores. El tiempo promedio de ejecución de los casos de esta clase fue de 41 segundos para el algoritmo 6R-aprendizaje normal y 42 segundos para los algoritmos 6R-no aprendizaje y estándar.

[10] TPP = Toward Peak Performance = Relativo al desempeño máximo.

Tabla 2.3
Tiempos promedio de ejecución

	A [Osman, 2002]	A/53	[Ropke, 2006]
Algoritmo rápido	SIH		6R-aprendizaje normal
Tiempo promedio de ejecución	2859.93	54	69
Algoritmo lento	SAH		Estándar
Tiempo promedio de ejecución	4025.69	76	73

Ropke y Pisinger también proporcionaron resultados computacionales de las otras 5 variantes del problema que resuelve su modelo, pero no se analizarán aquí por no ser el tema central de esta tesis.

Ropke y Pisinger consideran que su heurística unificada, para una amplia clase de problemas de rutas de vehículos con recolecciones, es capaz de proveer soluciones comparables a las obtenidas por heurísticas especializadas y piensan que los practicantes pueden enfocarse en esta sola heurística cada vez que un nuevo tipo de problema necesite ser resuelto. Estiman además que la combinación de varias vecindades hace más fácil a la heurística de búsqueda local explorar el espacio de soluciones, lo que permite encontrar soluciones de alta calidad. Asimismo consideran el monitoreo del estrato de aprendizaje que controla la elección de la vecindad como un estrato que mantiene un balance apropiado entre intensificación y diversificación.

Algoritmo de Brandão Este investigador resolvió el PRVR por medio de un algoritmo de BT. Para generar una solución inicial desarrolló dos métodos: uno basado en el PRV abierto y el otro basado en un K-árbol. Después de generar una solución inicial se aplica a ésta el algoritmo de BT.

Desarrolló tres versiones del Algoritmo de BT (ABT): ABT-abierto, ABT-K-árbol y ABT-K-árbol_r.

Resultados computacionales. Brandão implementó y ejecutó su algoritmo en una computadora HP Vectra VEi8 Pentium III a 500 MHz (72.5 Mflop/s según [Bra06, Gaj09]). De las tres versiones que desarrolló de su algoritmo, la versión ABT-K-árbol_r parece ser la que obtuvo mejores resultados, por lo que se reportan los tiempos de ejecución de esta.

Probó su algoritmo con los 62 primeros casos de la primera clase (25 a 150 clientes). Calculó a doble precisión la distancia euclidiana entre clientes, la multiplicó por 10 y después la redondeó al entero más cercano. El valor de la solución final se divide entre 10 y se redondea al entero más cercano. Mejoró el costo conocido de 6 casos. En promedio sus soluciones son similares

a las mejores soluciones conocidas a la fecha de publicación de su artículo. El tiempo promedio de ejecución del algoritmo ABT-K-árbol_r en los casos de esta clase es de 815 segundos.

Brandão examinó su algoritmo con los 33 casos de la segunda clase (21 a 100 clientes). Mejoró el costo conocido de un caso (eilB101_66), sin embargo, en promedio sus soluciones son 0.02 % peores por los casos en los que no llegó a la mejor solución conocida. El tiempo promedio de ejecución del algoritmo ABT-K-árbol_r en los casos de esta clase es de 297 segundos.

La Tabla 2.4 presenta especificaciones de las computadoras y sus valores en Mflops (de la referencia Linpack [Don14]), empleadas por Brandão (B2006), Toth y Vigo (TV99), Osman y Wassan (OW2002) y Ropke y Pisinger (RP2006) para resolver el PRVR. Para comparar los tiempos de ejecución se eligieron las implementaciones que produjeron mejores resultados, de Brandão: ABT-K-árbol_r, de Osman y Wassan: SAH y de Ropke y Pisinger: 6R-no aprendizaje. La Tabla 2.5 muestra la comparación de los algoritmos antes mencionados.

Tabla 2.4
Computadoras empleadas por diferentes heurísticas

Autores	Computadora	Mflop/s	Factor
TV99	IBM 486/33	0.94	0.0130
OW2002	Sun SPARC 1000 a 50 MHz	10	0.1379
RP2006	Pentium IV a 1.5 GHz	326	4.4966
B2006	HP Vectra Pentium III 500 MHz	72.5	1

Tabla 2.5
Comparación de diferentes heurísticas para el PRVR

	TV99	OW2002	RP2006	B2006
62 primeros casos de prueba de la primera clase				
Mejor solución promedio	295,045.92	291,261.78	290,896.73	290,764.60
Tiempo promedio de ejecución**	0.63	555.27	319.47	815.00
33 casos de prueba de la segunda clase				
Mejor solución promedio	714.12	708.42	701.00	701.09
Tiempo promedio de ejecución**	1.49	219.56	189.95	297.00

**Aproximado (en segundos), respecto a la computadora HP Vectra Pentium III 500 MHz.

Algoritmo de Gajpal y Abad Su enfoque de múltiples colonias de hormigas está inspirado en el procedimiento primero grupo, segundo ruta

del PRV (Fisher y Jaikumar [Fis81]). Usan dos tipos de hormigas para construir una solución: hormigas vehículo y hormigas ruta. El primer tipo de hormigas se usa para asignar los clientes a los vehículos, el segundo para construir una ruta con los clientes asignados a un vehículo. Definen una intensidad del rastro para cada tipo de hormigas (que se usa para construir la solución), la cual se actualiza durante el proceso de búsqueda. Después de generar una solución, para mejorarla le aplican tres heurísticas: 2-opt, inserción y/o intercambio de clientes entre rutas e intercambio de subrutas entre rutas.

Resultados computacionales. Implementaron su algoritmo en una computadora Intel Xeon con 2.4 GHz (884 Mflop/s de acuerdo a [Don14, Gaj09]).

Examinaron su algoritmo con los 62 primeros casos de la primera clase, calcularon la distancia euclidiana entre clientes sin redondear, al final redondean la longitud total del recorrido a dos decimales. En promedio las soluciones de Gajpal y Abad son 0.03 % mejores que las obtenidas con los algoritmos antes descritos [Osm02, Rop06, Bra06]. Encontraron cuatro soluciones mejores que las conocidas, las cuales presentaron en su artículo. El tiempo promedio de ejecución de un caso es de 67.57 segundos (25 a 150 clientes).

Asimismo, probaron su algoritmo con los 33 casos de la segunda clase. En promedio las soluciones de Gajpal y Abad son 0.003 % mejores que las obtenidas con los algoritmos antes descritos [Osm02, Rop06, Bra06]. También encontraron una solución mejor para uno de éstos casos y la incluyeron en su artículo. El tiempo promedio de ejecución de un caso es de 25.64 segundos (21 a 100 clientes).

2.2. PCTF

Este problema formulado en 1984 por Golden et al., puede definirse formalmente como sigue [Bal08, Gol84].

Sea $G = (V, A)$ una gráfica completa dirigida, con $V = \{0\} \cup L$ un conjunto de vértices, donde el vértice 0 corresponde al almacén, $L = \{1, \ldots, n\}$ es el conjunto de vértices correspondiente a los n clientes, y $A = \{(i, j) : i, j \in V, i \neq j\}$ es un conjunto de arcos.

La flota está compuesta por w tipos de vehículos diferentes estacionados en el almacén, con $W = \{v_1, \ldots, v_w\}$. Cada vehículo comienza y termina su ruta en el almacén. Se considera que hay un número ilimitado de cada tipo de vehículo. Cada tipo de vehículo $u \in W$ tiene una capacidad igual a Q_u y

un costo fijo f_u.

Cada vértice $j \in V$ está localizado geográficamente en (x_j, y_j), y cada cliente $j \in L$ requiere el suministro de z_j bienes. Se supone $z_0 = 0$.

El costo c_{ij} asociado con el arco $(i,\ j) \in A$ se define como la distancia euclidiana entre los vértices i y j (se asume simetría y que el costo es independiente del tipo de vehículo empleado). El costo de una ruta corresponde a la suma de los costos de los arcos que forman la ruta, más el costo fijo del vehículo.

El PCTF consiste en diseñar un conjunto de rutas de costo mínimo, tales que:

(i) cada vehículo atiende una sola ruta,

(ii) cada cliente es visitado exactamente una vez por únicamente un vehículo.

(iii) la demanda total de los clientes visitados en la ruta no debe exceder la capacidad del vehículo que la atiende.

2.2.1. Casos de prueba

En la tercera clase se encuentran 12 casos de prueba propuestos por Golden et al. para el PCTF [Gol84]. Solamente se consideraron en esta clase los 12 casos en los que se proporcionan las coordenadas de los clientes (casos de prueba 3 a 6 y 13 a 20). Se hará referencia a dichos casos con la misma numeración que los autores. En estos casos de prueba el número total de vértices es 20, 50, 75 y 100 y el número de diferentes tipos de vehículos varía de 3 a 6. Cada tipo de vehículo tiene asociada una capacidad y un costo fijo de adquisición o arrendamiento. En estos casos se considera que se dispone de un número ilimitado de vehículos de cada tipo.

2.2.2. Revisión de la literatura

El capítulo *Routing a Heterogeneous Fleet of Vehicles* [Bal08], presenta un panorama de los enfoques empleados para resolver este problema, revisa las cotas inferiores y los algoritmos heurísticos propuestos. A la fecha de publicación de este capítulo no se conocía el algoritmo exacto desarrollado por Pessoa et al. [Pes07] para la PCTF.

El artículo de Hoff et al. analiza los aspectos industriales, en el transporte marítimo y terrestre, de combinar composición de la flota y rutas de vehículos [Hof10]. Da argumentos que muestran que generalmente la industria emplea una flota heterogénea de vehículos. Hace notar que el transporte eficiente es cada vez más importante para la sociedad (debido: al

cambio climático, a otras preocupaciones ambientales y a que el costo de distribución representa una parte importante del costo de un producto). El crecimiento económico, el incremento en el consumo y la globalización aumentan las necesidades de transporte. En todas las modalidades de transporte, los propietarios de bienes y los proveedores de transportes realizan racionalizaciones estructurales a través de fusiones y adquisiciones. La fuerte competencia tanto entre proveedores de transporte como entre propietarios de bienes conduce a una mayor demanda de eficiencia, servicio al cliente y reducción de costos en el transporte. La industria enfrenta retos de composición de la flota en todos los niveles de decisión. Para los proveedores de transporte y los propietarios de bienes, un objetivo es lograr un balance óptimo entre la flota de vehículos propia y el transporte a subcontratar, así como decidir la composición correcta de la flota en vista de las necesidades de transporte. Asimismo, critican que no se tome en cuenta en el PRV los aspectos estocástico y de riesgo. Y que no se generen soluciones robustas y resilientes. Consideran que no hay buenos casos de prueba, por lo que proporcionaron una lista de los atributos que consideran debe tener un caso de prueba común.

La Sección 9.3 del capítulo *Four Variants of the Vehicle Routing Problem* [Tot14] proporciona un estudio de los artículos del problema de rutas de vehículos heterogéneos (PRVH) publicados entre 2008 y principios de 2014.

Los problemas de rutas de vehículos con una flota heterogénea tienen muchas variantes, de las cuales se reúnen aquí las examinadas en esta sección.

1. FICV, flota ilimitada de vehículos con costo variable por tipo de vehículo.

2. FICFV, flota ilimitada de vehículos con costos fijos y variables por tipo de vehículo.

3. FFCV, flota fija de vehículos con costo variable por tipo de vehículo.

4. FFCFV, flota fija de vehículos con costos fijos y variables por tipo de vehículo.

5. PRVDU, problema de rutas de vehículos que dependen de la ubicación[11].

[11] Algunos clientes solo pueden ser visitados por un subconjunto de los vehículos existentes. Usualmente esto ocurre cuando la ubicación de un cliente limita su acceso a vehículos con algunas características particulares.

No se examinaron los resultados de las variantes antes mencionadas, aunque se menciona qué investigadores las resolvieron. Sólo se analiza los resultados obtenidos en los casos de las clases tercera y cuarta.

Algoritmos exactos

Algoritmo de Pessoa, Aragão y Uchoa Presentaron un algoritmo de ramificación, corte y precio para resolver los casos de prueba de la tercera clase (PCTF) [Pes07].

Resultados computacionales. Ejecutaron su algoritmo en una computadora con un procesador Core 2 Duo a 2.13 GHz con 2 GB de memoria RAM. Resolvieron los programas lineales y el modelo conjunto partición con CPLEX 10.0. Examinaron su algoritmo con los 12 casos de prueba de la tercera clase. Probaron que era óptima la solución de nueve de ellos. Obtuvieron la solución óptima de un caso de prueba con 75 clientes. El tiempo promedio de ejecución de los 9 casos en los que se probó que la solución obtenida es óptima es de 9502 segundos.

Algoritmo de Baldacci y Mingozzi Desarrollaron un método exacto unificado basado en una formulación de conjunto partición que resuelve siete variantes del PRV: dos con una flota de vehículos homogéneos y cinco con una flota de vehículos heterogéneos. Las variantes con flota heterogénea son: PCTF, FICFV, FICV, FFCV y PRVDU.

Resultados computacionales. Ejecutaron su algoritmo en una computadora con un procesador AMD Athlon 64 X2 Dual Core 4200+ a 2.6 GHz con 3 GB de memoria RAM. Usaron CPLEX 10.1. Obtuvieron la solución óptima de casos de prueba con hasta 100 clientes [Bal09]. Probaron que era óptima la solución de 11 de los 12 casos de prueba de la tercera clase. El tiempo promedio de ejecución de los 11 casos de la tercera clase en los que probaron que la solución obtenida es óptima es de 9037 segundos.

Algoritmos heurísticos

Algoritmo de Choi y Tcha Presentaron un modelo conjunto cubierta, resolvieron la relajación de programación lineal de éste por medio de generación de columna. Modificaron un par de esquemas de programación dinámica (desarrollados para el PRV) para generar columnas factibles eficazmente. Basados en las cotas inferiores obtenidas, usaron un algoritmo de ramificación y acotamiento para encontrar una solución [Cho07]. Probaron

su algoritmo en los 12 casos de la tercera clase, en 12 casos con FICV y en 12 casos con FICFV.

Resultados computacionales. Ejecutaron su algoritmo en una computadora personal con Pentium IV 2.6 GHz con 523 MB de memoria RAM, codificaron su algoritmo en MS Visual C++ 6.0. Usaron CPLEX 6.6.

A la fecha de publicación de su artículo, en los casos de la tercera clase: mejoraron cinco de las soluciones conocidas y en promedio sus soluciones son 0.05 % mejores que las conocidas. El tiempo promedio de ejecución de un caso es de 109.75 segundos.

Metaheurísticas

Algoritmo de Taillard Presentó un método heurístico de generación de columna. Usando BT resolvió, para cada tipo de vehículo, un problema de rutas de vehículos homogéneos. Después, las rutas del problema de rutas de vehículos homogéneos se combinan para producir una solución del problema de rutas de vehículos heterogéneos [Tai99].

Planteó el problema de programación lineal binaria para el PCTF (y FICV) y para la variante con FFCV, e hizo notar que aunque los problemas de programación lineal binaria son NP-duros, pueden resolverse óptimamente con un software como CPLEX [Chu95].

Resultados computacionales. Ejecutó su algoritmo en una Sun SPARC a 50 MHz. Resolvió los 12 casos de la tercera clase. En todos los casos, Taillard mejoró el costo conocido a la fecha de la publicación de su artículo. En promedio, sus soluciones son 0.67 % mejores que las conocidas. El tiempo de ejecución de los cuatro primeros casos (3 a 6) de la tercera clase es de 20 a 40 segundos. El tiempo promedio de ejecución de los últimos ocho casos de la tercera clase es de 2648 segundos.

Taillard propuso y dio resultados de 8 casos con FFCV. Asimismo, propuso y dio resultados de 8 casos con FICV.

Algoritmo de Gendreau, Laporte, Musaraganyi y Taillard Emplearon BT y un procedimiento de memoria adaptativa (desarrollado por Rochat y Taillard [Roc95]) para resolver el PCTF y la variante con FICV [Gen99].

Resultados computacionales. Resolvieron los 12 casos de la tercera clase en una Sun Sparc 10. En tres casos mejoraron los resultados obtenidos por Taillard [Tai99], pero obtuvieron peores resultados en los últimos cuatro casos. En promedio sus soluciones son 0.1 % peores que las de Taillard. El tiempo promedio de ejecución de los primeros cuatro casos de la tercera clase

es de 223.5 segundos, mientras que el tiempo promedio de ejecución de los últimos ocho casos de la tercera clase es de 1036.75 segundos.

Resolvieron los 8 casos con FICV propuestos por Taillard.

Incluyeron en su artículo la mejor solución que obtuvieron de cada uno de los 20 casos que resolvieron (vehículo empleado, ruta).

Algoritmo de Wassan y Osman Desarrollaron cinco variantes de BT para resolver el problema de rutas de vehículos con una flota heterogénea [Was02]. Resolvieron con su algoritmo los 20 casos propuestos por Golden et al. [Gol84] (12 de ellos están en la tercera clase) y los 8 casos con FICV propuestos por Taillard.

Resultados computacionales. Codificaron su algoritmo en Fortran 77, y lo ejecutaron en un servidor Sun Sparc 1000 con un procesador a 50 MHz (10 Mflop/s). En los casos de la tercera clase mejoraron una de las soluciones conocidas, en promedio sus soluciones son 0.12 % peores que las soluciones conocidas a la fecha de publicación de su artículo. El tiempo promedio de ejecución de un caso es de 1873.92 segundos. Incluyeron en su artículo la mejor solución que obtuvieron en algunos casos tanto de la tercera clase, como de la variante con FICV.

Algoritmo de Imran, Salhi y Wassan Emplearon Búsqueda en Vecindades Variables (BVV) para resolver el problema de rutas de vehículos con una flota heterogénea e implementaron dos versiones de este [Imr09]. Probaron su algoritmo en los 12 casos de la tercera clase, en los 8 casos con FICV propuestos por Taillard y en 12 casos con FICFV.

Resultados computacionales. Programaron su algoritmo en C++ y lo ejecutaron en una computadora Pentium M con un procesador a 1.7 GHz con 1 GB de memoria RAM. La segunda versión de su algoritmo fue con la que obtuvieron mejores resultados, con ella encontraron la mejor solución conocida en 9 de los 12 casos de la tercera clase, en promedio sus soluciones son 0.06 % peores que las soluciones conocidas a la fecha de publicación de su artículo. El tiempo promedio de ejecución de un caso, de la segunda versión de su algoritmo, es de 500.50 segundos.

Algoritmo de Tütüncü Presentó un algoritmo GRAMPS (Greedy Randomised Adaptive Memory Programming Search) [Tut10]. Propuso una nueva variante, el problema de rutas de vehículos con recolecciones con flota fija de vehículos con costo variable por tipo de vehículo. Implementó su algoritmo en un sistema visual interactivo, el cual permite la intervención de

la persona que toma las decisiones en el proceso de solución del problema. Propuso 18 casos de prueba para este problema.

Resultados computacionales. Se probó el algoritmo GRAMPS en el sistema visual interactivo en una computadora Pentium 4 con un procesador a 2.66 GHz y 504 MB de memoria. En cada caso de prueba se realizan 100 corridas de GRAMPS y un refinamiento de la solución. Probó el algoritmo con los casos FFCV y FICV propuestos por Taillard. Proporcionó resultados computacionales de los 18 casos de prueba propuestos. No resolvió los casos de la tercera clase.

Algoritmo de Subramanian, Penna, Uchoa y Ochi Propusieron un algoritmo híbrido basado en la metaheurística de Búsqueda Local Iterada (BLI), que en la fase de búsqueda local usa vecindades variables descendentes con ordenamiento de vecindades aleatorio, combinado con un modelo de partición [Sub12], el cual emplearon para resolver los casos de la tercera clase y cuatro variantes del problema: FFCV, FICV, FICFV y FFCFV.

Resultados computacionales. Codificaron su algoritmo en C++ y lo ejecutaron en una computadora con un procesador Intel Core i7 a 2.93 GHz con 8 GB de memoria RAM. El modelo partición se resolvió usando CPLEX 12.3. En los 12 casos de la tercera clase obtuvieron la solución conocida de 11 casos, mejoraron la solución conocida de un caso y la incluyeron en su artículo. Sus soluciones son las mejores conocidas a la fecha de publicación de su artículo. El tiempo promedio de ejecución de un caso es de 9.09 segundos.

2.3. PCTFR

Esta es la variante del PRVR con una flota heterogénea, para la cual se propone un algoritmo de solución.

El problema de la composición y tamaño de la flota para el PRVR puede formalmente definirse como sigue [Tot99, Bal08, Gar12].

Sea $G = (V, A)$ una gráfica completa dirigida, con $V = \{0\} \cup L \cup B$ un conjunto de vértices, donde el vértice 0 corresponde al almacén y $L = \{1, \ldots, n\}$ y $B = \{n+1, \ldots, n+m\}$ son los subconjuntos de vértices correspondientes a los clientes de entrega y recolección, respectivamente, y $A = \{(i,\ j) : i,\ j \in V, i \neq j\}$ es un conjunto de arcos.

La flota está compuesta por w tipos de vehículos diferentes estacionados en el almacén, con $W = \{v_1, \ldots, v_w\}$. Cada vehículo comienza y termina su ruta en el almacén. Se considera que hay un número ilimitado de cada tipo

de vehículo. Cada tipo de vehículo $u \in W$ tiene una capacidad igual a Q_u y un costo fijo f_u.

Cada vértice $j \in V$ está localizado geográficamente en (x_j, y_j), y cada vértice $j \in L \cup B$ tiene z_j bienes de demanda, si $j \in L$, u oferta, si $j \in B$. Se supone $z_0 = 0$.

La versión más general del problema consiste en diseñar un conjunto de rutas de costo mínimo, tales que:

(i) cada vehículo atiende una sola ruta,

(ii) cada cliente es visitado exactamente una vez por únicamente un vehículo.

(iii) los clientes de recolección, si los hay, son atendidos después de los clientes de entrega,

(iv) no se permite que una ruta sólo contenga clientes de recolección,

(v) la carga total asociada tanto a los clientes de entrega, como a los clientes de recolección no debe exceder la capacidad del vehículo que la atiende.

El costo total de la solución es la suma de los costos de todas las rutas, donde, el costo de cada ruta considera el costo fijo del vehículo y un costo unitario por unidad de distancia recorrida en la ruta.

La restricción (iii) se debe a que en los puntos de recolección de bienes habría que reordenar la carga lo cual no se considera factible [Goe89].

La distancia entre cada par de vértices en V es la misma en las dos direcciones.

El PCTF difiere del PRV en que considera una flota heterogénea compuesta por un número ilimitado de vehículos de cada tipo. En el PCTF, además del tamaño de la flota, la combinación de vehículos en la flota debe ser optimizada. El PCTF puede verse como el caso particular del PCTFR en el que no hay clientes de recolección, es decir, cuando $B = \varnothing$.

Aunque el problema de rutas de vehículos se suele usar para modelar casos reales con un único objetivo. En la vida real pueden haber varios costos asociados con las rutas: como el tamaño de la flota, la utilidad recolectada, entre otros.

En el PCTF y en el PCTFR se tienen w tipos de vehículos diferentes y hay un número ilimitado de cada tipo de vehículo. Se puede considerar a estos problemas como problemas multiobjetivo con dos funciones objetivo a minimizar simultáneamente: el número de rutas y el costo total de la solución.

Así, las dos funciones objetivo a minimizar son el número de rutas o vehículos empleados

$$f_1(S) = numR \tag{2.1}$$

Y el costo total de la solución

$$f_2(S) = \sum_{i=1}^{numR} CostoRuta_i \tag{2.2}$$

2.3.1. Casos de prueba

En la cuarta clase hay 36 casos de prueba[12] propuestos por Salhi et al. para el PCTFR [Sal13]. Usaron 12 casos de prueba de Golden et al. con 20, 50, 75 y 100 vértices [Gol84], y, como Toth y Vigo, de cada caso de prueba para la PCTF generaron tres casos de prueba para el PCTFR cada uno correspondiente a un porcentaje de clientes de entrega de 50 %, 66 % y 80 % [Tot97]. El conjunto de vértices se dividió definiendo como cliente de recolección al primero de cada dos, tres y cinco vértices, respectivamente. Al igual que en los casos de la tercera clase se tienen diferentes tipos de vehículos y cada tipo de vehículo tiene una capacidad y un costo fijo.

2.3.2. Revisión de la literatura

Algoritmos heurísticos

Algoritmo de Salhi, Wassan y Hajarat Propusieron una nueva variante del problema de rutas de vehículos a la que llamaron problema de la composición y tamaño de la flota para el problema de rutas de vehículos con recolecciones (PCTFR), en la cual:

(i) hay clientes de entrega y recolección,

(ii) se tiene una flota heterogénea de vehículos W compuesta por un número ilimitado de vehículos de cada tipo. Cada vehículo $u \in W$ tiene un costo fijo f_u y una capacidad Q_u.

(iii) los clientes de recolección, si los hay, son atendidos después de los clientes de entrega,

(iv) no se permite que una ruta sólo contenga clientes de recolección.

El problema consiste en diseñar un conjunto de rutas de costo mínimo que satisfaga la demanda de los clientes y las restricciones.

Asimismo, propusieron 36 casos, los casos de prueba de la cuarta clase.

Presentaron una formulación matemática del problema. Proporcionaron soluciones óptimas de los casos con 20 vértices y cotas superior e inferior para los demás casos, para ello restringieron a 2 horas el uso de la unidad

[12] Vea sitio *Centre for Logistics and Heuristic Optimisation - Research - University of Kent*, la dirección de este sitio es: www.kent.ac.uk/kbs/research/research-centres/clho/. (Datasets, Routing Data Sets, VFM data). Fecha de la última consulta 14/abr/2015.

de procesamiento central en CPLEX. Para resolver el problema propusieron tres heurísticas hibridas basadas en los modelos conjunto cubierta y conjunto partición [Sal13].

Resultados computacionales. Las soluciones óptimas y las cotas las obtuvieron con IBM ILOG CPLEX 12.3 en una computadora personal con un procesador Intel Core 2Duo a 2.8 GHz. Los algoritmos heurísticos los implementaron y ejecutaron en una computadora personal Intel Pentium 4 con un procesador a 3 GHz.

Obtuvieron la solución óptima de los 12 casos con 20 vértices, de la cuarta clase, en dos de ellos (casos HWS10 y HWS12) se requirió correr CPLEX más de 2 horas para obtener la solución óptima. Dentro de las 2 horas de CPLEX, también encontraron la solución óptima de un caso con 50 clientes (caso HWS17). Reportaron las cotas superior e inferior determinadas dentro de las 2 horas de tiempo máximo permitido de los 23 casos restantes.

La heurística 2 es con la que consiguieron mejores resultados, sin embargo, esta consume tanto tiempo de computadora que CPLEX no pudo resolver dos casos con 100 vértices (casos HWS32 y HWS33). Seguida de las heurísticas 3 y 1. Además resolvieron los casos propuestos con el método Sweep-alpha desarrollado por Hajarat y presentaron los resultados obtenidos por el mismo. El tiempo promedio de ejecución en los casos de la cuarta clase del método Sweep-alpha y de las heurísticas 1, 2 y 3 es de 19.48, 62.94, 668.66 y 156.47 segundos, respectivamente. La desviación promedio con respecto a las mejores soluciones encontradas del método Sweep-alpha y de las heurísticas 1, 2 y 3 es de 8.41 %, 6.09 %, 0.37 % y 2.08 %, respectivamente.

Adicionalmente, probaron su propuesta en los casos de la tercera clase, cuando no hay recolección, esto es, cuando a todos los clientes sólo se les entrega mercancía. Sus resultados son 2.31 % peores que los mejores resultados conocidos a la fecha de su estudio. No proporcionaron tiempo de ejecución de los casos de la tercera clase.

Algoritmo de Belloso, Juan y Faulin Presentaron una heurística de inicio múltiple parcialmente aleatoria [Bel17] que usa un método iterativo para resolver el PCTFR propuesto por Salhi et al. [Sal13].

Resultados computacionales. Ejecutaron su algoritmo en una computadora personal con procesador Intel R Core TM i7 CPU M 640 a 2.8 GHz con 3.42 GB de memoria RAM. Probaron su algoritmo con los 36 casos de prueba de la cuarta clase. Mejoraron 20 de las soluciones conocidas. En promedio sus soluciones son 0.54 % mejores que las conocidas. El tiempo promedio de ejecución de un caso es de 75.31 segundos.

Metaheurísticas

Algoritmo de Penna, Subramanian, Ochi, Vidal y Prins Presentaron una metaheurística híbrida que combina BLI con vecindades variables descendentes y una formulación de conjunto partición [Pen17].

Resultados computacionales. Codificaron su algoritmo en C++ y lo ejecutaron en una computadora con un procesador Intel Core i7 a 2.93 GHz con 8 GB de memoria RAM. Se utilizó un único hilo para todas las pruebas. Los modelos conjunto partición se resolvieron utilizando CPLEX 12.5.1. Probaron su algoritmo con 13 variantes del PRVH y con los casos de la cuarta clase.

En los casos de la cuarta clase sólo se compararon con Salhi et al. (aun cuando desde el 19 de febrero de 2017 han estado disponibles los resultados obtenidos, en los casos de la cuarta clase, por Belloso et al). Salhi et al. obtuvieron la solución exacta de 13 de los 36 casos que propusieron. Penna et al. se compararon con las mejores soluciones encontradas por Salhi et al., considerando dentro de éstas, a las 13 soluciones óptimas obtenidas por Salhi et al. En los casos de la cuarta clase: obtuvieron la mejor solución conocida en 13 de 36 casos y mejoraron la solución conocida de 21 casos. En promedio sus soluciones son 1.13 % mejores que las soluciones conocidas a la fecha de publicación de su artículo. El tiempo promedio de ejecución de los algoritmos de Salhi et al., Belloso et al. y Penna et al. es de 668.66, 75.31 y 5.6 segundos respectivamente.

2.4. Problemas multiobjetivo de rutas de vehículos

2.4.1. Revisión de la literatura

La revisión de Jozefowiez [Joz08] de los artículos relacionados con problemas de rutas de vehículos multiobjetivo presenta un visión global de éstos. Entre otras cosas muestra la diversidad de objetivos empleados, por ejemplo:

- Lee y Ueng propusieron un objetivo de balance de la longitud de las rutas para incrementar la equidad de las soluciones producidas [Lee98].

- Feillet, Dejax y Gendreau describieron una clase de problema llamado el Problema del Agente Viajero con Ganancias (PAVG) [Fei05]. En este problema, una ganancia asociada a cada cliente puede ser recolectada cuando el cliente es visitado, pero no es necesario visitar a todos los clientes. Los objetivos del PAVG son: maximizar la ganancia y minimizar la longitud

de la ruta. Muchos enfoques han sido propuestos para resolver el PAVG [Del95, Lap90, Bal89, Kat88, Awe98, Kel88, Kel85].

- Corberán, Fernández, Laguna y Martí [Cor02]; Pacheco y Martí [Pac06] entre otros, minimizaron la longitud de la ruta más larga.

En el capítulo *From Single-Objective to Multi-Objective Vehicle Routing Problems: Motivations, Case Studies, and Methods* [Joz08b], presentan posibles motivaciones para aplicar optimización multiobjetivo a problemas de rutas de vehículos, usos potenciales y los beneficios de hacerlo. Para ilustrar este hecho, describen dos problemas con dos objetivos. Proponen un método para resolverlos y lo implementan para estos problemas. Sugieren definir problemas de rutas de vehículos multiobjetivo generales que puedan ser usados como punto de partida para problemas más complicados.

Metaheurísticas

Algoritmo de Deb, Pratap, Agarwal y Meyarivan Propusieron un procedimiento de ordenamiento no dominado, un procedimiento de estimación de la densidad y un operador de comparación de amontonamiento que emplean en su Algoritmo Genético (AG) denominado Nondominated Sorting Genetic Algorithm II (NSGA-II) [Deb02]. Compararon su algoritmo con otros dos algoritmos PAES (Pareto Archived Evolution Strategy [Kno99]) y SPEA (Strength Pareto Evolutionary Algorithms [Zit99]). Emplearon casos de prueba que no tienen que ver con el tema de esta tesis, por lo que no se examinaron. Mostraron el buen desempeño de su algoritmo. Hicieron notar que algunos algoritmos sólo cubren una parte del frente de Pareto.

Algoritmo de García Nájera Este investigador resolvió el PRVR y el Problema de Rutas de Vehículos con Recolección Selectiva[13] (PRVRS) como problemas multiobjetivo con una adaptación de un Algoritmo Evolutivo Multiobjetivo (AEM) propuesto por García Nájera y Bullinaria [Gar11]. Las tres funciones objetivo que minimizó son el número de rutas, el costo total y la ganancia no recolectada. Propuso cómo construir la población inicial, el proceso de mutación y la forma de reparar las soluciones. Asignó aptitud

[13] En este problema todos los clientes de entrega deben ser visitados, sin embargo, visitar a los clientes de recolección es opcional. Cada cliente de recolección $j \in B$ tiene asociado un beneficio p_j, así $P = \sum_{j \in B} p_j$ es el beneficio total posible. El objetivo del PRVRS es determinar un conjunto de K rutas con costo mínimo, donde K es una constante que se conoce de antemano.

a las soluciones por medio del criterio de clasificación no dominada de Deb et al. [Deb02].

Resultados computacionales. Probó su algoritmo con los 68 casos de la primera clase (25 a 200 clientes), al parecer empleó la matriz de distancias con valores reales y redondeó a dos decimales sus resultados finales. Mejoró la solución conocida de los 6 casos con 200 vértices[14]. En promedio sus soluciones son 0.04 % peores que las mejores soluciones conocidas. El tiempo promedio de ejecución de los casos de esta clase es de 162.39 segundos.

Ejecutó su AEM y el algoritmo NSGA-II [Deb02] en la misma computadora, con los mismos valores en todos los parámetros, procesos y operadores, excepto en diversidad donde cada algoritmo empleó su propio mecanismo de diversidad.

Para el problema biobjetivo (minimizar el número de rutas y el costo total) comparó su AEM con el algoritmo NSGA-II por medio de las métricas hipervolumen [Zit98] y diversidad [Gar12] sobre las soluciones no dominadas encontradas por cada algoritmo. Se aplicó la prueba t de Student sobre los valores hipervolumen y diversidad, no se encontró una diferencia significativa, por lo que se concluye que ambos algoritmos se desempeñan igualmente bien en el problema biobjetivo.

En el problema triobjetivo también se aplicó las métricas hipervolumen y diversidad a las soluciones no dominadas encontradas por los algoritmos. La prueba t de Student de los valores hipervolumen mostró que hay una diferencia significativa en 21 casos, en 8 de ellos las soluciones del AEM tiene un hipervolumen mayor y en los 13 restantes las soluciones del algoritmo NSGA-II son mejores respecto al hipervolumen. La prueba t de Student aplicada a los valores de diversidad mostró que en 49 casos las soluciones del AEM son significativamente más diversas que las del algoritmo NSGA-II.

Algoritmo de Anbuudayasankar, Ganesh, Lenny y Ducq Este equipo de investigadores propuso el siguiente problema biobjetivo de rutas de vehículos con recolecciones forzadas (PBRVRF): un conjunto de N nodos que demanda servicios de entrega o recolección debe ser visitado por un una flota de vehículos homogéneos ubicados en el almacén. Hay un grupo de nodos de recolección que es forzoso visitar, la secuencia de visita a estos nodos es asignada por la optimización. Los objetivos del PBRVRF son minimizar: el costo total y la duración total de las rutas [Anb12].

[14] En el sitio *Linehaul - Backhaul,* cuya dirección es: http://www2.isye.gatech.edu/~mgoetsch/lineback.html. Se pueden descargar las soluciones obtenidas por Goetschalckx y Jacobs-Blecha a los 68 casos de prueba propuestos por ellos mismos. Fecha de la última consulta 22/jul/2015.

Para resolver el problema desarrollaron un AG y dos heurísticas: ahorro modificado con eliminación de arco (AMEA) y ahorro modificado con intercambio de nodo (AMIN). En cada caso emplearon sumas ponderadas para resolver el problema biobjetivo.

Resultados computacionales. Implementaron su algoritmo en una computadora personal con un procesador de 1.7 GHz Pentium IV. Probaron sus algoritmos con 10 de los 68 casos de la primera clase. No especificaron si calcularon la distancia entre clientes con reales o con enteros, reportaron sus resultados con enteros. Con el algoritmo que obtuvieron mejores resultados en cada uno de los casos fue el AG, después con la heurística AMIN y por último con la heurística AMEA. En todos los casos la mejor solución encontrada por estos investigadores es peor que la mejor solución conocida. En promedio las soluciones de su AG son 15.9 % peores. El tiempo promedio de ejecución en los casos de esta clase de su AG es de 68.7 segundos.

Asimismo, probaron sus algoritmos con todos los casos de prueba de la segunda clase. Nuevamente los mejores resultados los obtuvieron con el AG, seguido de las heurísticas AMIN y AMEA. En promedio las soluciones de su AG son 2.7 % peores que las mejores soluciones conocidas. El tiempo promedio de ejecución de su AG en los casos de esta clase es de 43.1 segundos.

También probaron su algoritmo con nueve casos reales de la industria bancaria del PBRVRF. Emplearon tiempo del recorrido en lugar de distancia porque la distancia no considera el tráfico. En cada uno de estos casos se tienen 5 vehículos disponibles de capacidad 30, el número total de vértices está entre 21 y 100. Reportaron, para cada caso, el tiempo total del recorrido y tiempo del recorrido de la ruta más larga de sus tres algoritmos.

Algoritmo de García-Nájera, Bullinaria, Gutiérrez-Andrade Resolvieron las siguientes cinco variantes: PRVR, PRVRM, PRVERS, PRVRS, y PRV con recolección mixta y selectiva (PRVRMS) como problemas de optimización con dos y tres objetivos con un algoritmo evolutivo multiobjetivo de selección basada en similitud (AEMSBS) [Gar15], el cual se fundamenta en una aplicación anterior de técnicas de computación evolutiva al PRV con ventanas de tiempo [Gar11]. Las tres funciones objetivo que minimizaron son el número de rutas, el costo total del recorrido y la ganancia no recolectada total. Diseñaron una representación general de la solución y un conjunto de operadores evolutivos para el PRVR, y crearon procedimientos modificados de generación de la solución inicial y mutación para resolver el PRVRM, PRVERS, PRVRS, y PRVRMS.

Resultados computacionales. Probaron su algoritmo con los primeros

62 casos de prueba de la primera clase. Compararon su AEMSBS con los algoritmos NSGA-II y AEMBD (Algoritmo Evolutivo Multiobjetivo Basado en Descomposición) [Zha07]. Los tres algoritmos antes mencionados resolvieron 30 veces cada caso de prueba para proveer estadísticas confiables.

NSGA-II fue implementado usando los mismos operadores de selección, cruce y mutación que se diseñaron para el AEMSBS. NSGA-II elige a ambos padres de acuerdo a la aptitud más alta. El AEMSBS elige a uno de los padres con base en la aptitud más alta y el otro según la similitud de solución más baja. Para determinar la supervivencia, el AEMSBS toma en cuenta la similitud de la solución, mientras que NSGA-II emplea una medida de distancia de amontonamiento.

Se seleccionó el AEMBD porque es uno de los mejores algoritmos multiobjetivos actuales y se ha demostrado que ofrece un desempeño mejorado en dos problemas estrechamente relacionados con el PRV [Mei11, Pen09]. El AEMBD se implementó con los mismos operadores de cruce y mutación que el AEMSBS.

Para hacer una comparación lo más justa posible, los valores de los parámetros para NSGA-II son iguales a los del AEMSBS. Una comparación justa con el AEMBD fue más difícil. En particular: una distribución razonable de los vectores de peso y los subproblemas correspondientes, que emplea el AEMBD, a veces significa permitir un mayor tamaño de población para el AEMBD, que para el AEMSBS y NSGA-II; como el AEMBD mantiene un archivo de soluciones, tiene otra ventaja sobre el AEMSBS y NSGA-II [Pen09]; el procedimiento de mejora del AEMBD fue reemplazado por la misma etapa de mutación que el AEMSBS y NSGA-II, para evitar otra ventaja injusta [Mei11].

Desempeño del AEMSBS en el PRVR con un objetivo

El AEMSBS se configuró para minimizar el número de rutas y el costo total del recorrido simultáneamente, para resolver los primeros 62 casos de prueba de la primera clase del PRVR como problemas biobjetivo. Compararon el desempeño del AEMSBS con el de algoritmos para el PRVR con un objetivo. Para cada caso de prueba sólo la solución del AEMSBS con K rutas y el menor costo fue comparada con la mejor solución conocida. En promedio las soluciones del AEMSBS son 0.41 % peores que las mejores soluciones conocidas. Por lo que el algoritmo multiobjetivo propuesto, AEMSBS, no es mucho peor que los algoritmos para el PRVR con un objetivo, a pesar de la desventaja de trabajar para difundir las soluciones en la región entera del frente de Pareto. El AEMSBS encontró soluciones con menos de K rutas en 17 casos de prueba, 11 de estas soluciones tienen un costo menor que el mejor costo conocido, es decir, dominan a las mejores soluciones conocidas.

Las seis soluciones restantes no tienen un costo menor al mejor conocido, pero tienen menos rutas, por lo que son no dominadas por las mejores soluciones conocidas.

PRVR biobjetivo, minimizar el número de rutas y el costo total del recorrido

Para cada caso de prueba y cada repetición del AEMSBS, NSGA-II y AEMBD se calculó el hipervolumen cubierto por el conjunto de soluciones no dominadas. Se aplicó la prueba t de Student a los dos pares de vectores de 30 valores del hipervolumen, de AEMSBS y NSGA-II, y AEMSBS y AEMBD, respectivamente, para determinar si la diferencia en el hipervolumen promedio fue estadísticamente significativa con un nivel de confianza del 95 %. Para cada aproximación de Pareto la diversidad de las soluciones no dominadas fue calculada y se usó una prueba t de Student (de la misma manera que para el hipervolumen) para determinar si las diferencias en diversidad fueron significativas.

La diferencia en el hipervolumen cubierto por las soluciones no dominadas del AEMSBS y NSGA-II es significativa en cinco casos de prueba, en todos ellos las soluciones del AEMSBS son mejores. En 23 casos de prueba hay una diferencia significativa entre el hipervolumen cubierto por el AEMSBS y el AEMBD, con 17 casos de prueba en los que el AEMSBS es mejor y seis en los que el AEMBD es mejor. Respecto a la diversidad de la población, hay una diferencia significativa entre las soluciones encontradas por el AEMSBS y NSGA-II en cuatro casos de prueba, el tres de ellos AEMSBS es superior, y en uno NSGA-II es superior. Hay más diferencias significativas entre el AEMSBS y el AEMBD, el AEMSBS tiene una mayor diversidad en 37 casos de prueba, y el AEMBD es superior en dos casos.

En todos los casos de prueba el tamaño promedio de la aproximación de Pareto es no mucho más que uno para los tres algoritmos, lo que sugiere que los dos objetivos rara vez están en conflicto.

PRVRS biobjetivo, minimizar el costo total del recorrido y la ganancia no recolectada total

Para cada caso de prueba y cada repetición del AEMSBS, NSGA-II y AEMBD se calcularon el hipervolumen y la diversidad del conjunto de soluciones no dominadas. Después, para cada caso de prueba, estas medidas de desempeño fueron sometidas a una prueba t de Student como anteriormente.

La diferencia en el hipervolumen cubierto por el conjunto de soluciones no dominadas del AEMSBS y NSGA-II es significativa en 36 de 62 casos de prueba, las soluciones del AEMSBS definen un hipervolumen más grande que las de NSGA-II en todos estos casos. Además, las soluciones del AEMSBS delimitan un hipervolumen significativamente más grande que las del AEM-

BD en los 62 casos de prueba. Respecto a la diversidad de la población, hay una diferencia significativa entre el AEMSBS y NSGA-II en 59 casos de prueba, la diversidad de las soluciones del AEMSBS es mayor en todos estos casos. Asimismo, las soluciones del AEMSBS son significativamente más diversas que las del AEMBD en los 62 casos de prueba. Una mayor diversidad en el conjunto de soluciones del AEMSBS se correlaciona con un mayor hipervolumen.

PRVRS triobjetivo, minimizar el número de rutas, el costo total del recorrido y la ganancia no recolectada total

Los tres algoritmos fueron probados en el PRVRS triobjetivo. Como antes, los conjuntos de soluciones no dominadas fueron registrados para cada ejecución del algoritmo, se calcularon los hipervolumenes y las diversidades y fueron sometidos a pruebas t de Student.

El AEMSBS delimita hipervolumenes que son significativamente mejores que NSGA-II en 38 casos de prueba y mejores que AEMBD en los 62 casos de prueba, mientras que las soluciones de NSGA-II tienen un mejor hipervolumen en un único caso de prueba. Respecto a la diversidad de las soluciones, las soluciones del AEMSBS son significativamente más diversas que las de NSGA-II en 60 de 62 casos de prueba, y en 61 casos de prueba en comparación con las soluciones del AEMBD.

Mostraron que el AEMSBS y el AEMBD fueron más apropiados que NSGA-II para encontrar soluciones extremas en donde todos los clientes de recolección fueron atendidos a expensas del o de los otros objetivos.

García-Nájera et al. también proporcionaron resultados computacionales de otras dos variantes del problema con uno, dos y tres objetivos, pero no se analizarán aquí por no ser el tema central de esta tesis.

2.5. Resumen

Se definieron formalmente tres variantes del PRV: el PRVR, el PCTF y el PCTFR. Las dos primeras variantes pueden considerarse casos particulares del PCTFR. Si en el PCTFR se considera una flota compuesta por exactamente K vehículos de un único tipo, cuyo costo fijo es cero, el problema se reduce al PRVR. Asimismo, si en el PCTFR se considera que no hay clientes de recolección, el problema se reduce al PCTF. El objetivo del PCTFR es diseñar un conjunto de rutas de costo mínimo, pero puede haber otras funciones objetivo a optimizar, por ejemplo, la duración total de las rutas o el número de rutas. En la revisión de la literatura se encontró que se han optimizado el balance de la longitud de las rutas, la ganancia no

recolectada, o la longitud de la ruta más larga.

En los estudios revisados de problemas con un objetivo, la metaheurística más empleada fue BT. Muchos de los estudios, entre ellos [Osm02, Gaj09, Sub12, Pen17], emplearon una o más de las siguientes heurísticas de mejora: 2-opt, 3-opt e intercambios de clientes entre rutas.

Los algoritmos híbridos [Sub12, Pen17] resolvieron varias variantes del PCTF y PRVH, respectivamente, y obtuvieron soluciones que superan a las mejores soluciones conocidas, en un tiempo de ejecución muy corto. Estos algoritmos generan rutas por medio de una metaheurística, que se utilizan para crear un conjunto de rutas prometedoras para el modelo conjunto partición (MCP). El MCP se resuelve luego mediante un solucionador de programación entera mixta, que interactúa con la metaheurística durante su ejecución. Las metaheurísticas empleadas por [Sub12, Pen17] son búsqueda local iterada y búsqueda local iterada con vecindades variables descendentes, respectivamente.

Los mejores algoritmos para el PRVR, el PCTF y el PCTFR son [Gaj09, Sub12, Pen17] respectivamente. Aunque hay un par de casos de prueba de la cuarta clase en los que la solución obtenida en [Sal13] es mejor que la obtenida en [Pen17].

De los cuatro estudios multiobjetivo revisados, uno [Deb02] no resolvió problemas de rutas de vehículos. En uno de ellos [Anb12] compararon sus resultados con los de algoritmos de problemas con un único objetivo. Los cuatro estudios multiobjetivo emplearon algoritmos evolutivos.

La sugerencia y posible conclusión de [Gar15], en el PRVR biobjetivo, minimizar el número de rutas y el costo total del recorrido, respecto a que estos objetivos rara vez están en conflicto, es de tomarse en cuenta, pues según se observó no se detectaron las bondades de su algoritmo en este problema. Si lo que se quiere es probar un algoritmo multiobjetivo, hay que emplear un problema adecuado, uno donde los objetivos estén en conflicto.

Capítulo 3

Optimización multiobjetivo

Optimizar es encontrar una o más soluciones factibles correspondientes a valores extremos de uno o más objetivos. La necesidad de encontrar tales soluciones extremas en un problema proviene en su mayoría de propósitos en el diseño de una solución tales como encontrar el costo mínimo de fabricación, la confiabilidad máxima posible, u otros.

La mayoría de los problemas de búsqueda y optimización del mundo real implican naturalmente múltiples objetivos. El principio extremista antes mencionado no puede ser aplicado a un solo objetivo, cuando el resto de los objetivos también son importantes. Diferentes soluciones pueden producir compensaciones entre objetivos diferentes. Una solución que es extrema con respecto a un objetivo requiere transigir con otros objetivos. Esto prohíbe elegir una solución que es óptima con respecto a un único objetivo.

Considere la toma de decisión involucrada al elegir en qué banco invertir a plazo fijo una suma dada de dinero. Suponga que la tasa bruta anual que otorgan los bancos va del 2 % al 7.51 % (ver Figura 3.1). Considere las dos tasas extremas, es decir, la del 2 % (solución A) y la del 7.51 % (solución F). Si la tasa es el único objetivo de esta toma de decisión, la elección óptima es la solución F. Salvo excepciones, se espera que la tasa más alta sea la más difícil de obtener. La Figura 3.1 indica que obtener la tasa más alta tiene un hipotético grado de dificultad de 6. Para quien desee invertir su dinero con el menor grado de dificultad, la elección es la solución A (con un hipotético grado de dificultad de 1). Entre estas dos soluciones extremas, existen muchas otras soluciones, con una compensación entre tasa (rendimiento) y dificultad. Varias de estas soluciones, con diferentes tasas y niveles de dificultad, se muestran en esta figura. Entre cualesquiera dos de tales soluciones, una es mejor en términos de un objetivo, pero esta mejoría

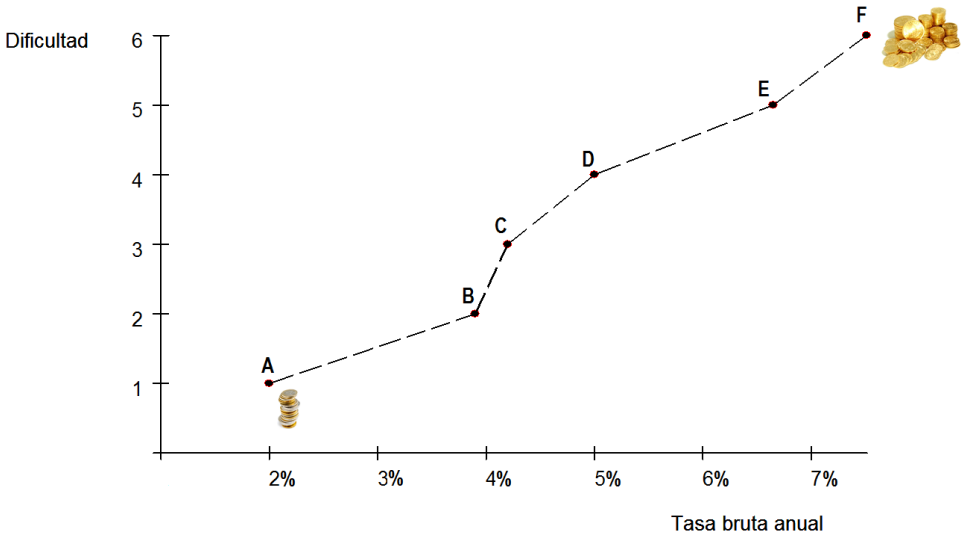

Figura 3.1: Hipotéticas soluciones de compensación para el problema de la toma de decisión de en qué banco invertir.

proviene de un sacrificio en el otro objetivo [Deb01].

3.1. Problema de optimización multiobjetivo

Un problema de optimización multiobjetivo tiene varias funciones objetivo, las cuales han de ser minimizadas o maximizadas, y restricciones que cualquier solución factible debe satisfacer. La forma general de un problema de optimización multiobjetivo es la siguiente [Deb01]:

$$\left.\begin{array}{rlc} \text{Minimizar/Maximizar} & f_u(x) & u = 1, 2, \ldots, a \\ \text{sujeto a} & g_j(x) \geq 0 & j = 1, 2, \ldots, J \\ & h_w(x) = 0 & w = 1, 2, \ldots, W \\ & x_i^L \leq x_i \leq x_i^U & i = 1, 2, \ldots p \end{array}\right\} \quad (3.1)$$

Una solución x es un vector de p variables de decisión: $x = (x_1, x_2, \ldots, x_p)^T$. El último conjunto de restricciones se denomina límites de las variables, restringe cada variable de decisión x_i a tomar un valor entre los límites inferior y superior, x_i^L y x_i^U respectivamente. Una solución x que satisface las

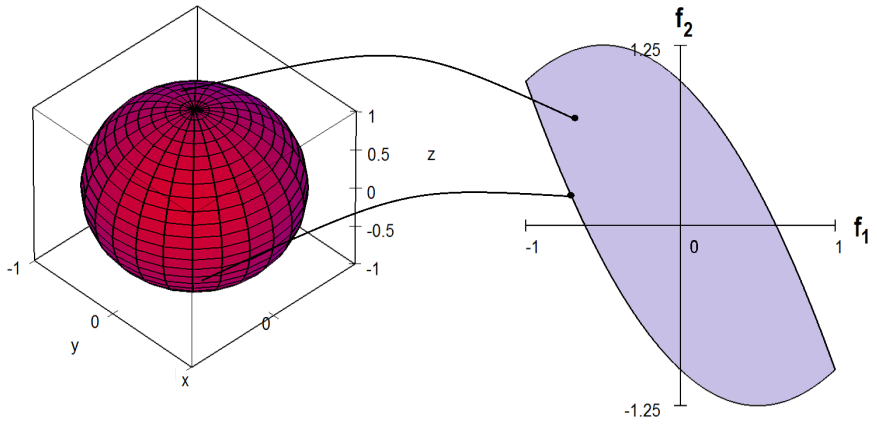

Figura 3.2: Espacios factible de dimensión 3 y espacio objetivo de dimensión 2 para el Ejemplo 3.1.

$J + W + 2p$ restricciones se denomina solución factible, mientras que una solución que no satisface alguna de las $J+W+2p$ restricciones se denomina solución no factible. El conjunto de todas las soluciones factibles es llamado el espacio factible de las variables de decisión S o región factible.

En la optimización multiobjetivo, las funciones objetivo constituyen un espacio multidimensional. Este espacio es llamado espacio objetivo Z. Para cada solución x en el espacio de las variables de decisión, existe un punto en el espacio objetivo. La asignación se lleva a cabo entre un vector solución $p-dimensional$ y un vector objetivo $a-dimensional$.

Ejemplo 3.1. Considere el siguiente problema de optimización

$$\begin{aligned} \text{Minimizar} \quad & f_1(x,y,z) = z \\ \text{Minimizar} \quad & f_2(x,y,z) = -x^2 - y^2 - z \\ \text{sujeto a} \quad & \\ & x^2 + y^2 + z^2 \leq 1 \end{aligned}$$

En la Figura 3.2 se muestran el espacio factible de las variables de decisión (izquierda) y el espacio objetivo (derecha). Hay un mapeo entre estos dos espacios. A un vector factible de dimensión 3 se asigna un vector objetivo de dimensión 2. Por ejemplo al vector $(x,\, y,\, z) = (\frac{1}{2},\, \frac{1}{2},\, -\frac{1}{\sqrt{2}})$ se asigna $(f_1(x,y,z) = -\frac{1}{\sqrt{2}}, f_2(x,y,z) = \frac{1}{\sqrt{2}} - \frac{1}{2})$.

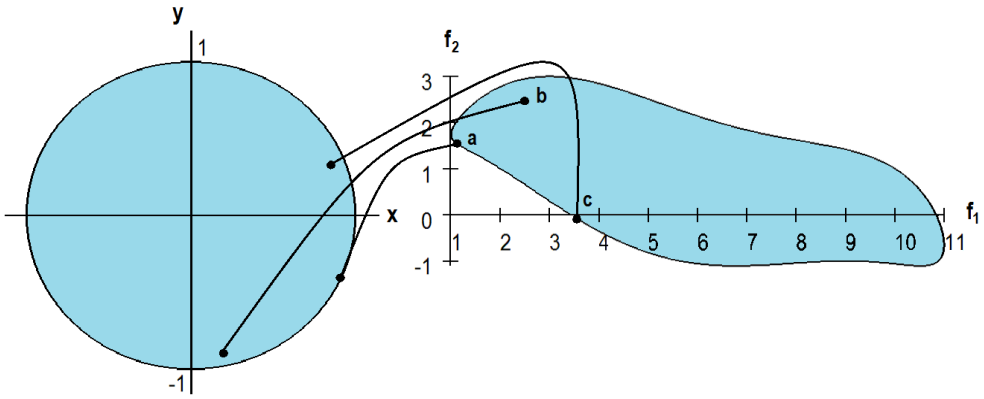

Figura 3.3: Espacios factible (izquierda) y objetivo (derecha) para el Ejemplo 3.2.

Ejemplo 3.2. Considere el siguiente problema de optimización con dos objetivos

$$\begin{aligned} \text{Minimizar} \quad & f_1(x,y) = 6 - 4x + 3y \\ \text{Minimizar} \quad & f_2(x,y) = x^4 + y^4 - 2y \\ \text{sujeto a} \quad & \\ & x^2 + y^2 \leq 1 \end{aligned}$$

En la Figura 3.3 se presentan, del lado izquierdo, el espacio factible de las variables de decisión y, a la derecha, el espacio objetivo. Asimismo, se muestran tres soluciones factibles y sus respectivas imágenes en el espacio objetivo. Entre las soluciones **a** y **b**, dado que se quiere minimizar tanto a f_1 como a f_2, **a** es mejor que **b** con respecto a ambos objetivos. Mientras que, si se compara **a** con **c**, **a** es mejor que **c** con respecto al objetivo f_1, pero **c** es mejor que **a** respecto a f_2. En el caso de las soluciones **a** y **c** se dice que las soluciones son no dominadas (más adelante se define cuando una solución no domina a otra). Las soluciones **a** y **c** se denominan soluciones óptimas de Pareto. La curva formada por la unión de estas soluciones se conoce como frente óptimo de Pareto. Aunque, no en todos los problemas se forma una curva, pueden haber frentes discontinuos.

3.2. Objetivos que no están en conflicto

En la optimización multiobjetivo, cuando los objetivos están en conflicto, hay más de una solución óptima de Pareto, pero si los objetivos no están

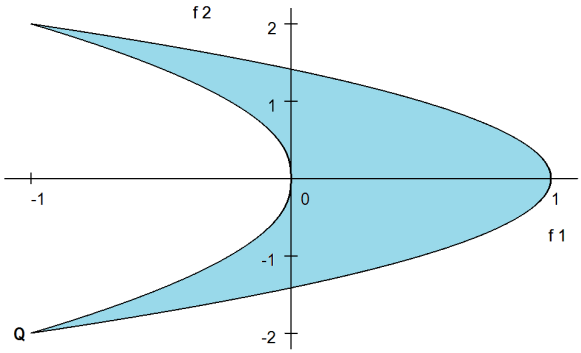

Figura 3.4: Las funciones objetivo f_1 y f_2 no están en conflicto, por ello dan lugar a una única solución óptima **Q**.

en conflicto, puede haber una única solución óptima de Pareto aun cuando este hecho no resulte evidente.

Ejemplo 3.3. Considere el siguiente problema de optimización:

$$\begin{aligned} \text{Minimizar} \quad & f_1(x,y) = -xy \\ \text{Minimizar} \quad & f_2(x,y) = x + y \\ \text{sujeto a} \quad & \\ & x^2 + y^2 \leq 2 \end{aligned}$$

La Figura 3.4 muestra que el frente óptimo de Pareto de éste problema consta de una única solución óptima, Q.

Un algoritmo de optimización multiobjetivo tiene dos metas:

1. Encontrar un conjunto de soluciones cuyas funciones objetivo estén tan cercanas como sea posible al frente óptimo de Pareto.

2. Encontrar un conjunto de soluciones tan diverso como sea posible.

Un algoritmo de optimización multiobjetivo debe incluir rutinas explícitas o implícitas a cargo de estas dos metas.

La optimización multiobjetivo trata con dos espacios, el de las variables de decisión y el espacio objetivo, en ambos puede definirse la diversidad entre soluciones. La diversidad entre dos soluciones en el espacio de las variables de decisión indica que tan diferentes son. Por otra parte, dos soluciones en el espacio objetivo son diversas si la distancia euclidiana entre sus evaluaciones de las funciones objetivo, en este espacio, es grande.

Aunque el espacio objetivo y el de las variables de decisión están relacionados por un mapeo, frecuentemente este mapeo es no lineal, por lo que la proximidad de dos puntos en el espacio de las variables de decisión no significa proximidad de las imágenes de dichos puntos en el espacio objetivo. El proceso de un algoritmo en el espacio de las variables de decisión puede ser rastreado en el espacio objetivo. En algunos algoritmos, el proceso resultante en el espacio objetivo se usa para dirigir la búsqueda en el espacio de las variables de decisión. Cuando esto ocurre, el proceso en ambos espacios debe ser llevado a cabo de manera que los puntos creados en el espacio de las variables de decisión estén a favor de la diversidad necesaria en el espacio objetivo.

Ahora se definen varios conceptos que se emplean en optimización multiobjetivo.

En general cada uno de los a objetivos en conflicto tiene una solución óptima distinta. El vector objetivo ideal se forma con estos a valores objetivos óptimos.

Definición 3.1. Las coordenadas del vector objetivo ideal se obtienen minimizando cada función objetivo por separado, esto es, la u-ésima componente del vector objetivo ideal z^* es la solución del siguiente problema:

$$\begin{aligned} \text{Minimizar} \quad & f_u(x) \\ \text{sujeto a} \quad & x \in S \end{aligned}$$

Si la solución mínima de la u-ésima función objetivo es el vector x_u^* y $f_u(x_u^*) = f_u^*$, el vector objetivo ideal es (ver Figura 3.5):

$$z^* = f^* = (f_1^*,\ f_2^*, \ldots, f_a^*)$$

Usualmente no existe una solución $x \in S$ que corresponda al vector objetivo ideal. Si existiera, la solución mínima de todos los objetivos sería la misma, los objetivos no estarían en conflicto y la solución del problema sería única. El vector objetivo ideal se emplea como referencia. Muchos algoritmos requieren una cota inferior para cada función objetivo para normalizar los valores objetivos a una escala común.

Definición 3.2. Un vector objetivo utópico z^{**} tiene cada una de sus componentes marginalmente más pequeña que la del vector objetivo ideal, es decir, $z_i^{**} = z_i^* - \epsilon_i$ con $\epsilon_i > 0$ para todo $i \in \{1, \ldots, a\}$, $z_i^* = f_i^*$ (ver Figura 3.5).

Definición 3.3. El vector objetivo nadir z^{nad} constituye una cota superior de cada objetivo en la región entera del frente de Pareto, no en el espacio de búsqueda entero.

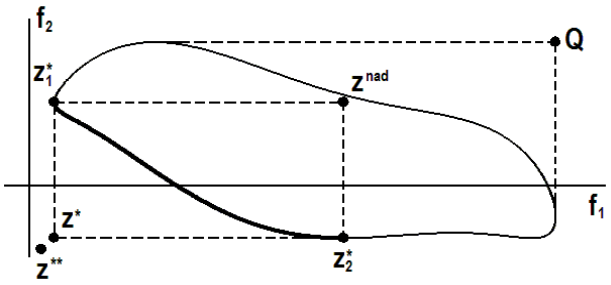

Figura 3.5: Vectores objetivo ideal (z^*), utópico (z^{**}) y nadir (z^{nad}).

Calcular el vector objetivo nadir es difícil. Para dos objetivos, si $z_1^* = (f_1(x_1^*), f_2(x_1^*))$ y $z_2^* = (f_1(x_2^*), f_2(x_2^*))$ son las soluciones mínimas en el espacio objetivo de f_1 y f_2, respectivamente, entonces se puede estimar el vector objetivo nadir como $z^{nad} = (f_1(x_2^*), f_2(x_1^*))$, ver Figura 3.5.

Dependiendo de la convexidad y continuidad del frente óptimo de Pareto, el vector objetivo nadir puede corresponder o no a una solución.

No se debe confundir el vector objetivo nadir con el vector $Q = (q_1, q_2)$ de la Figura 3.5, donde q_i es el máximo de la función objetivo f_i en el espacio de búsqueda entero.

Los vectores objetivo ideal y nadir pueden emplearse para normalizar cada objetivo en el rango completo de la región óptima de Pareto:

$$f_i^{norm} = \frac{f_i - z_i^*}{z_i^{nad} - z_i^*}$$

3.3. Dominación

Definición 3.4. La solución x_1 domina a la solución x_2, si:

1. La solución x_1 no es peor que x_2 en todos los objetivos, es decir, $f_j(x_1)$ no es peor que $f_j(x_2)$ para toda $j = 1, 2, \ldots, a$.

2. La solución x_1 es estrictamente mejor que x_2 en al menos un objetivo, es decir, $f_j(x_1)$ es mejor que $f_j(x_2)$ para al menos una $j = 1, 2, \ldots, a$.

Si se viola alguna de las dos condiciones anteriores, la solución x_1 no domina a la solución x_2. La solución x_1 domina a la solución x_2 se denota por $x_1 \preceq x_2$.

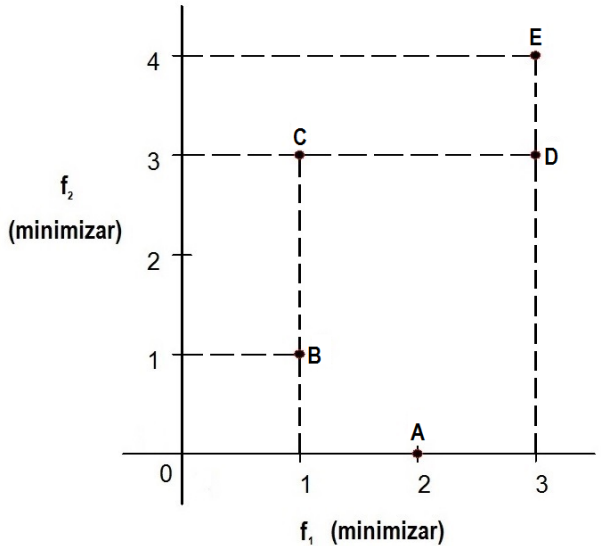

Figura 3.6: Una población con cinco soluciones.

Ejemplo 3.4. Considere un problema de optimización con dos objetivos y cinco soluciones diferentes (ver Figura 3.6). Suponga que las funciones objetivo f_1 y f_2 necesitan ser minimizadas. Como ambas funciones objetivo son importantes, es difícil encontrar una solución que sea la mejor con respecto a ambos objetivos. Sin embargo, se puede usar la Definición 3.4 de dominación para decidir cuál solución es mejor entre cualesquiera dos soluciones dadas en términos de ambos objetivos. Si las soluciones B y C son comparadas, B no es peor que C con respecto a ambas funciones objetivo y B es mejor que C respecto a la función objetivo 2. Como se cumplen las dos condiciones de la Definición 3.4, se dice que la solución B domina a la solución C. Si se comparan las soluciones B y D, B no es peor que D respecto a ambas funciones objetivo y B es mejor que D respecto a ambas funciones objetivo. Las dos condiciones de dominación de la Definición 3.4 se satisfacen, por lo que la solución B domina a la solución D. Si se comparan las soluciones B y A, se observa que B es mejor que A respecto a la función objetivo 1 y B es peor que A respecto a la función objetivo 2. Así, la primera condición de la Definición 3.4 no se cumple, no se puede concluir que la solución B domina a la solución A, ni que la solución A domina a la solución B. Cuando esto ocurre, se dice que las soluciones A y B son no dominadas una con respecto a la otra. Cuando ambos objetivos son importantes, no se puede decir cuál

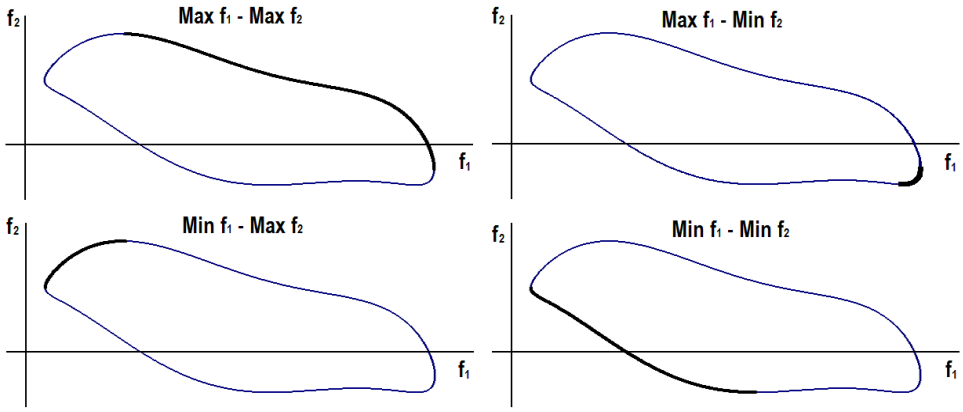

Figura 3.7: En cada problema está marcado con línea gruesa el frente óptimo de Pareto.

de las dos soluciones es mejor.

Para un conjunto finito de puntos dados del espacio objetivo, se pueden efectuar todas las posibles comparaciones por pares y encontrar cuáles soluciones dominan a cuáles otras y cuáles son no dominadas una con respecto a la otra. Al final, se espera tener un conjunto de soluciones, dos cualesquiera de las cuales son no dominadas una con respecto a la otra. Además, para cualquier solución fuera de este conjunto, siempre se puede encontrar una solución en este conjunto que domina a la primera. Esto significa que las soluciones en este conjunto son mejores comparadas con el resto de las soluciones. Este conjunto es llamado conjunto no dominado del conjunto de soluciones dado. En el ejemplo anterior, las soluciones A y B constituyen el conjunto no dominado del conjunto de cinco soluciones dado.

Definición 3.5. (Conjunto no dominado) Entre un conjunto de soluciones P, el conjunto no dominado de soluciones P^* es aquel que no es dominado por ningún miembro de P.

Definición 3.6. (Conjunto globalmente óptimo de Pareto) El conjunto no dominado de todo el espacio de búsqueda factible S, es llamado conjunto óptimo de Pareto o conjunto globalmente óptimo de Pareto.

En la Figura 3.7 está marcado con línea gruesa el frente óptimo de Pareto de los cuatro posibles problemas correspondientes a dos funciones objetivo. El frente óptimo de Pareto siempre se encuentra en una región particular de la frontera del espacio objetivo.

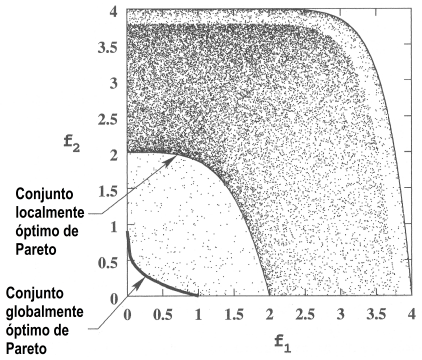

Figura 3.8: Problema biobjetivo con un conjunto localmente óptimo de Pareto y un conjunto globalmente óptimo de Pareto.

Empleando la definición de dominación un algoritmo multiobjetivo puede tratar cualquiera de los cuatro casos a que hace referencia la Figura 3.7, sin embargo, para evitar confusiones, generalmente los problemas de maximización se convierten en problemas de minimización y se trata el problema multiobjetivo como un problema en donde cada uno de los objetivos es de minimización.

Definición 3.7. Si para cada $x \in P^{\blacklozenge}$ no existe y en una vecindad de x, tal que $\|y - x\| \leq \epsilon$, con $\epsilon > 0$, que domine a algún elemento de $P^{\blacklozenge}$, entonces $P^{\blacklozenge}$ constituye un conjunto localmente óptimo de Pareto.

En la Figura 3.8 [Deb01, p. 351] se presenta un problema biobjetivo con un conjunto no convexo localmente óptimo de Pareto y un conjunto convexo globalmente óptimo de Pareto.

Definición 3.8. La solución x_1 domina fuertemente a la solución x_2 (o $x_1 \prec x_2$), si la solución x_1 es estrictamente mejor que la solución x_2 en todos los a objetivos.

3.4. Métricas de desempeño

Los primeros algoritmos multiobjetivo demostraban su funcionamiento mostrando las soluciones no dominadas obtenidas junto con las auténticas soluciones óptimas de Pareto. En esos estudios se daba énfasis a mostrar que tan cerca estaban las soluciones obtenidas del frente óptimo de Pareto. Actualmente existen muchos algoritmos multiobjetivo distintos, para compararlos es necesario hacerlo en términos de su desempeño.

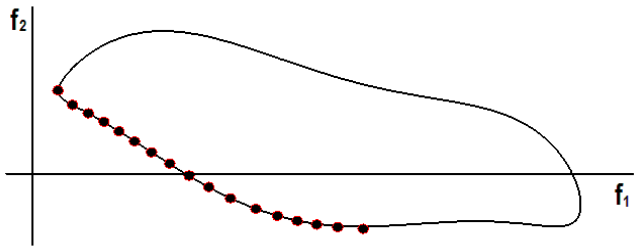

Figura 3.9: Conjunto de soluciones no dominadas ideal.

Como ya se había mencionado antes, hay dos metas en la optimización multiobjetivo: (1) Encontrar soluciones tan cerca de las soluciones óptimas de Pareto como sea posible, y (2) encontrar soluciones tan diversas como sea posible en el frente no dominado obtenido. La primera meta requiere una búsqueda hacia la región óptima de Pareto, mientras que la segunda meta requiere una búsqueda a lo largo del frente óptimo de Pareto. Asimismo, la diversidad puede ser separada en dos medidas diferentes: grado (extensión de las soluciones extremas) y distribución (distancia relativa entre soluciones) [Zit00].

Un algoritmo multiobjetivo se denomina bueno, si ambas metas se satisfacen adecuadamente. Con un buen algoritmo multiobjetivo, un usuario espera encontrar tanto soluciones cercanas al verdadero frente de Pareto, como soluciones que se esparzan uniformemente y abarquen la región entera del frente de Pareto. La Figura 3.9 muestra el desempeño de un algoritmo multiobjetivo ideal. Claramente todas las soluciones no dominadas obtenidas caen en el frente óptimo de Pareto, están uniformemente distribuidas y abarcan la región entera del frente de Pareto.

Las Figuras 3.10 y 3.11 presentan los conjuntos no dominados de soluciones obtenidas, al resolver un mismo problema, por los algoritmos S y T, respectivamente. El algoritmo S obtiene soluciones que convergen bastante bien al frente óptimo de Pareto, pero que no tienen una buena distribución entre sí. Mientras que el algoritmo T tiene una buena distribución entre soluciones, pero sus soluciones no están cerca del verdadero frente óptimo de Pareto.

Converger al frente óptimo de Pareto y conservar la diversidad del conjunto de soluciones son dos metas distintas y un tanto conflictivas de la optimización multiobjetivo, no hay una sola métrica que pueda decidir el desempeño de un algoritmo en un sentido absoluto. Hay una clara necesidad

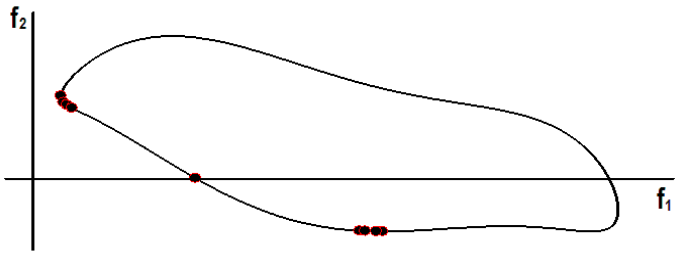

Figura 3.10: Las soluciones no dominadas del algoritmo S tienen buena convergencia, pero mala distribución.

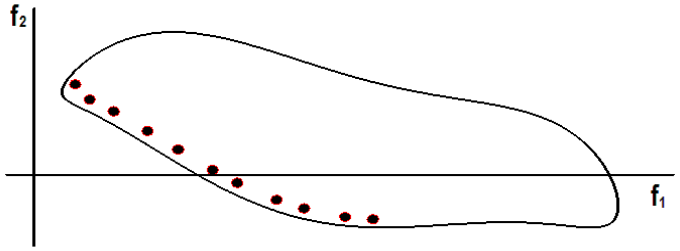

Figura 3.11: Las soluciones no dominadas del algoritmo T tienen buena distribución, pero mala convergencia.

de tener al menos dos métricas de desempeño para evaluar adecuadamente ambos objetivos de la optimización multiobjetivo.

En las siguientes secciones se presentan algunas métricas comúnmente usadas en la literatura. La primera métrica que se abordará puede ser usada para medir el progreso hacia el frente óptimo de Pareto explícitamente. La segunda métrica puede ser usada para medir la diversidad entre las soluciones obtenidas explícitamente. La tercera métrica usa dos tipos de métricas que miden ambas metas de la optimización multiobjetivo de manera implícita.

3.4.1. Métrica del conjunto cobertura

Esta métrica puede ser usada para darse una idea de la extensión relativa de las soluciones entre dos conjuntos de soluciones T y Q. La métrica del conjunto cobertura $M_1(T, Q)$ [Zit00, Deb01] calcula la porción de soluciones en Q, que son dominadas por soluciones de T.

$$M_1(T, Q) = \frac{|\{x_2 \in Q | \exists\ x_1 \in T : x_1 \preceq x_2\}|}{|T|}$$

El valor métrico $M_1(T, Q) = 1$ significa que todos los elementos de Q son dominados por T. Y $M_1(T, Q) = 0$ significa que ningún elemento de Q es dominado por T. Como el operador de dominación no es un operador simétrico, $M_1(T, Q)$ no es necesariamente igual a $1 - M_1(Q, T)$. Por lo que es necesario calcular tanto $M_1(T, Q)$ como $M_1(Q, T)$ para saber cuántas soluciones de T están cubiertas por Q y viceversa. Es interesante observar en la ecuación anterior que la cardinalidad de los conjuntos de soluciones T y Q no necesita ser la misma.

Esta métrica puede ser usada para comparar el desempeño de dos algoritmos, pero también puede ser usada para evaluar el desempeño de un algoritmo, haciendo $T = P^*$, donde P^* es el conjunto óptimo de Pareto. La métrica $M_1(P^*, Q)$ determinará la porción de soluciones en Q, que son dominadas por elementos de P^*. Esta métrica provee una buena estimación de la convergencia si un conjunto grande de P^* es elegido. Para los conjuntos P^* y Q que se muestran en la Figura 3.12, $M_1(P^*, Q) = 2/5 = 0.4$, pues dos soluciones (B y E) de Q son dominadas por soluciones en P^*. $M_1(Q, P^*)$ siempre es cero.

3.4.2. Métrica de extensión

Deb et al. sugirieron la siguiente métrica

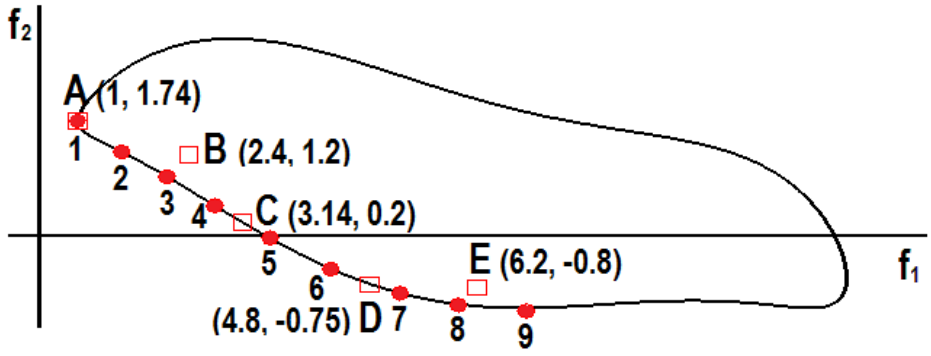

Figura 3.12: Métrica M_1. Las soluciones no dominadas del conjunto Q se muestran como cuadrados, y las soluciones elegidas del conjunto óptimo de Pareto P^* se muestran como círculos rellenos.

$$M_2(Q, P^*) = \frac{\sum_{i=1}^{a} d_i^e + \sum_{i=1}^{|Q|} \left| d_i - \overline{d} \right|}{\sum_{i=1}^{a} d_i^e + |Q| \, \overline{d}}$$

donde d_i puede ser cualquier distancia entre soluciones vecinas y $\overline{d}$ es el valor promedio de estas distancias [Deb01, Deb02]. La distancia euclidiana o la suma de las diferencias absolutas en los valores de las funciones objetivo pueden ser usadas para calcular d_i. El parámetro d_i^e es la distancia entre las soluciones extremas de Q y P^* correspondiente a la $i - ésima$ función objetivo.

Para un problema con dos objetivos, los parámetros d_1^e y d_2^e se muestran en la Figura 3.13. Cada d_i puede tomarse como la distancia euclidiana entre las soluciones consecutivas $i - ésima$ e $(i+1) - ésima$. Así, el término $|Q|$ en la ecuación anterior puede ser reemplazado por $(|Q| - 1)$. La métrica toma un valor cero para una distribución ideal, sólo cuando $d_i^e = 0$ y todos los valores d_i^e son iguales a $\overline{d}$. El primer término en el numerador de la métrica significa que en el conjunto de soluciones no dominadas obtenido el verdadero extremo de las soluciones óptimas de Pareto existe. El segundo término en el numerador de la métrica significa que la distribución de las soluciones intermedias es uniforme. Tal conjunto es un resultado ideal de cualquier algoritmo multiobjetivo. Así, para una distribución ideal de solu-

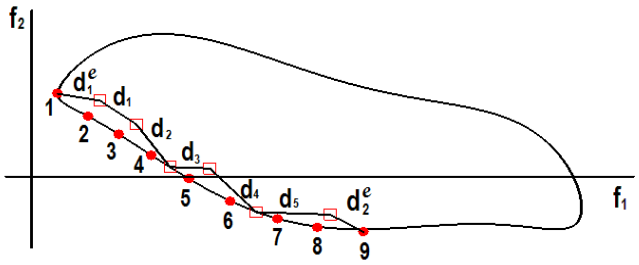

Figura 3.13: Métrica M_2. Las soluciones no dominadas del conjunto Q se muestran como cuadrados, y las soluciones elegidas del conjunto óptimo de Pareto P^* se muestran como círculos rellenos. Las soluciones 1 y 9 son soluciones extremas.

ciones $M_2(Q, P^*) = 0$. Considere otro caso, donde la distribución de soluciones no dominadas obtenida es uniforme, pero están agrupadas en un solo lugar. Tal distribución hará cero a todos los valores $\left|d_i - \overline{d}\right|$, pero originará valores distintos de cero para d_i^e. La métrica correspondiente se convierte en $M_2(Q, P^*) = \sum_{i=1}^{a} d_i^e / \left(\sum_{i=1}^{a} d_i^e + (|Q| - 1)\,\overline{d}\right)$. Esta cantidad se encuentra dentro del intervalo $[0, 1]$. Ya que el denominador mide la longitud de la aproximación por partes de la frontera de Pareto, el valor $M_2(Q, P^*)$ aumenta con d_i^e. Así, a medida que las soluciones se agrupan cada vez más cerca de la distribución ideal, el valor $M_2(Q, P^*)$ aumenta de cero a uno. Para una distribución no uniforme de las soluciones no dominadas, el segundo término en el numerador es diferente de cero y hace al valor $M_2(Q, P^*)$ más grande que con una distribución uniforme. Así, para malas distribuciones el valor de $M_2(Q, P^*)$ puede ser mayor a uno.

Para los conjuntos P^* y Q que se muestran en la Figura 3.12 las soluciones extremas A y 1 de Q y P^*, respectivamente, son iguales, por lo que $d_1^e = 0$. Para calcular las distancias se usó la distancia euclidiana. Como la solución 9 no está en Q, se calculó d_2^e =0.74, y los valores de las d_i

$$\begin{array}{ll} d_1 = 1.50 & d_2 = 1.24 \\ d_3 = 1.91 & d_4 = 1.40 \end{array}$$

El promedio de los valores de las d_i es $\overline{d}$ =1.51. Finalmente se calcula la métrica de extensión.

$$\begin{aligned} M_2(Q, P^*) &= \frac{0 + 0.74 + |1.5 - 1.51| + |1.24 - 1.51| + |1.91 - 1.51| + |1.4 - 1.51|}{0 + 0.74 + 4 \times 1.51} \\ &= 0.23 \end{aligned}$$

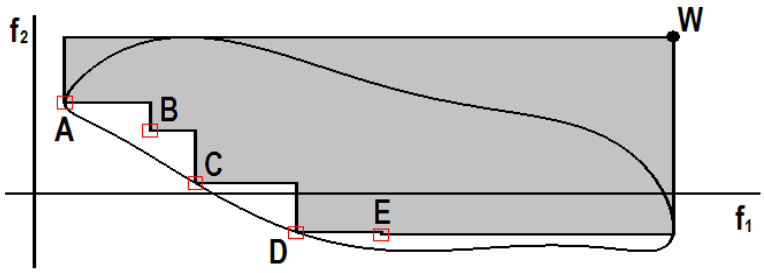

Figura 3.14: Hipervolumen encerrado por las soluciones no dominadas.

Como este valor es cercano a cero la distribución no es mala. Un algoritmo que encuentre un valor más pequeño de $M_2(Q, P^*)$ será capaz de encontrar un mejor conjunto diverso de soluciones no dominadas.

3.4.3. Métrica hipervolumen

Esta métrica calcula el volumen (en el espacio objetivo) cubierto por miembros de Q (región gris en la Figura 3.14) para problemas donde todos los objetivos son de minimización [Zit98]. Para cada $i \in Q$, se construye un hipercubo v_i con un punto de referencia W y la solución i como la esquina diagonal del hipercubo. El punto de referencia puede encontrarse construyendo un vector con los peores valores de la función objetivo, $W = (f_1^{\text{máx}}, f_2^{\text{máx}})$. Después se encuentra la unión de todos los hipercubos y calcula su hipervolumen

$$M_3(Q, W) = volumen(\cup_{i=1}^{|Q|} v_i)$$

La Figura 3.14 muestra el punto de referencia elegido W. El hipervolumen se muestra como la región en gris. Se desea un algoritmo con un valor grande de $M_3(Q, W)$. Para el ejemplo del problema que se muestra en la Figura 3.12, el hipervolumen $M_3(Q, W)$ se calculó con $W = (11, 3)$.

$$\begin{aligned} M_3(Q,W) &= (11-6.2)\times(3-(-0.8)) + (6.2-4.8)\times(3-(-0.75)) \\ &\quad +(4.8-3.14)\times(3-0.2) + (3.14-2.4)\times(3-1.2) \\ &\quad +(2.4-1)\times(3-1.74) \\ &= 31.23 \end{aligned}$$

La escala de los objetivos afecta a esta métrica. Por ejemplo, si la primera función objetivo toma valores de un orden de magnitud mayor que la segunda función objetivo, una unidad de mejora en f_1 reduciría $M_3(Q, W)$ mucho más

que una unidad de mejora en f_2. Para eliminar esta dificultad, la métrica antes descrita puede ser evaluada usando valores normalizados de la función objetivo.

Otra forma de eliminar este sesgo y poder calcular un valor normalizado de esta métrica es usar la métrica [Van99]

$$M_4(Q, P^*) = \frac{M_3(Q, W)}{M_3(P^*, W)}$$

Para un problema donde todos los objetivos son de minimización, el mejor valor de $M_4(Q, P^*)$ es uno (cuando $Q = P^*$). La Tabla 3.1 muestra los valores de las funciones objetivo f_1 y f_2 en las soluciones óptimas de Pareto que se muestran en la Figura 3.12.

Tabla 3.1
Valores de f_1 y f_2 de las soluciones de la Figura 3.12.

Solución	f_1	f_2	Solución	f_1	f_2
1	1.00	1.74	6	4.30	-0.53
2	1.58	1.26	7	5.20	-0.90
3	2.17	0.88	8	5.96	-1.08
4	2.79	0.44	9	6.84	-1.17
5	3.51	-0.05			

Así $M_3(P^*, W)$ =34.40 y $M_4(Q, P^*)$ =31.23/34.40=0.91. Como este valor es cercano a uno, el conjunto obtenido está cerca del conjunto óptimo de Pareto. Claramente, los valores de las métricas M_3 y M_4 dependen del punto de referencia elegido, W.

3.5. Resumen

Este capítulo presenta los principios de la optimización multiobjetivo. En un problema de optimización multiobjetivo, no se puede elegir una solución que es óptima con respecto a un único objetivo, cuando el resto de los objetivos también son importantes. Un algoritmo de optimización multiobjetivo debe encontrar soluciones tan cerca de las soluciones óptimas de Pareto como sea posible, y tan diversas como sea posible en el frente no dominado obtenido.

Proporciona algunas definiciones útiles, como el concepto de dominación, y muestra cómo usar tres métricas de desempeño: cobertura, extensión e hipervolumen. La métrica hipervolumen usa dos tipos de métricas que miden el progreso hacia el frente óptimo de Pareto y la diversidad entre las soluciones obtenidas de manera implícita.

Capítulo 4

Algoritmo de Búsqueda Dispersa para el PCTFR

4.1. Introducción

La BD es un método [Mar06] que ha sido aplicado en la resolución de un gran número de problemas de optimización. La primera descripción del método fue publicada por Fred Glover [Glo77].

Una descripción general del método de BD es la siguiente: se crea cada solución por medio de un método que genera diversidad entre soluciones, así se construye un conjunto de soluciones iniciales D, del que se extrae un conjunto de referencia E que contiene b soluciones (normalmente b es pequeño, por ejemplo, no mayor a 20 [Mar06]). El número de elementos de D generalmente es al menos diez veces el número de elementos del conjunto de referencia E [Mar06]. El conjunto de referencia inicial E contiene las $b/2$ mejores soluciones del conjunto D. Las $b/2$ restantes se extraen de D por el criterio de máxima distancia con las ya incluidas en el conjunto de referencia E. Para ello se emplea una función de distancia, la cual mide la distancia entre las soluciones. Como la BD se basa en combinar las soluciones del conjunto de referencia E, se consideran subconjuntos de 2 o más soluciones del conjunto de referencia E y se combinan mediante una rutina diseñada para tal efecto. La solución o soluciones que se obtienen de esta combinación pueden ser inmediatamente introducidas en el conjunto de referencia E (actualización dinámica) o almacenadas temporalmente en una lista hasta terminar de realizar todas las combinaciones y después determinar qué soluciones entran en éste (actualización estática). Las soluciones combinadas pueden entrar en el conjunto de referencia E y reemplazar a

algunas de las ya incluidas si las mejoran. Así, el conjunto de referencia E mantiene un tamaño b constante, pero va mejorando a lo largo de la búsqueda. Asimismo, el método de BD contiene un método de mejora, esto es, un método de búsqueda local para mejorar las soluciones que genera, tanto las del conjunto D, como las soluciones combinadas. El proceso de combinación se repite hasta que se satisface un criterio de terminación.

Así, para desarrollar un algoritmo de BD se requiere:

1. Una rutina generadora de soluciones diversas.
2. Un método para mejorar las soluciones generadas.
3. Un método para actualizar el conjunto de referencia.
4. Un método para generar subconjuntos de soluciones.
5. Un método para combinar soluciones.

Se empleó BD tanto en el algoritmo que resuelve el problema con un único objetivo, como en el algoritmo biobjetivo

4.2. Rutina generadora de soluciones diversas

Para generar diversificación se elaboraron procedimientos para generar tanto una solución aleatoria, como una solución diversa.

En todos los procedimientos que generan una solución, las rutas de cada solución son numeradas desde 1 hasta el número de rutas de la solución correspondiente.

Solución aleatoria. Para generar una ruta se elige aleatoriamente un cliente de entrega y se elige aleatoriamente un vehículo, con capacidad mayor o igual a la demanda del cliente. Se continúa agregando clientes de entrega elegidos aleatoriamente a la ruta mientras no se rebasa la capacidad del vehículo.

En las rutas impares si con un cliente de entrega se rebasa la capacidad del vehículo la ruta termina en el cliente anterior a ese.

En las rutas pares, si con algún cliente elegido aleatoriamente se rebasa la capacidad del vehículo, se determina la capacidad restante en el vehículo (si la hay), se determina qué clientes pueden ser atendidos por el vehículo (si los hay) y se elige aleatoriamente alguno de ellos, hasta que no haya capacidad restante en el vehículo, no haya clientes cuya demanda pueda ser

satisfecha con la capacidad restante en el vehículo, o no haya clientes por asignar.

Después de cualquiera de los dos casos anteriores, se agregan los clientes de recolección: se elige aleatoriamente un cliente de recolección y se agrega a la ruta si la oferta del cliente no rebasa la capacidad del vehículo. Como antes, se continúa agregando clientes de recolección mientras no se rebasa la capacidad del vehículo y hay clientes de recolección por asignar. Si con algún cliente de recolección elegido aleatoriamente se rebasa la capacidad del vehículo, la ruta termina en el cliente anterior a ese. Al terminar de generar una ruta, se ajusta la elección del vehículo: se elige como vehículo que va a atender a la ruta al más pequeño que puede servir tanto a los clientes de entrega como a los de recolección.

Cada ruta se obtiene repitiendo el procedimiento anterior hasta que todos los clientes de entrega y recolección son asignados a una ruta, respetando las restricciones del problema. Después de generar una solución, se le aplica el procedimiento 2-opt (que se describe más adelante). La solución se puede representar con una gráfica dirigida $G_1 = (V, A_1)$ donde $V = \{0\} \cup L \cup B$, $L = \{1, \ldots, n\}$, $B = \{n+1, \ldots, n+m\}$ y A_1 es el conjunto de arcos que pertenecen a la solución (ver un ejemplo de G_1 en la Figura 4.2). La matriz de adyacencia de G_1 es MS (ver un ejemplo de MS en la Figura 4.3) de $(n+m+1) \times (n+m+1)$ de componentes:

$$ms_{ij} = \begin{cases} 1 \text{ si } (i,j) \in A_1 \\ 0 \text{ si } (i,j) \notin A_1 \end{cases}$$

Se genera una solución diversa construyendo las rutas de ésta con los clientes que se han empleado con menor frecuencia (ver ejemplo de la matriz de frecuencias en la Figura 4.1). Para ello se crea una matriz de frecuencias F de $(n+m+1) \times (n+m+1)$ la cual se actualiza en cada iteración del algoritmo. Cada componente f_{ij} de la matriz de frecuencias indica el número de veces que el cliente i ha precedido al cliente j. Cada vez que se genera una solución aleatoria, una solución diversa o una solución combinada se actualiza la matriz de frecuencias F, haciendo $F = F + MS$ [Cun97], donde MS es la matriz de adyacencia correspondiente a la solución aleatoria, a la solución diversa o a la solución combinada.

Solución diversa. Se elige aleatoriamente de entre los clientes que se han asignado con menor frecuencia usando la matriz de frecuencias (ver un ejemplo de la matriz de frecuencias en la Figura 4.1). La forma en cómo se eligen los clientes es lo único que difiere entre la solución diversa y la solución aleatoria. Después de generar una solución se le aplica el procedimiento 2-

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
0	0	102	78	98	101	69	104	101	81	93	93	0	0	0	0	0	0	0	0	0	0	920
1	38	0	7	12	14	23	10	7	13	5	16	1	6	8	7	6	8	8	1	4	6	200
2	20	8	0	13	9	14	15	15	13	21	7	5	3	4	9	9	4	11	8	9	3	200
3	27	6	16	0	18	14	10	11	11	12	11	7	9	8	8	6	6	4	3	7	6	200
4	40	9	10	13	0	11	5	10	10	16	13	0	8	12	8	5	7	4	2	7	10	200
5	25	14	9	8	7	0	20	9	29	10	15	1	8	7	4	8	8	3	8	5	2	200
6	33	11	11	9	3	16	0	15	10	14	11	7	13	4	3	10	10	10	3	2	5	200
7	30	10	16	15	10	7	9	0	9	11	8	12	4	7	7	5	8	8	11	5	8	200
8	20	16	17	10	13	26	6	5	0	7	17	9	6	8	8	6	2	4	8	5	7	200
9	36	11	26	16	10	9	13	19	8	0	9	6	3	9	4	3	7	2	2	2	5	200
10	41	13	10	6	15	11	8	8	16	11	0	4	7	8	4	5	3	10	8	5	7	200
11	64	0	0	0	0	0	0	0	0	0	0	0	17	18	22	9	25	11	14	11	9	200
12	66	0	0	0	0	0	0	0	0	0	0	30	0	9	19	13	21	8	16	13	5	200
13	60	0	0	0	0	0	0	0	0	0	0	7	19	0	16	14	17	14	15	22	16	200
14	75	0	0	0	0	0	0	0	0	0	0	28	24	9	0	17	11	10	9	7	10	200
15	57	0	0	0	0	0	0	0	0	0	0	25	17	10	18	0	17	13	17	18	8	200
16	62	0	0	0	0	0	0	0	0	0	0	13	21	18	8	31	0	9	12	14	12	200
17	78	0	0	0	0	0	0	0	0	0	0	12	11	16	13	8	9	0	18	16	19	200
18	51	0	0	0	0	0	0	0	0	0	0	11	10	13	13	20	12	16	0	24	30	200
19	52	0	0	0	0	0	0	0	0	0	0	10	5	12	14	13	17	23	22	0	32	200
20	45	0	0	0	0	0	0	0	0	0	0	12	9	20	15	12	8	32	23	24	0	200
	920	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	

Figura 4.1: Ejemplo de una matriz de frecuencias con $n = 10$ y $m = 10$.

opt, se convierte la solución en la matriz de adyacencia que le corresponde y se suma ésta a la matriz de frecuencias.

Se eligió como tamaño del conjunto de referencia b el tamaño del caso a resolver. Se crea un conjunto de soluciones iniciales D del cual se extrae el conjunto de referencia. El conjunto D consta de $10b$ soluciones.

Para determinar qué porcentaje de soluciones diversas en el conjunto D era más adecuado, se probó experimentalmente con 20 %, 33 %, 40 %, 50 % y 100 %. En cada caso, se eligió como porcentaje a emplear aquel con el cual se obtuvieron mejores resultados. La Tabla 4.1 muestra, dependiendo del tamaño del caso, el porcentaje de soluciones diversas que se emplea:

Tabla 4.1
Porcentaje de soluciones diversas en el conjunto D.

Tamaño del caso $n+m$	Porcentaje de soluciones diversas en el conjunto D
$n+m \leq 25$	33 %
$25 < n+m \leq 75$	100 %
$n+m > 75$	50 %

Si el tamaño del caso es menor a 26 por cada dos iteraciones en que se genera una solución aleatoria, hay una en la que se genera una solución diversa. Si el tamaño del caso está entre 26 y 75 la primera solución que se

genera es aleatoria y todas las demás soluciones a generar deben ser soluciones diversas. Si el tamaño del caso es mayor a 75 se genera una solución aleatoria, por cada solución diversa.

Cada solución que se genera se introduce en el conjunto de soluciones D si es diferente a las ya contenidas. Si es igual, se vuelve a generar la solución aleatoria o diversa, según corresponda, hasta que el conjunto D contenga $10b$ soluciones distintas.

Para diversificar las soluciones del conjunto de referencia se emplea el siguiente procedimiento.

Procedimiento de diversificación. Se emplea una lista para guardar las soluciones del conjunto de referencia, en la cual las soluciones están ordenadas de menor a mayor respecto al costo de la solución, el cual está constituido por el costo del vehículo más un costo unitario por unidad de distancia recorrida en cada ruta de la solución. Sean $E_1, E_2, \ldots, E_u$ los primeros u elementos de la lista de soluciones antes mencionada, donde u es igual a n más la parte entra de $m/2$. Se genera una solución diversa S, con el procedimiento antes descrito. Sean ME_i y MS las matrices de adyacencia correspondientes a las soluciones E_i y S, y sea $MT_i = ME_i + MS$. Para producir una nueva solución se emplea el método para combinar soluciones (el cual se explica más adelante) usando la matriz MT_i, en lugar de la matriz MT del método para generar subconjuntos de soluciones, que se describe en breve. Esto se hace para i desde 1 hasta u.

4.3. Método de mejora

Como método para mejorar las soluciones generadas se implementó 2-opt. Este método de búsqueda local propuesto por Croes [Cro58] para resolver el problema del agente viajero, consiste en eliminar dos arcos (t_i, t_{i+1}) y (t_j, t_{j+1}) de una ruta dada $(t_1, \ldots t_i, t_{i+1}, \ldots, t_j, t_{j+1}, \ldots, t_S)$ y reconectar la ruta en el otro orden posible, es decir, invertir el orden de la subruta $(t_{i+1}, \ldots, t_j)$, la nueva ruta generada es $(t_1, \ldots t_i, t_j, \ldots, t_{i+1}, t_{j+1}, \ldots, t_S)$.

4.4. Construcción del conjunto de referencia

Sea E el conjunto de referencia. La mitad de los elementos de E se forma con las soluciones de menor costo del conjunto de soluciones iniciales D, la otra mitad con los elementos de D más distantes [Mar06, Sección 2] a los ya incluidos en E.

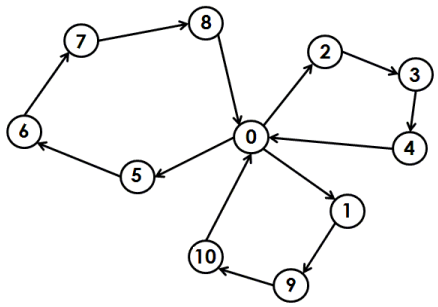

Figura 4.2: Solución factible del PRVR con $L = \{1, ..., 8\}$, $B = \{9, 10\}$

Distancia entre soluciones. Una solución factible al problema de rutas de vehículos con recolecciones es un conjunto de rutas que satisfacen las restricciones, la cual se puede representar con una gráfica $G_1 = (V, A_1)$ donde $V = \{0\} \cup L \cup B$, $L = \{1, \ldots, n\}$, $B = \{n+1, \ldots n+m\}$ y A_1 es el conjunto de arcos que pertenecen a la solución. A su vez, la gráfica correspondiente a una solución S se puede representar por medio de la matriz de adyacencia respectiva MS de $(n+m+1) \times (n+m+1)$, como se vio antes.

Caso ejemplo, suponga que $L = \{1, \ldots, 8\}$ son los clientes de entrega y $B = \{9, 10\}$ son los clientes de recolección o proveedores y que la Figura 4.2 es una solución factible de este caso del problema de rutas de vehículos con recolecciones.

La Figura 4.3 muestra la matriz de adyacencia correspondiente a la gráfica de la Figura 4.2.

Sea T otra solución factible del problema de rutas de vehículos con recolecciones y sea MT su matriz de adyacencia. Como distancia entre las soluciones S y T se emplea.

$$d(S,T) = \sum_{i=0}^{n+m} \sum_{j=0}^{n+m} (ms_{ij} - mt_{ij})^2$$

La máxima distancia entre las soluciones S y T es $2(n+m)+r_S+r_T$ donde r_S y r_T son el número de rutas de las soluciones S y T respectivamente.

$$MS = \begin{array}{c} \begin{matrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{matrix} \\ \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{matrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \end{matrix} \end{array}$$

Figura 4.3: Matriz de adyacencia correspondiente a la gráfica de la Figura 4.2.

4.5. Método para generar subconjuntos de soluciones

Se elige aleatoriamente un número r entre 2 y 5 [Mar06, Sección 3.5], en donde r es el número de soluciones que se combinarán. Aleatoriamente se eligen r soluciones distintas del conjunto de referencia E, se convierte cada una de ellas en la matriz de adyacencia que le corresponde, se suman las matrices de adyacencia de las r soluciones y se guardan en la matriz MT.

4.6. Método para combinar soluciones

Sean $L = \{1, \ldots, n\}$ y $B = \{n+1, \ldots, n+m\}$ los subconjuntos correspondientes a los clientes de entrega y de recolección, respectivamente. Se llama al método para generar subconjuntos, se emplea la matriz MT que se crea en éste. De entre las entradas 1 a n del renglón cero de la matriz MT (ver un ejemplo de la matriz MT en la Figura 4.4), con mayor valor se elige aleatoriamente un cliente, cli, $L = L - \{cli\}$, después se elige aleatoriamente un vehículo, con capacidad mayor o igual a la demanda del cliente cli. Luego de entre las entradas 1 a n del renglón cli de la matriz MT con mayor valor se elige aleatoriamente un cliente que no haya sido

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
0	0	0	0	0	1	1	2	2	0	1	0	0	0	0	0	0	0	0	0	0	0	7
1	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	2
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	2
3	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
4	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
5	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	2
6	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	2
7	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
8	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	2
9	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
10	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	2
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	2
12	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	2
13	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
14	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	2
15	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	2
16	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	2
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	2
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	2
19	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	2
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	2
	7	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	

Figura 4.4: Ejemplo de una matriz MT con $n = 10$ y $m = 10$.

asignado y cuya demanda no exceda la capacidad del vehículo. De esta manera se continúa agregando clientes a la ruta. Para construir la solución combinada se procede como en la solución aleatoria. Para agregar los clientes de recolección a la ruta, sea $cliA$ el cliente anterior en la ruta. De entre las entradas $n + 1$ a $n + m$ del renglón $cliA$ de la matriz MT, con mayor valor se elige aleatoriamente un cliente de recolección cli y se agrega a la ruta, si la oferta del cliente no rebasa la capacidad del vehículo $B = B - \{cli\}$, de igual modo se continúa agregando clientes de recolección a la ruta mientras no se rebasa la capacidad del vehículo y haya clientes de recolección por asignar. Al terminar de construir una ruta se ajusta la elección del vehículo. Cuando se termina de construir una solución se le aplica el procedimiento 2-opt, se convierte la solución en la matriz de adyacencia que le corresponde y se suma ésta a la matriz de frecuencias.

La idea de cómo construir las soluciones combinadas es análoga a la propuesta por Cung et al. [Cun97].

4.7. Método para actualizar el conjunto de referencia

La Tabla 4.2 muestra el número de iteraciones que realiza el algoritmo de BD.

Tabla 4.2
Número de iteraciones del algoritmo de BD.

Tamaño del caso $n+m$	Iteraciones a realizar
$n+m \leq 25$	25000
$25 < n+m \leq 50$	50000
$50 < n+m \leq 75$	185000
$n+m > 75$	225000

Se realiza una iteración j del procedimiento de diversificación si:

- j es múltiplo de diez y $(n+m) < 75$, o
- j es múltiplo de diez y $(n+m) \geq 75$ y $j \leq 5000$

En otro caso, se realiza una iteración combinando soluciones. La frecuencia con que se combinan soluciones y se emplea el procedimiento de diversificación parecen ser adecuadas [Cun97]. Las soluciones generadas en cada iteración entran al conjunto de referencia si son diferentes de los elementos en él y su costo es menor al del último elemento del conjunto de referencia.

4.8. Intersección entre segmentos

Los problemas que se van a resolver son euclidianos, pues cada vértice j está localizado geográficamente en (x_j, y_j). Se sabe que una ruta sin cruces es de mejor calidad (menor costo) que una en la que hay cruces [Seg11], por lo que se desarrolló e implementó un algoritmo [Cor01] que detecta si hay alguna intersección en un conjunto de n segmentos de recta. El Algoritmo 4.1 muestra el pseudocódigo presentado por Cormen et al. [Cor01].

Algoritmo 4.1: Presentado por Cormen et al.

$T = \varnothing$
Ordene los puntos extremos de los segmentos en S de izquierda a derecha
Para cada punto p en la lista ordenada de puntos extremos
    Si p es el extremo izquierdo de un segmento s entonces
        `Inserta(`T`, `s`)`
        Si (`Arriba(`T`, `s`)` existe e interseca a s) o
            (`Abajo(`T`, `s`)` existe e interseca a s)
          entonces devuelva VERDADERO
    Si p es el extremo derecho de un segmento s entonces
        Si ambos `Arriba(`T`, `s`)` y `Abajo(`T`, `s`)` existen y
            `Arriba(`T`, `s`)` interseca a `Abajo(`T`, `s`)`
          entonces devuelva VERDADERO
        `Elimina(`T`, `s`)`
Devuelva FALSO

El algoritmo anterior recibe como entrada un conjunto de segmentos S, devuelve verdadero si los segmentos se intersectan, en caso contrario devuelve falso. Además

`Inserta(`T`, `s`)` inserta al segmento s en el conjunto T.

`Elimina(`T`, `s`)` elimina al segmento s en el conjunto T.

`Arriba(`T`, `s`)` devuelve el segmento inmediatamente arriba del segmento s en T.

`Abajo(`T`, `s`)` devuelve el segmento inmediatamente abajo del segmento s en T.

4.9. Resultados experimentales

El algoritmo de BD se implementó en Delphi 5 en una computadora personal con procesador Intel Core i5 a 2.53 GHz y 4 GB de memoria RAM, cada caso se resolvió 30 veces.

4.9.1. Casos de la tercera clase del PCTF

Se resolvieron los casos de prueba de la tercera clase, esto es, se resolvió el PCTF propuesto por Golden et al. [Gol84], el cual puede verse como

un caso particular del PCTFR en el que no hay clientes de recolección, es decir, cuando $B = \varnothing$. Empleando en el conjunto de soluciones iniciales, D, el porcentaje de soluciones diversas establecido en la Sección 4.2 se obtuvo una desviación respecto de las mejores soluciones conocidas del 4.12 %. Para reducir la desviación se determinó el porcentaje de soluciones diversas más adecuado para estos casos. La Tabla 4.3 muestra el porcentaje de soluciones diversas en el conjunto D a emplear en cada caso:

Tabla 4.3
Porcentaje de soluciones diversas en el conjunto D.

Caso	Porcentaje de soluciones diversas en el conjunto D
18	20 %
5,15	33 %
20	50 %
3,4,6,13,14,16,17,19	100 %

Asimismo, se observó que el número de rutas de las soluciones conocidas es mayor al obtenido por el algoritmo propuesto. Incluir en las rutas pares (de la solución aleatoria, la solución diversa y la solución combinada) todos los clientes posibles reduce el número de rutas que se generan. Para mejorar los resultados del algoritmo en estos casos se decidió no incluir a todos los clientes posibles en las rutas pares. La Tabla 4.4 muestra un resumen de los resultados obtenidos, después de realizar estas dos modificaciones en el algoritmo. Las primeras tres columnas muestran: el nombre del caso de prueba, la mejor solución conocida reportada en la literatura y el número de rutas de ésta. Las siguientes cinco columnas muestran resultados de BD: la mejor solución, el número de rutas que le corresponden, el promedio del mejor resultado de cada una de las 30 repeticiones, la desviación estándar de la mejor solución de las 30 repeticiones del algoritmo y el porcentaje de la diferencia entre la mejor solución de BD y la mejor solución conocida.

BD fue capaz de obtener la solución óptima de los cuatro casos con 20 clientes. La desviación promedio de BD con respecto a las mejores soluciones conocidas es del 3.32 %. Es importante notar que la solución conocida de 11 de estos 12 casos de prueba es la solución óptima.

Tabla 4.4
Comparación de resultados para los casos de la tercera clase, propuestos por Golden et al. para el PCTF.

			BD				
Caso	Mejor costo conocido	No. de rutas	Costo mejor sol.	No. de rutas	Costo sol. prom.	Desv. est.	%
3	961.03[a]	6	961.03	6	964.73	4.58	0.00
4	6437.33[a]	6	6437.33	6	6503.08	134.02	0.00
5	1007.05[a]	5	1007.05	5	1013.13	4.41	0.00
6	6516.47[a]	6	6516.47	6	6588.42	105.61	0.00
13	2406.36[a]	10	2484.86	8	2548.43	24.77	3.26
14	9119.03[a]	8	9218.13	8	9327.06	47.52	1.09
15	2586.37[a]	13	2630.49	10	2698.48	25.36	1.71
16	2720.43[a]	9	2804.38	9	2857.64	25.58	3.09
17	1734.53[b]	8	1802.46	7	1846.21	19.06	3.92
18	2369.65[a]	16	2647.82	13	2736.98	31.50	11.74
19	8661.81[b]	15	9391.41	13	9646.01	74.18	8.42
20	4032.81	19	4299.67	12	4426.90	48.35	6.62
Promedio							3.32

[a] Pessoa et al. (2007) probaron que esta solución es óptima.
[b] Baldacci et al. (2009) probaron que esta solución es óptima.

En la Tabla 4.5, las cuatro primeras columnas muestran: el nombre del caso de prueba, el número de clientes, el estudio que primero obtuvo la mejor solución conocida y el tiempo de ejecución respectivo (si proporcionaron la información correspondiente al caso). Las últimas tres columnas muestran información de BD: el tiempo promedio de ejecución de las 30 repeticiones de BD, el total de iteraciones realizadas por el algoritmo y la iteración en que se obtuvo la mejor solución. Los autores de los estudios que primero obtuvieron la mejor solución conocida usaron diferentes formas de presentar sus resultados. Tanto Taillard, como Choi y Tcha proporcionaron el tiempo promedio de ejecución de cinco corridas. Gendreau et al. reportaron el tiempo de ejecución de la mejor solución. Brandão proporcionó el tiempo promedio de ejecución, aunque no específico cuantas corridas realizó de su algoritmo. Subramanian et al. reportaron el tiempo promedio de ejecución de 10 corridas. Es difícil comparar los tiempos de cómputo de los algoritmos mencionados por las diferentes formas en que fueron presentados los resultados y porque los tiempos de computo son relativos a los equipos en donde se ejecutaron.

Tabla 4.5
Comparación del tiempo de calculó para los casos de la tercera clase, propuestos por Golden et al. para el PCTF.

				BD		
Caso	n	Referencia	Tiempo (s)	Tiempo prom. (s)	Total de iteraciones	Iteración en que llegó a mejor sol.
3	20	Taillard (1999)	[a,b]	18.55	25000	13093
4	20	Taillard (1999)	[a,b]	18.37	25000	11052
5	20	Gendreau et al. (1999)	168[c]	18.59	25000	14378
6	20	Taillard (1999)	[a,b]	18.28	25000	18106
13	50	Choi y Tcha (2007)	8[d]	191.49	50000	39695
14	50	Taillard (1999)	570[a]	184.60	50000	43384
15	50	Taillard (1999)	334[a]	189.49	50000	44339
16	50	Choi y Tcha (2007)	7[d]	190.83	50000	37306
17	75	Brandão (2009)	322[e]	238.94	185000	141663
18	75	Brandão (2009)	267[e]	225.06	185000	133868
19	100	Taillard (1999)	12528[a]	409.65	225000	82465
20	100	Subramanian et al. (2012)	46[f]	415.07	225000	177307
Promedio				176.58		

[a] Sun Sparc a 50 MGz.
[b] El tiempo de ejecución de este caso fue de 20 a 40 segundos.
[c] Sun Sparc 10.
[d] Pentium IV 2.6 GHz.
[e] Pentium M 1.4 GHz.
[f] Intel Core i7 2.93 GHz.

4.9.2. Casos de la cuarta clase del PCTFR

Se resolvieron los casos de prueba de la cuarta clase, es decir, se resolvió la variante del PCTFR propuesta por Salhi et al. [Sal13]. Para determinar el porcentaje de soluciones diversas más adecuado para estos casos de prueba, se corrieron los casos con 20 % y 100 % de soluciones diversas.

La Tabla 4.6 presenta el resumen de los resultados en los casos de la cuarta clase. Las primeras tres columnas muestran: el nombre del caso de prueba, el número de clientes y la mejor solución conocida reportada en la literatura. Las siguientes cuatro columnas muestran: la mejor solución de Salhi et al. [Sal13], la mejor solución de Belloso et al. [Bel17], la mejor solución de BD con 20 % o 100 % de soluciones diversas (en el conjunto de

soluciones iniciales D) y la mejor solución de Penna et al. [Pen17]. El último renglón de la tabla muestra el porcentaje de desviación promedio de los diferentes algoritmos respecto a la mejor solución conocida.

BD fue capaz de encontrar soluciones que superan a las mejores obtenidas por los algoritmos de Salhi et al. y de Belloso et al. en ocho casos de prueba. Salhi et al. obtuvieron la solución exacta de 13 casos, con lo que se comprobó que la solución obtenida por BD en los casos HWS9 y HWS10 es óptima (ver Figura 4.5). La desviación de Salhi et al., Belloso et al., BD y Penna et al. de las mejores soluciones conocidas es de 2.34 %, 1.76 %, 3.85 % y 0.04 %, respectivamente.

Salhi et al. desarrollaron tres heurísticas para resolver los casos de prueba del PCTFR. La heurística 2 fue la que obtuvo mejores resultados, por lo que se usó ésta para comparar los tiempos de ejecución de los algoritmos.

Salhi et al. corrieron sus algoritmos en una computadora con un procesador Pentium 4 a 3 GHz. Belloso et al. usaron una computadora con un procesador Intel i7 a 2.8 GHz. Penna et al. ejecutaron su algoritmo en una computadora con un procesador Intel Core i7 a 2.93 GHz. El tiempo promedio de ejecución de Salhi et al., Belloso et al., BD y Penna et al. es de 709.76, 75.31, 196.32 y 5.06 segundos, respectivamente.

Tabla 4.6
Comparación de resultados para los casos de la cuarta clase, propuestos por Salhi et al. para el PCTFR.

		Mejor sol.	Mejor solución			
Caso	$n+m$	conocida	Salhi et al	Belloso et al.	BD	Penna et al.
HWS1	20	720.57	726.48	734.03	726.47	720.57
HWS2	20	818.12	818.12	820.99	831.78	818.12
HWS3	20	848.32	848.59	848.32	860.23	848.32
HWS4	20	4342.48	4350.65	4342.48	4350.65	4342.48
HWS5	20	5357.98	5366.39	5357.98	5367.26	5357.98
HWS6	20	5421.65	5875.23	5872.52	5447.81	5421.65
HWS7	20	729.50	767.93	729.50	730.84	729.50
HWS8	20	838.11	872.97	838.11	850.20	838.11
HWS9	20	890.76	903.18	890.76	890.76	890.76
HWS10	20	4349.13	4365.44	4349.13	4349.13	4349.13
HWS11	20	5363.58	5414.50	5363.58	5376.64	5363.58
HWS12	20	5497.98	5928.78	5523.50	5519.06	5497.98
HWS13	50	1590.47	1625.70	1632.60	1622.20	1590.47
HWS14	50	1771.53	1811.63	1793.26	1842.12	1771.53
HWS15	50	1999.05	2018.93	2030.37	2127.07	1999.05

Tabla 4.6 (continuación)

		Mejor sol.	Mejor solución			
Caso	$n+m$	conocida	Salhi et al	Belloso et al.	BD	Penna et al.
HWS16	50	5551.19	5561.67	5562.94	5561.08	5551.19
HWS17	50	6547.93	6570.39	6571.77	6583.72	6547.93
HWS18	50	7120.52	7599.08	7413.27	7181.21	7120.52
HWS19	50	1616.21	1704.41	1659.86	1620.29	1616.21
HWS20	50	2015.67	2037.23	2038.22	2057.92	2015.67
HWS21	50	2295.57	2340.09	2341.17	2363.65	2295.57
HWS22	50	1717.60	1774.71	1746.28	1739.74	1717.60
HWS23	50	2096.10	2166.52	2164.65	2187.70	2096.10
HWS24	50	2401.04	2430.88	2425.83	2496.19	2401.04
HWS25	75	1285.86	1332.02	1313.59	1307.87	1285.86
HWS26	75	1399.36	1421.04	1412.69	1436.60	1399.36
HWS27	75	1513.10	1534.65	1543.10	1581.23	1513.10
HWS28	75	1572.38	1617.85	1631.70	1736.59	1572.38
HWS29	75	1760.95	1799.76	1808.46	1986.37	1760.95
HWS30	75	1950.99	1990.46	2001.79	2226.88	1950.99
HWS31	100	4943.29	4943.29	4990.84	5385.57	4963.08
HWS32	100	5993.30	*6035.96	6019.82	6554.84	5993.30
HWS33	100	7097.81	*7601.09	7593.25	7803.70	7097.81
HWS34	100	2465.41	2465.41	2516.30	2728.89	2494.95
HWS35	100	2927.20	2971.98	2998.21	3176.83	2927.20
HWS36	100	3450.73	3533.90	3563.17	3764.87	3450.73
% desv. prom.			2.34	1.76	3.85	0.04

*En estos casos se consideró el tiempo de la heurística 3, la heurística 2 no dio solución porqué consumió demasiado tiempo de computadora.

4.10. Resumen

Este capítulo propone un algoritmo de BD para resolver el PCTFR. Para implementar el algoritmo se elaboraron procedimientos para generar tanto una solución aleatoria, como una solución diversa y un procedimiento de diversificación. Asimismo, se propuso una forma de calcular la distancia entre dos soluciones y métodos: para generar subconjuntos, para combinar soluciones, para actualizar el conjunto de referencia.

Se implementaron: un algoritmo que detecta si hay alguna intersección en un conjunto de segmentos dados [Cor01] y el método de mejora 2-opt.

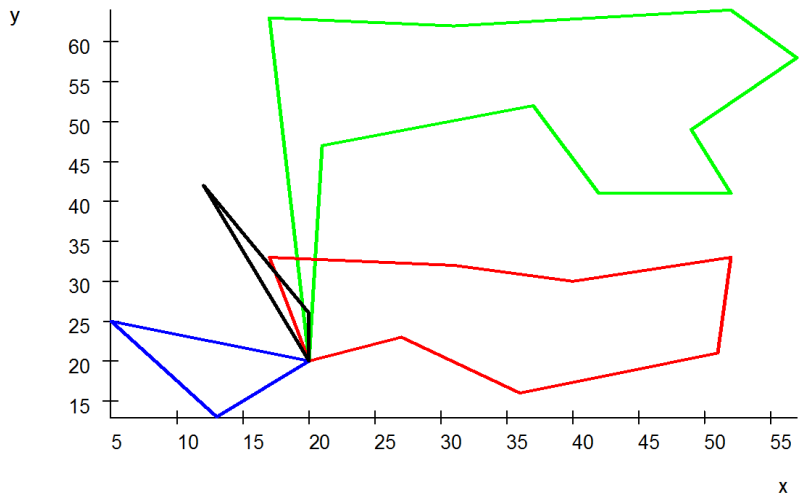

Figura 4.5: Gráfica de la solución óptima del caso HWS9.

BD se probó con los casos de prueba de las clases tercera y cuarta. Tanto en los casos de la tercera clase, como en los casos de la cuarta clase la desviación de BD es de más del 3 %, a pesar de ello, BD fue capaz de encontrar un total de 6 soluciones óptimas en los casos de las clases tercera y cuarta.

Capítulo 5

Algoritmo multiobjetivo

Este capítulo contiene la versión multiobjetivo del algoritmo, así como los procedimientos y métodos empleados en esta. El algoritmo multiobjetivo usa los procedimientos de clasificación no dominada y de asignación de distancia de amontonamiento propuestos por Deb et al. [Deb01, Deb02], los cuales se presentan en las Secciones 5.2 y 5.3, respectivamente. La Sección 5.6 muestra los resultados experimentales obtenidos en los casos de prueba de las clases tercera y cuarta.

El algoritmo de BD multiobjetivo (BDMO) emplea los siguientes métodos y procedimientos del algoritmo para el problema con un objetivo:

- Método de mejora,
- Método para generar subconjuntos,
- Método para combinar soluciones,
- Procedimiento para generar una solución diversa (en la Sección 4.2), y
- Procedimiento para generar una solución aleatoria (en la Sección 4.2).

5.1. Procedimiento de diversificación

Se crea una lista en la cual se guardan las soluciones del conjunto de referencia. En esta lista las soluciones siempre están ordenadas de menor a mayor con respecto al costo de la solución. (Se requiere que las soluciones estén ordenadas respecto a uno de los objetivos, podrían haberse ordenado respecto al número de rutas.) Sean $E_1, E_2, \ldots, E_u$ los primeros u elementos de la lista de soluciones antes mencionada.

- Si $n + m \leq 50$ entonces $u = 9$
- Si $n + m > 50$ entonces u es igual a n más la parte entra de $m/2$.

Se genera una solución diversa S, con el procedimiento descrito en la Sección 4.2. Sean ME_i y MS las matrices de adyacencia correspondientes a las soluciones E_i y S, y sea $MT_i = ME_i + MS$. Para producir una nueva solución se emplea el método para combinar soluciones usando la matriz MT_i, en lugar de la matriz MT que produce el método para generar subconjuntos. Esto se hace para i desde 1 hasta u.

Se modificó este procedimiento con respecto al empleado en el algoritmo para el problema con un objetivo, para que el número de evaluaciones de la función objetivo que realiza el algoritmo fuese acorde al tamaño del caso a resolver.

5.2. Clasificación de la población

La mayoría de los algoritmos evolutivos de optimización multiobjetivo sólo requieren encontrar el mejor frente no dominado en una población. Estos algoritmos clasifican a la población en dos conjuntos: el conjunto no dominado y el conjunto dominado restante. Sin embargo, existen algoritmos que requieren que la población entera sea clasificada en varios niveles no dominados. En tales algoritmos, la población necesita ser clasificada en niveles ascendentes no dominados. Las mejores soluciones no dominadas son llamadas soluciones no dominadas de nivel 1. Las soluciones no dominadas de la población restante son entonces encontradas y denominadas soluciones no dominadas de nivel 2. Este procedimiento continúa hasta que todos los miembros de la población son clasificados en un nivel no dominado. Es importante reiterar que las soluciones no dominadas de nivel 1 son mejores que las soluciones no dominadas de nivel 2, etcétera.

Para cada solución se calcula: (i) n_i (el contador de dominación), el número de soluciones que dominan a la solución i, y (ii) S_i, el conjunto de soluciones que la solución i domina. Esto requiere $O(aN^2)$ comparaciones, donde a es el número de objetivos y N es el tamaño de la población. Al terminar este procedimiento, todas las soluciones en el primer frente no dominado tendrán como contador de dominación a cero. Luego, para cada una de estas soluciones (cada solución i con $n_i = 0$), se revisa cada miembro j de su conjunto S_i y se reduce su contador de dominación en uno. Al hacerlo, si para cualquier miembro j el contador de dominación se vuelve cero, dicho miembro se pone en una lista separada P_1. Después de que las modificaciones en S_i son realizadas para cada i con $n_i = 0$, todas las soluciones de P_1 pertenecerán al segundo frente no dominado. El procedimiento anterior puede continuar con cada miembro de P_1 y el tercer frente no dominado

puede ser identificado. Este proceso continua hasta que todas las soluciones son clasificadas. El Algoritmo 5.1 es el algoritmo correspondiente, el cual fue propuesto por Deb et al. [Deb02, Deb01].

Algoritmo 5.1: de clasificación no dominada

Paso 1. Para cada $i \in P$, $n_i = 0$, $S_i = \varnothing$. Para todo $j \neq i$, $j \in P$ realice el Paso 2 y luego el Paso 3.
Paso 2. Si i domina a j, hacer $S_p = S_p \cup \{j\}$. Si j domina a i, hacer $n_i = n_i + 1$.
Paso 3. Si $n_i = 0$, guarde a i en el primer frente no dominado P_1. Haga al contador de frontera $k = 1$.
Paso 4. Mientras $P_k \neq \varnothing$, realice los siguientes pasos.
Paso 5. Hacer $Q = \varnothing$ para almacenar las siguientes soluciones no dominadas. Para cada $i \in P_k$ y para cada $j \in S_i$,
- Hacer $n_j = n_j - 1$.
- Si $n_j = 0$, hacer $Q = Q \cup \{j\}$.
Paso 6. Hacer $k = k + 1$ y $P_k = Q$. Ir al paso 4.

Los Pasos 1 a 3 encuentran las soluciones en el primer frente no dominado, la complejidad computacional de estos es $O(aN^2)$. Los Pasos 4 a 6 repetidamente encuentran frentes mayores y requieren a lo más $O(N^2)$ comparaciones, como se verá a continuación.

Para cada i en el segundo o posteriores niveles de dominación, el contador de dominación n_i puede a lo más ser $N - 1$. Así, cada solución i será examinada a lo más $N - 1$ veces antes de que su contador de dominación se vuelva cero. En este punto, la solución es asignada a un nivel no dominado particular y nunca será examinada otra vez. Como hay a lo más $N - 1$ soluciones como esta, la complejidad de identificar el segundo frente y los que le siguen es $O(N^2)$. Así, la complejidad total del procedimiento es $O(aN^2)$. Es importante notar que aunque la complejidad computacional del Algoritmo 5.1 es de $O(aN^2)$, los requerimientos de almacenamiento son de $O(N^2)$.

5.3. Distancia de amontonamiento

El Algoritmo 5.2 [Deb02, Deb01] se usa para calcular la distancia de amontonamiento de cada punto en el conjunto F.

Algoritmo 5.2: Asignación de distancia de amontonamiento

Paso 1. Llame $l = |F|$ al número de soluciones en F. Para cada $i \in F$ haga $d_i = 0$.

Paso 2. Para cada función objetivo $m = 1, 2, \ldots, a$, ordene los elementos del conjunto de menor a mayor con respecto al objetivo f_m.

Paso 3. Para $m = 1, 2, \ldots, a$, asigne una distancia grande a las soluciones de los extremos, $d_{I_1^m} = d_{I_l^m} = \infty$, a todas las otras soluciones $j = 2, \ldots, (l-1)$, asigne:

$$d_{I_j^m} = d_{I_j^m} + \frac{f_m^{(I_{j+1}^m)} - f_m^{(I_{j-1}^m)}}{f_m^{\text{máx}} - f_m^{\text{mín}}}$$

El índice I_j^m denota el índice de la solución del $j - ésimo$ miembro en la lista ordenada, respecto a la función objetivo m. Así, para cualquier objetivo m, I_1^m e I_l^m denotan los valores de la función objetivo menor y mayor, respectivamente. El segundo término del lado derecho de la última ecuación es la diferencia en los valores de la función objetivo m entre dos soluciones vecinas en ambos lados de la solución I_j^m. Así, esta métrica denota la mitad del perímetro del cuboide encerrado con las soluciones vecinas más cercanas colocadas en los vértices del cuboide. Cabe señalar que para cualquier solución i las mismas dos soluciones $(i+1)$ e $(i-1)$ no necesitan ser vecinas en todos los objetivos, particularmente para $a \geq 3$. Los valores $f_m^{\text{máx}}$ y $f_m^{\text{mín}}$ pueden establecerse como los valores máximo y mínimo poblacional de la $m - ésima$ función objetivo.

La métrica anterior requiere se realicen a clasificaciones en el Paso 2, cada una requiere $O(N \log N)$ cálculos. Así, la complejidad computacional de esta métrica es $O(aN \log N)$.

5.4. Construcción del conjunto de referencia

El conjunto de referencia E se extrae del conjunto de soluciones iniciales D y consta de $n + m$ soluciones.

Se clasifica a la población en niveles no dominados, con el procedimiento descrito en la Sección 5.2. Cada vez que se determina un frente no dominado, su tamaño es usado para comprobar si puede ser incluido en el conjunto de referencia. Si esto no es posible, no se necesita seguir clasificando la población. Esto ayudará a reducir el tiempo de corrida del algoritmo [Deb01]. Cuando un frente de Pareto no dominado no puede ser incluido en el conjunto de referencia, se emplea el procedimiento descrito en la Sección

anterior para elegir las soluciones más ampliamente espaciadas usando los valores de la distancia de amontonamiento ordenadas de mayor a menor.

5.5. Método para actualizar el conjunto de referencia

La Tabla 5.1 muestra el número de iteraciones que realiza el algoritmo de BDMO:

Tabla 5.1
Número de iteraciones del algoritmo de BDMO.

Tamaño del caso $n+m$	Iteraciones a realizar
$n+m \leq 25$	12500
$25 < n+m \leq 50$	50000
$50 < n+m \leq 75$	185000
$n+m > 75$	225000

En el algoritmo para el problema con un objetivo se observó que el número de iteraciones en los casos con 20 clientes estaba sobrado, por ello, se redujo el número de iteraciones de los casos con menos de 26 clientes. Se obtuvieron resultados similares a los obtenidos realizando más iteraciones.

Se realiza una iteración j del procedimiento de diversificación si:

- j es múltiplo de diez y $(n+m) < 75$, o
- j es múltiplo de diez y $(n+m) \geq 75$ y $j \leq 5000$

En otro caso, se realiza una iteración combinando soluciones. La frecuencia con que se combinan soluciones y se emplea el procedimiento de diversificación parecen ser adecuadas [Cun97].

Durante la ejecución del algoritmo, cada vez que se genera una solución se guarda en una lista, si la solución obtenida es diferente de las ya contenidas en la lista. En esta lista las soluciones siempre están ordenadas de menor a mayor con respecto al costo de la solución. Cuando se genera una solución y la lista ya tiene 200 elementos, se actualiza el conjunto de referencia aplicando el procedimiento descrito en el segundo párrafo de la sección anterior.

5.6. Resultados experimentales

BDMO se implementó en Delphi 5 en una computadora personal con procesador Intel Core i5 a 2.53 GHz y 4 GB de memoria RAM, cada caso se resolvió 30 veces.

5.6.1. Casos de la tercera clase del PCTF biobjetivo

Se configuró BDMO para minimizar el número de rutas, objetivo $f_1(S)$ en la ecuación 2.1, y el costo total de la solución, $f_2(S)$ en la ecuación 2.2, simultáneamente, para resolver los casos de la tercera clase del PCTF como problemas biobjetivo. En la configuración de BDMO no se incluye a todos los clientes posibles en las rutas pares. Los casos de prueba de la tercera clase, pueden verse como un caso particular del PCTFR en el que no hay clientes de recolección, es decir, cuando $B = \varnothing$.

Tabla 5.2
Porcentaje de soluciones diversas en el conjunto D.

Caso	Porcentaje de soluciones diversas en el conjunto D
14, 15, 20	20 %
5, 6, 13, 16	33 %
3, 4, 17, 18	40 %
19	50 %

Se determinó qué porcentaje de soluciones diversas en el conjunto D era más adecuado para el algoritmo de BDMO en los casos de la tercera clase. Se probó experimentalmente con 20 %, 33 %, 40 %, 50 % y 100 %. En cada caso, se eligió como porcentaje a emplear aquel con el cual se obtuvieron mejores resultados. La Tabla 5.2 muestra el porcentaje de soluciones diversas en el conjunto D que se empleó en cada caso:

La Tabla 5.3 presenta resultados obtenidos por BDMO para el PCTF biobjetivo. Las primeras dos columnas muestran: el nombre del caso de prueba y el número de clientes. Las siguientes dos columnas muestran: el número de veces que BDMO evaluó la función objetivo y el tiempo de ejecución promedio de las 30 repeticiones de BDMO. Las dos últimas columnas muestran: la cardinalidad de la aproximación al primer frente encontrado por BDMO y las soluciones en éste. Cada solución en el primer frente se da como una pareja ordenada, la primera entrada de cada pareja presenta el número de rutas de la solución, la segunda, el costo de la solución.

El tiempo promedio de ejecución de BDMO es de 134.89 segundos. El tiempo de ejecución de BDMO es menor que el de BD porqué en BDMO se redujo en el número de iteraciones de los casos con menos de 26 clientes y se redujo el número de soluciones que se combinan con la solución diversa generada en el procedimiento de diversificación de los casos con menos de 51 clientes.

Tabla 5.3
Resultados de BDMO para el PCTF biobjetivo de los casos de la tercera clase, propuestos por Golden et al.

Caso	n	Núm. eva. f. o.	Tiempo prom. (s)	Cardinalidad 1er. frente	1er. frente
3	20	23950	6.82	3	(6,961.03), (5,974.79), (3,983.18)
4	20	23950	6.92	3	(6,6437.33), (5,6884.14), (3,7300.18)
5	20	23950	6.68	2	(5,1007.05), (3,1011.68)
6	20	23950	6.66	3	(6,6516.47), (5,6949.53), (3,7311.61)
13	50	95500	65.78	4	(8,2475.88), (7,2525.31), (6,2529.5), (5,2566.23)
14	50	95500	65.85	5	(8,9153.58), (7,9599.57), (6,10099.5), (5,11000.25), (4,11959.61)
15	50	95000	68.95	4	(10,2651.4), (7,2674.52), (6,2693.96), (5,2795.29)
16	50	95500	68.21	2	(9,2804.38), (6,2836.44)
17	75	223250	238.22	3	(7,1808.68), (6,1852.82), (4,1888.25)
18	75	223250	246.05	9	(14,2616.87), (11,2740.24), (10,2775.56), (9,2835.42), (8,2844.95), (7,2929.6), (6,3025.28), (5,3173.6), (4,3450.63)
19	100	276000	437.95	8	(12,9309.72), (11,9482.03), (10,9665.23), (9,9701.47), (8, 9891.18), (7,10232.28), (6,10744.99), (5,11293.95)
20	100	276000	400.64	5	(13,4334.5), (11,4404.69), (10,4484.76), (9,4549.05), (8,4557.66),
Promedio			134.89		

En 2 de los 12 casos de prueba la cardinalidad del primer frente es 2. Las Figuras 5.1 y 5.2 muestran el primer frente (de cardinalidad dos) y todas las soluciones generadas por BDMO para el caso 5, respectivamente.

Las Figuras 5.3, 5.4, 5.5 y 5.6 muestran el primer frente del PCTF biobjetivo de los casos 3, 14, 17 y 20 con 20, 50, 75 y 100 clientes, respectivamente.

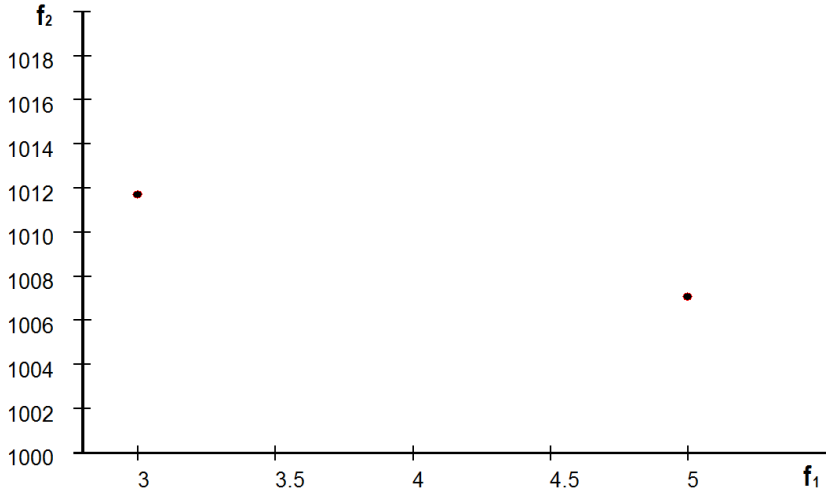

Figura 5.1: Primer frente obtenido por BDMO para el caso 5 del PCTF biobjetivo.

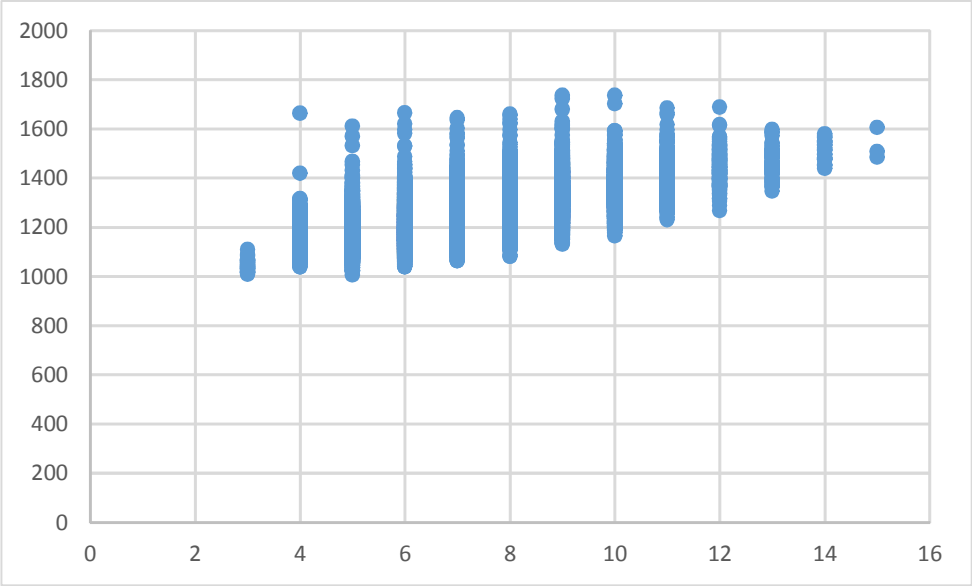

Figura 5.2: Soluciones generadas por BDMO al resolver el caso 5 del PCTF biobjetivo.

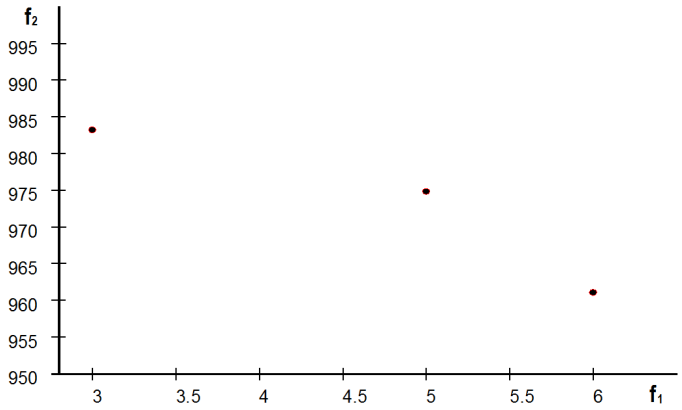

Figura 5.3: Primer frente obtenido por BDMO para el PCTF biobjetivo del caso 3.

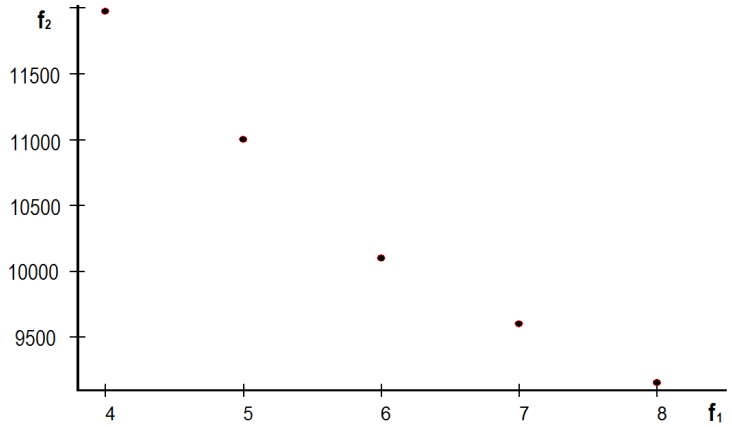

Figura 5.4: Primer frente obtenido por BDMO para el PCTF biobjetivo del caso 14.

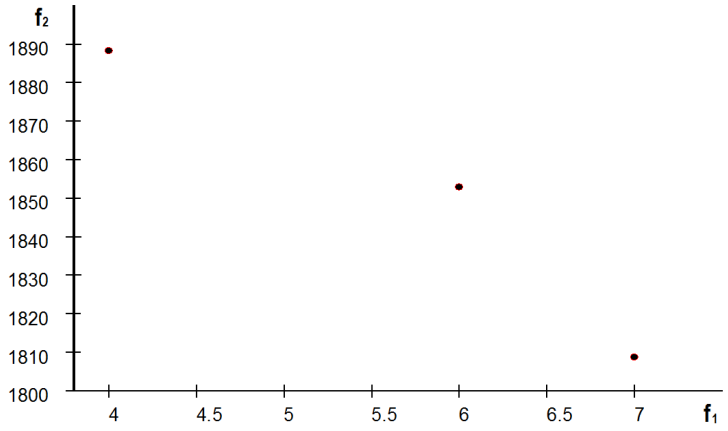

Figura 5.5: Primer frente obtenido por BDMO para el PCTF biobjetivo del caso 17.

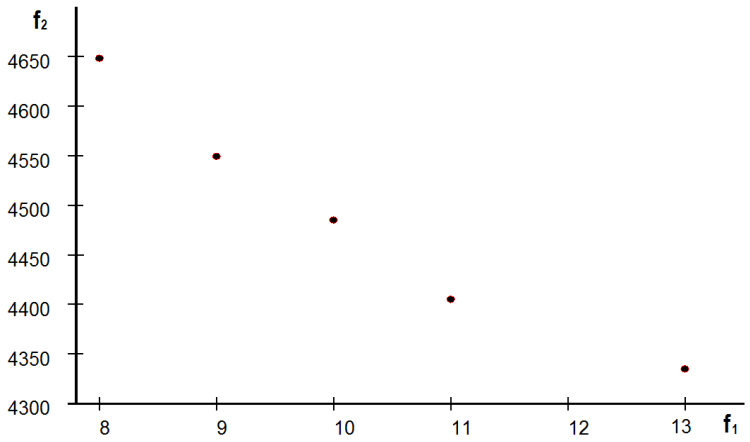

Figura 5.6: Primer frente obtenido por BDMO para el PCTF biobjetivo del caso 20.

A continuación se comparan los resultados de BDMO biobjetivo, minimizar el número de rutas, objetivo $f_1(S)$ en la ecuación 2.1, y el costo total de la solución, objetivo $f_2(S)$ en la ecuación 2.2, con las mejores soluciones conocidas del PCTF.

La Tabla 5.4 presenta los resultados de BDMO con un mínimo de rutas, es decir, respecto a la primera función objetivo, minimizar el número de rutas. La primera columna muestra el nombre del caso. Las siguientes dos columnas muestran: el costo de la mejor solución conocida y el número de rutas que le corresponden. Las siguientes dos columnas muestran: el costo de la mejor solución con menos rutas encontrada por BDMO después de 30 corridas y su correspondiente número de rutas. Las últimas dos columnas muestran: el porcentaje de la diferencia entre la mejor solución de BDMO y la mejor solución conocida con respecto a costo (%C) y número de rutas (%R), respectivamente. En todos los casos las mejores soluciones conocidas tienen más rutas que las encontradas por BDMO. Globalmente, las soluciones de BDMO están en promedio 14.7 % por arriba del costo conocido, sin embargo, reducen el número de rutas en un 52.87 %

Tabla 5.4
Casos de la tercera clase, propuestos por Golden et al. del PCTF biobjetivo.

	Mínimo de rutas					
	Mejor sol. conocida		BDMO			
Caso	Costo	Núm. de rutas	Costo	Núm. de rutas	%C	%R
3	961.03	6	983.18	3	2.30	-50.00
4	6437.33	6	7300.18	3	13.40	-50.00
5	1007.05	5	1011.68	3	0.46	-40.00
6	6516.47	6	7311.61	3	12.20	-50.00
13	2406.36	10	2566.23	5	6.64	-50.00
14	9119.03	8	11959.61	4	31.15	-50.00
15	2586.37	13	2795.29	5	8.08	-61.54
16	2720.43	9	2836.44	6	4.26	-33.33
17	1734.53	8	1888.25	4	8.86	-50.00
18	2369.65	16	3450.63	4	45.62	-75.00
19	8661.81	15	11293.95	5	30.39	-66.77
20	4032.81	19	4557.66	8	13.01	-57.89
Promedio					14.70	-52.87

La Tabla 5.5 presenta el resumen de los resultados de BDMO con un mínimo de costo total, es decir, respecto a la segunda función objetivo, minimizar el costo total de la solución [Gar10]. Las tres primeras columnas muestran: el nombre del caso, la mejor solución reportada en la literatura y el número de rutas (NR) que le corresponden. Las siguientes cuatro columnas muestran resultados de BDMO: la mejor solución, el NR de ésta, el promedio del mejor resultado de cada una de las 30 repeticiones y la desviación estándar de la mejor solución de las 30 repeticiones del algoritmo. Las dos últimas columnas muestran el porcentaje de la diferencia entre la mejor solución de BDMO y la mejor solución conocida con respecto a costo (%C) y número de rutas (%R), respectivamente.

BDMO encontró las mejores soluciones conocidas en todos los casos con 20 clientes. Globalmente, las soluciones de BDMO están en promedio 3.21 % por arriba del costo conocido, sin embargo, reducen el número de rutas en 9.97 %.

Tabla 5.5
Casos de la tercera clase, propuestos por Golden et al. del PCTF biobjetivo.

Mínimo de costo total								
	Mejor sol. conocida		BDMO					
Caso	Costo	NR	Costo	NR	Costo sol. prom.	Desv. est.	%C	%R
3	961.03	6	961.03	6	969.78	5.58	0.00	0.00
4	6437.33	6	6437.33	6	6629.93	208.94	0.00	0.00
5	1007.05	5	1007.05	5	1017.19	6.02	0.00	0.00
6	6516.47	6	6516.47	6	6695.49	201.78	0.00	0.00
13	2406.36	10	2475.88	8	2543.65	22.81	2.89	-20.00
14	9119.03	8	9153.58	8	9308.90	128.55	0.38	0.00
15	2586.37	13	2651.40	10	2718.32	23.25	2.51	-23.08
16	2720.43	9	2804.38	9	2873.27	27.71	3.09	0.00
17	1734.53	8	1808.68	7	1858.49	19.48	4.27	-12.50
18	2369.65	16	2616.87	14	2713.37	34.87	10.43	-12.50
19	8661.81	15	9309.72	12	9532.25	83.44	7.48	-20.00
20	4032.81	19	4334.50	13	4463.78	39.43	7.48	-31.58
Promedio							3.21	-9.97

5.6.2. Casos de la cuarta clase del PCTFR biobjetivo

Se resolvieron los casos de la cuarta clase del PCTFR como problemas biobjetivo, es decir, se configuro BDMO para minimizar el número de rutas,

objetivo $f_1(S)$ en la ecuación 2.1, y el costo total de la solución, $f_2(S)$ en la ecuación 2.2, simultáneamente.

Se determinó qué porcentaje de soluciones diversas en el conjunto D era más adecuado para el algoritmo de BDMO en los casos de la cuarta clase. Se probó experimentalmente con 20 %, 33 %, 40 %, 50 % y 100 %. En cada caso, se eligió como porcentaje a emplear aquel con el cual se obtuvieron mejores resultados. La Tabla 5.6 muestra el porcentaje de soluciones diversas en el conjunto D que se empleó en cada caso:

Tabla 5.6
Porcentaje de soluciones diversas en el conjunto D.

Caso	Porcentaje de soluciones diversas en el conjunto D
HWS23, HWS25, HWS30, HWS31, HWS34	20 %
HWS24, HWS27	33 %
HWS1 a HWS12, HWS14, HWS17, HWS20 HWS26, HWS35	40 %
HWS16, HWS19, HWS29, HWS32	50 %
HWS13, HWS15, HWS18, HWS21, HWS22, HWS28, HWS33, HWS36	100 %

La Tabla 5.7 presenta resultados obtenidos por BDMO para el PCTFR biobjetivo, usando los mismos encabezados que en la Tabla 5.3. El tiempo promedio de ejecución de BDMO es de 157.74 segundos. El tiempo promedio de ejecución de BDMO en los casos de la cuarta clase es menor al de BD, al igual que en los casos de la tercera clase, por las mismas razones: en BDMO se redujo en el número de iteraciones de los casos con menos de 26 clientes; y se redujo el número de soluciones que se combinan con la solución diversa generada en el procedimiento de diversificación de los casos con menos de 51 clientes.

Tabla 5.7
Resultados de BDMO para el PCTFR biobjetivo de los casos de la cuarta clase, propuestos por Salhi et al.

Caso	$n+m$	Núm. eva. f. o.	Tiempo prom. (s)	Cardinalidad 1er. frente	1er. frente
HWS1	20	23950	7.02	4	(6,726.47), (4,726.67), (3,742.45), (2,774.54)
HWS2	20	23950	6.92	2	(5,831.13), (3,833.16)
HWS3	20	23950	6.85	2	(4, 862.6), (3,885.91)

Tabla 5.7 (continuación)

Caso	$n+m$	Núm. eva. f. o.	Tiempo prom. (s)	Cardinalidad 1er. frente	1er. frente
HWS4	20	23950	6.72	2	(3,4350.65), (2,4834.34)
HWS5	20	23950	6.76	2	(3,5367.26), (2,6632.43)
HWS6	20	23950	6.60	3	(5,5447.81), (4,5894.02), (2,6293.9)
HWS7	20	23950	7.00	2	(3,730.84), (2,775.74)
HWS8	20	23950	6.82	2	(4,847.92), (3,850.2)
HWS9	20	23950	6.82	2	(4,890.76), (3,904.93)
HWS10	20	23950	6.72	2	(3,4349.13), (2,4829.61)
HWS11	20	23950	6.72	2	(3,5376.64), (2,6334.34)
HWS12	20	23950	6.06	3	(5,5519.06), (4,5953.65), (2,6307.16)
HWS13	50	95500	70.82	2	(4,1617.63), (3,1747.63)
HWS14	50	95500	68.69	3	(6,1842.09), (5,1855.37), (4,1896.9)
HWS15	50	95500	72.05	3	(6,2111.48), (5,2203.61), (4,2256.65)
HWS16	50	95500	65.48	3	(4,5572.67), (3,6053.14), (2,7542.45)
HWS17	50	95500	64.95	3	(5,6580.28), (4,7073), (3,8562.63)
HWS18	50	95500	65.07	4	(6,7178.02), (5,7648.03), (4,8576.01), (3,9061.48)
HWS19	50	95500	68.10	3	(5,1620.29), (4,1667.79), (3,1753.44)
HWS20	50	95500	65.08	4	(7,2057.92), (6,2092.24), (5,2125.33), (4,2212.37)
HWS21	50	95500	70.35	4	(8,2350.86), (7,2404.9), (6,2421.93), (5,2491.98)
HWS22	50	95500	68.95	3	(5,1739.71), (4,1820.74), (3,1823.72)
HWS23	50	95500	66.22	2	(6,2183.35), (4,2222.03)
HWS24	50	95500	66.14	2	(6,2498.58), (5,2677.41)
HWS25	75	213750	231.66	2	(5,1307.87), (3,1340.82)
HWS26	75	216750	232.90	3	(5,1442.03), (4, 1501.23), (3,1521.68)
HWS27	75	219250	258.14	3	(7,1571.68), (5,1605.78), (4,1662.12)

Tabla 5.7 (continuación)

Caso	$n+m$	Núm. eva. f. o.	Tiempo prom. (s)	Cardinalidad 1er. frente	1er. frente
HWS28	75	213750	232.88	6	(7,1727.18), (6,1823.82), (5,1879.22), (4,1925.45), (3,2074), (2,2313.75)
HWS29	75	216750	235.14	5	(8,1951.38), (7,2051.21), (5,2096.48), (4,2211.92), (3,2386.39)
HWS30	75	219250	258.00	6	(9,2153.88), (7,2333.44), (6,2404.27), (5,2554.69), (4,2722.2), (3,3146.07)
HWS31	100	263500	645.42	6	(8,5189.99), (7,5304.93), (6,5430.73), (5,5571.81), (4,5715.49), (3,6289.5)
HWS32	100	267500	398.02	6	(9,6367.94), (8,6494.47), (7,6623.79), (6,6754.07), (5,6949.76), (4,7518.1)
HWS33	100	271000	420.31	8	(11,7652.07), (10,7878.96), (9,7939.39), (8,7972.77), (7,8134.4), (6,8273.51), (5,8805.29), (4,9349.3)
HWS34	100	262500	1068.98	5	(8,2702.15), (7,2751.68), (6,2798.78), (5,2867.39), (4,2996.57)
HWS35	100	267500	396.83	4	(10,3147.09), (7,3178.01), (6,3402.14), (5,3562.51)
HWS36	100	271000	407.37	5	(10,3767.18), (9,3900.36), (8,3919.73), (7,3973.7), (6,4244.19)
Promedio			157.74		

En 13 de los 36 casos de prueba, la cardinalidad del primer frente es dos. Las Figuras 5.7 y 5.8 muestran el primer frente (de cardinalidad dos) y todas las soluciones generadas por BDMO, respectivamente, del caso HWS13 con 50 clientes.

Las Figuras 5.9, 5.10, 5.11 y 5.12 muestran el primer frente del PCTFR biobjetivo de los casos HWS6, HWS21, HWS29 y HWS33 con 20, 50, 75 y 100 clientes, respectivamente.

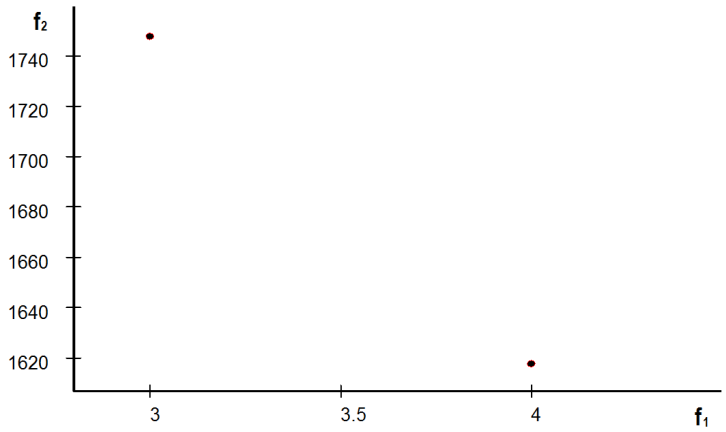

Figura 5.7: Primer frente de Pareto del PCTFR biobjetivo del caso HWS13.

Ahora se va a comparar los resultados de BDMO biobjetivo, minimizar el número de rutas, objetivo $f_1(S)$ en la ecuación 2.1, y el costo total de la solución, objetivo $f_2(S)$ en la ecuación 2.2, con las mejores soluciones conocidas del PCTFR[1].

La Tabla 5.8 presenta el resumen de los resultados de BDMO con un mínimo de rutas, es decir, respecto a la primera función objetivo, minimizar el número de rutas. Se emplearon los mismos encabezados que en la Tabla 5.4. En el caso HWS8 coincide el número de rutas de la mejor solución conocida, con el de la solución con un mínimo de rutas encontrada por BDMO, pero el costo de la solución encontrada está 1.44 % arriba del mejor costo conocido. En los 35 casos restantes, las mejores soluciones conocidas tienen más rutas que las soluciones con un mínimo de rutas encontradas por BDMO. Globalmente, las soluciones de BDMO con un mínimo de rutas están en promedio 16.88 % por arriba del mejor costo conocido, sin embargo, reducen el número de rutas en un 48.20 %.

[1]Salvo con las mejores soluciones de los casos HWS22, HWS31 y HWS34 (cuyos costos son 1717.60, 4943.29 y 2465.41, respectivamente), pues no se sabe cuál es el número de rutas de éstos. Se emplearon en estos casos el costo de la solución y el número de rutas que le corresponden, que amablemente fueron proporcionados por Puca Penna. En promedio el costo de estas soluciones difiere de las mejores soluciones conocidas en un 0.57 %.

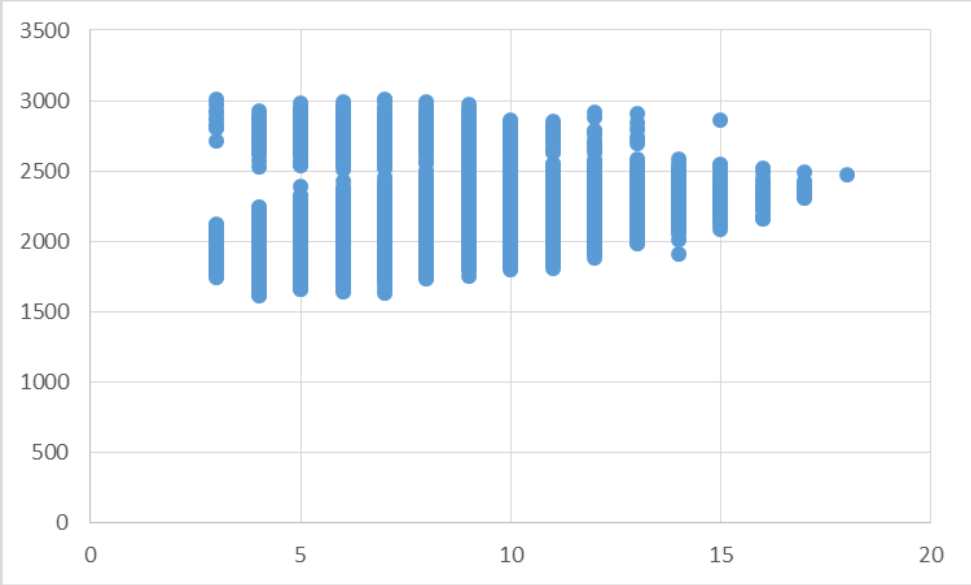

Figura 5.8: Soluciones generadas por BDMO al resolver el caso HWS13 del PCTFR biobjetivo.

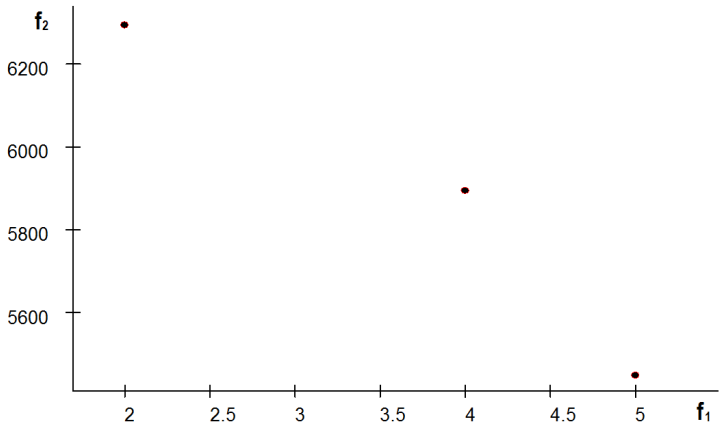

Figura 5.9: Primer frente de Pareto del PCTFR biobjetivo del caso HWS6.

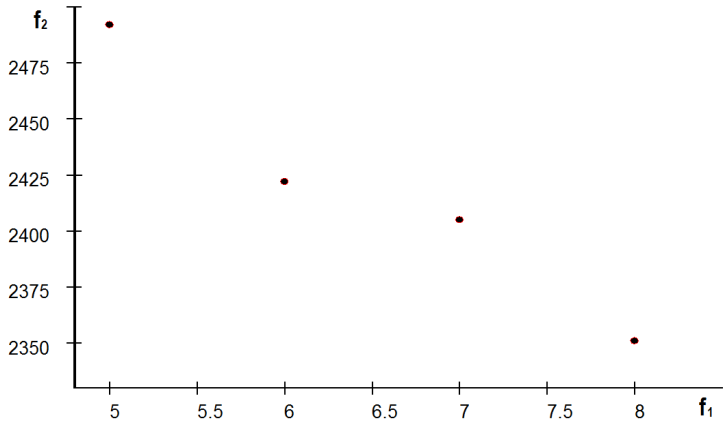

Figura 5.10: Primer frente de Pareto del PCTFR biobjetivo del caso HWS21.

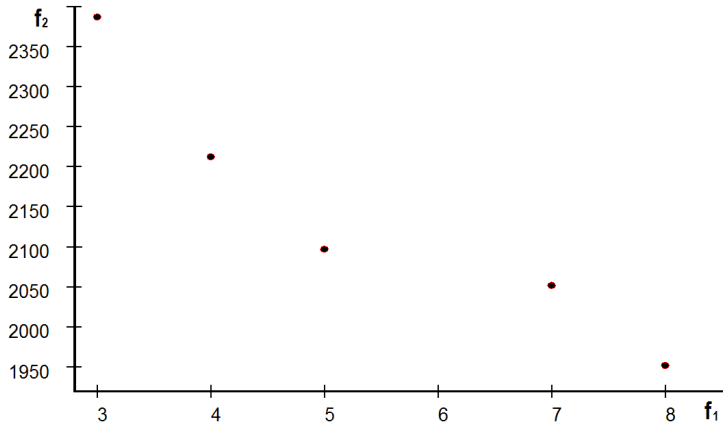

Figura 5.11: Primer frente de Pareto del PCTFR biobjetivo del caso HWS29.

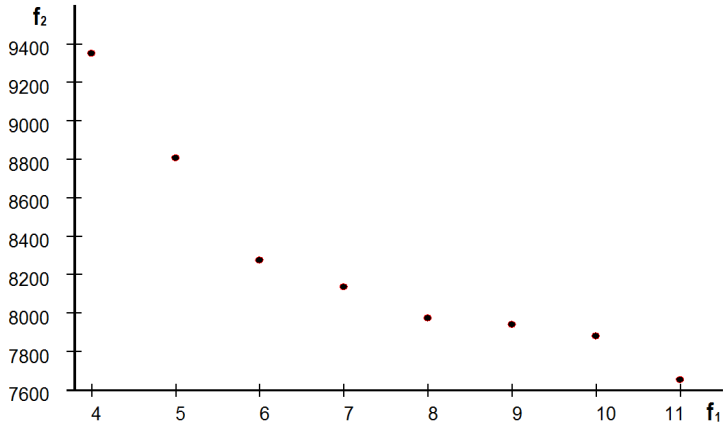

Figura 5.12: Primer frente de Pareto del PCTFR biobjetivo del caso HWS33.

Tabla 5.8
Casos de la cuarta clase, propuestos por Salhi et al. del PCTFR biobjetivo.

Mínimo de rutas						
	Mejor solución conocida		BDMO			
Caso	Costo	No. de rutas	Costo	No. de rutas	%C	%R
HWS1	720.57	6	774.54	2	7.49	-66.67
HWS2	818.12	6	833.16	3	1.84	-50.00
HWS3	848.32	4	885.91	3	4.43	-25.00
HWS4	4342.48	3	4834.34	2	11.33	-33.33
HWS5	5357.98	3	6632.43	2	23.79	-33.33
HWS6	5421.65	5	6293.90	2	16.09	-60.00
HWS7	729.50	3	775.74	2	6.34	-33.33
HWS8	838.11	3	850.20	3	1.44	0.00
HWS9	890.76	4	904.93	3	1.59	-25.00
HWS10	4349.13	3	4829.61	2	11.05	-33.33
HWS11	5363.58	3	6334.34	2	18.10	-33.33
HWS12	5497.98	5	6307.16	2	14.72	-60.00
HWS13	1590.47	10	1747.63	3	9.88	-70.00
HWS14	1771.53	8	1896.90	4	7.08	-50.00
HWS15	1999.05	8	2256.65	4	12.89	-50.00
HWS16	5551.19	4	7542.45	2	35.87	-50.00

Tabla 5.8 (continuación)

Mínimo de rutas						
	Mejor solución conocida		BDMO			
				No. de		
Caso	Costo	No. de rutas	Costo	rutas	%C	%R
HWS17	6547.93	5	8562.63	3	30.77	-40.00
HWS18	7120.52	6	9061.48	3	27.26	-50.00
HWS19	1616.21	6	1753.44	3	8.49	-50.00
HWS20	2015.67	8	2212.37	4	9.76	-50.00
HWS21	2295.57	8	2491.98	5	8.56	-37.50
HWS22	1720.24	5	1823.72	3	6.02	-40.00
HWS23	2096.10	7	2222.03	4	6.01	-42.86
HWS24	2401.04	8	2677.41	5	11.51	-37.50
HWS25	1285.86	5	1340.82	3	4.27	-40.00
HWS26	1399.36	5	1521.68	3	8.74	-40.00
HWS27	1513.10	8	1662.12	4	9.85	-50.00
HWS28	1572.38	11	2313.75	3	47.15	-72.73
HWS29	1760.95	12	2386.39	3	35.52	-75.00
HWS30	1950.99	13	3146.07	3	61.26	-76.92
HWS31	4963.08	8	6289.50	3	26.73	-62.50
HWS32	5993.30	10	7518.10	4	25.44	-60.00
HWS33	7097.81	12	9349.30	4	31.72	-66.67
HWS34	2494.95	9	2996.57	4	20.11	-55.56
HWS35	2927.20	11	3562.51	5	21.70	54.55
HWS36	3450.73	15	4244.19	6	22.99	-60.00
Promedio					16.88	-48.20

La Tabla 5.9 presenta el resumen de los resultados de BDMO con un mínimo de costo total, es decir, respecto a la segunda función objetivo, minimizar el costo total de la solución. En esta tabla se usaron los mismos encabezados que en la Tabla 5.5.

BDMO encontró las mejores soluciones conocidas en los casos HWS9 Y HWS10. En 16 casos la solución con un mínimo costo total encontrada, tiene el mismo número de rutas que la solución conocida, pero con un costo promedio 1.17 % mayor al costo conocido. En el caso HWS8 la mejor solución conocida domina a la mejor solución encontrada. En 17 casos las soluciones con un mínimo costo total de BDMO están 5.73 % arriba del mejor costo

conocido, sin embargo, reducen el número de rutas en un 22.35 %. Globalmente, las soluciones con un mínimo costo total de BDMO están 3 % arriba del mejor costo conocido, pero reducen el número de rutas en un 9.63 %.

Tabla 5.9
Casos de la cuarta clase, propuestos por Salhi et al. del PCTFR biobjetivo

Mínimo de costo total								
	Mejor sol. conocida		BDMO					
Caso	Costo	NR	Costo	NR	Costo sol. prom.	Desv. est.	%C	%R
HWS1	720.57	6	726.47	6	727.73	1.73	0.82	0.00
HWS2	818.12	6	831.13	5	837.13	5.94	1.59	-16.67
HWS3	848.32	4	862.60	4	870.58	2.29	1.68	0.00
HWS4	4342.48	3	4350.65	3	4350.97	0.64	0.19	0.00
HWS5	5357.98	3	5367.26	3	5367.30	0.04	0.17	0.00
HWS6	5421.65	5	5447.81	5	5529.73	167.83	0.48	0.00
HWS7	729.50	3	730.84	3	734.68	5.99	0.18	0.00
HWS8	838.11	3	847.92	4	855.99	4.08	1.17	33.33
HWS9	890.76	4	890.76	4	907.26	5.90	0.00	0.00
HWS10	4349.13	3	4349.13	3	4355.68	7.98	0.00	0.00
HWS11	5363.58	3	5376.64	3	5378.47	2.06	0.24	0.00
HWS12	5497.98	5	5519.06	5	5694.08	196.55	0.38	0.00
HWS13	1590.47	10	1617.63	4	1668.97	16.56	1.71	-60.00
HWS14	1771.53	8	1842.09	6	1883.15	15.34	3.98	-25.00
HWS15	1999.05	8	2111.48	6	2161.75	19.63	5.62	-25.00
HWS16	5551.19	4	5572.67	4	5594.36	11.12	0.39	0.00
HWS17	6547.93	5	6580.28	5	6608.67	15.00	0.49	0.00
HWS18	7120.52	6	7178.02	6	7382.90	144.07	0.81	0.00
HWS19	1616.21	6	1620.29	5	1663.04	23.97	0.25	-16.67
HWS20	2015.67	8	2057.92	7	2112.94	22.07	2.10	-12.50
HWS21	2295.57	8	2350.86	8	2406.81	21.14	2.41	0.00
HWS22	1720.24	5	1739.71	5	1781.75	17.58	1.13	0.00
HWS23	2096.10	7	2183.35	6	2223.81	23.49	4.16	-14.29
HWS24	2401.04	8	2498.58	6	2565.77	24.93	4.06	-25.00
HWS25	1285.86	5	1307.87	5	1332.01	20.28	1.71	0.00
HWS26	1399.36	5	1442.03	5	1483.41	19.98	3.05	0.00
HWS27	1513.10	8	1571.68	7	1611.43	15.12	3.87	-12.50
HWS28	1572.38	11	1727.18	7	1795.67	29.44	9.84	-36.36
HWS29	1760.95	12	1951.38	8	2057.62	37.48	10.81	-33.33
HWS30	1950.99	13	2153.88	9	2281.26	43.87	10.40	-30.77

Tabla 5.9 (continuación)

Mínimo de costo total								
	Mejor solución conocida		BDMO					
Caso	Costo	NR	Costo	NR	Costo sol. prom.	Desv. est.	%C	%R
HWS31	4963.08	8	5189.99	8	5368.34	54.00	4.57	0.00
HWS32	5993.30	10	6367.94	9	6516.59	59.34	6.25	-10.00
HWS33	7097.81	12	7652.07	11	7848.31	61.68	7.81	-8.33
HWS34	2494.95	9	2702.15	8	2795.97	34.10	8.30	-11.11
HWS35	2927.20	11	3147.09	10	3248.65	42.57	7.51	-9.09
HWS36	3450.73	15	3767.18	10	3888.04	47.84	9.17	-33.33
Promedio							3	-9.63

5.7. Resumen

En este capítulo se propone un algoritmo de BDMO para resolver el PCTFR biobjetivo. El cual emplea tres métodos y dos procedimientos de BD. El procedimiento de diversificación y el número de iteraciones del método para actualizar el conjunto de referencia, se modificaron de manera que el número de evaluaciones de la función objetivo que realiza BDMO fuera acorde al tamaño del caso a resolver.

Se incluyeron en este capítulo los procedimientos de clasificación de la población y distancia de amontonamiento propuestos en [Deb01, Deb02]. Los cuales se emplean para elegir y actualizar las soluciones del conjunto de referencia de BDMO.

Globalmente, respecto a la segunda función objetivo, minimizar el costo total de la solución, las soluciones de BDMO tanto en los casos de la tercera clase, como en los casos de la cuarta clase están en promedio 3 % arriba del costo conocido, sin embargo, reducen el número de rutas en un 10 %.

El comportamiento de los casos de las clases tercera y cuarta, respecto a la primera función objetivo, minimizar el número de rutas, no coincide, como con respecto a la segunda función objetivo. En los casos de la tercera clase, globalmente, las soluciones de BDMO están en promedio 15 % por arriba del costo conocido, sin embargo, reducen el número de rutas en un 53 %. En tanto que, en los casos de la cuarta clase, globalmente, las soluciones de BDMO están en promedio 17 % por arriba del costo conocido, sin embargo, reducen el número de rutas en un 48 %.

Capítulo 6

Conclusiones

Para obtener buenos resultados es importante determinar qué porcentaje de soluciones diversas debe emplearse en el conjunto de soluciones iniciales.

El número de evaluaciones de la función objetivo es importante, está relacionado directamente con el tiempo de ejecución del algoritmo, se debe tener presente al inicio del desarrollo de un algoritmo.

Se generó una solución diversa con el procedimiento descrito en la Sección 4.2 y se combinó con cada una de las soluciones del conjunto de referencia. Se observó que a veces se generan buenas soluciones (con un costo pequeño) combinando la solución diversa con las últimas soluciones del conjunto de referencia, es decir, con soluciones con un costo grande. Se trató de reducir el número de evaluaciones de la función objetivo, en los casos con 75 y 100 clientes, combinando la solución diversa con la mitad de los elementos del conjunto de referencia, sin embargo, no se logró llegar a los resultados que se obtienen si se combina la solución diversa con todos los elementos del conjunto de referencia. Se considera que no se llegó al resultado deseado por el argumento dado al inicio de este párrafo.

El hecho de que la flota de vehículos fuese heterogénea ayudó al algoritmo propuesto a mejorar las soluciones durante el proceso de búsqueda, cambiar de vehículo permitió cortar rutas, ayudó a la diversificación, sirvió como método de perturbación.

Emplear el criterio de clasificación no dominada propuesto por Deb, en lugar de permitir que un elemento entre en el conjunto de referencia sólo si es mejor que el peor elemento en éste último, permitió mejorar las soluciones obtenidas en un 0.19 %.

No se mejoran los resultados de los casos con 75 y 100 clientes incrementando el número de iteraciones que realiza el algoritmo. Esto se concluyó al

incrementar el número de iteraciones de dichos casos, cuando se buscó que el número de evaluaciones de la función objetivo que realiza el algoritmo fuese acorde al tamaño del caso a resolver. Se cree que para mejorar los resultados del algoritmo en estos casos habría que incluir más métodos de mejora y mecanismos para lograr una mayor diversidad entre las soluciones del conjunto de referencia.

En los casos de prueba propuestos por Golden et al. para el PCTF con un solo objetivo, incluir a todos los clientes posibles en las rutas pares, produce soluciones con menos rutas. Si no se incluye a todos los clientes posibles en las rutas pares se generan soluciones con más rutas. La desviación estándar de la mejor solución de las 30 corridas del algoritmo es menor cuando en las rutas pares entra todo cliente posible, y se incrementa cuando no se incluye a todos los clientes posibles en las rutas pares.

El método de búsqueda local 2-opt, permite eliminar todos los cruces en una ruta.

En [Mar06] dan un procedimiento básico de BD e indican cuándo realizar una aplicación del método de mejora. Los resultados de BD se ven afectados por la forma en cómo se aplica el método de mejora. La forma en cómo se obtuvieron mejores resultados fue aplicando 2-opt de manera que no hubiese cruces en las rutas de una solución.

Para mejorar los resultados del algoritmo propuesto se podría emplear la distancia de Jaccard [Gar15] para determinar cuáles elementos entran en el conjunto de referencia. Emplear la distancia entre soluciones propuesta en la Sección 4.4 como criterio para elegir los elementos del conjunto de referencia durante la ejecución del algoritmo. Implementar métodos de mejora con intercambios entre rutas. Los tres mejores algoritmos [Gaj09, Sub12, Pen17] para resolver el PRVR, el PCTF y el PCTFR, respectivamente, emplean intercambios entre rutas.

Bibliografía

[Anb12] Anbuudayasankar, S.P., Ganesh, K., Lenny Koh, S.C., Ducq, Y. *Modified savings heuristics and genetic algorithm for bi-objective vehicle routing problem with forced backhauls*, Expert Systems with Applications 39 (3), pp. 2296–2305, 2012.

[Awe98] Awerbuch, B., Azar, Y., Blum, A., Vempala, S., *New Approximation Guarantees for Minimum-Weight k-Trees and Prize-Collecting Salesmen*, SIAM Journal on Computing 28 (1), pp. 254 - 262, 1998.

[Bal89] Balas, E., *The prize collecting traveling salesman problem*, Networks 19 (6), pp. 621–636, 1989.

[Bal08] Baldacci, R., Battarra, M., Vigo, D., Routing a Heterogeneous Fleet of Vehicles. In Golden, B. L., Raghavan, S., Wasil, E., editors, *The Vehicle Routing Problem: Last Advances and New Challenges*, New York: Springer, pp.3-27, 2008.

[Bal09] Baldacci, R., Mingozzi, A., *A unified exact method for solving different classes of vehicle routing problems.* Mathematical Programming 120, pp. 347-380, 2009.

[Bel09] Belfiore, P., Yoshizaki, H. T. Y., *Scatter search for a real-life heterogeneous fleet vehicle routing problem with time windows and split deliveries in Brazil*, European Journal of Operational Research 199 (3),pp. 750-758, 2009.

[Bel13] Belfiore, P., Yoshizaki, H. T. Y., *Heuristic methods for the fleet size and mix vehicle routing problem with time windows and split deliveries,* Computers & Industrial Engineering 64 (2), pp. 589-601, 2013.

[Bel17] Belloso, J., Juan, A. A., Faulin, J., *An iterative biased-randomized heuristic for the fleet size and mix vehicle-routing problem with*

backhauls, International Transactions in Operational Research 00, pp. 1-13, 2017.

[Bra06] Brandão, J., *A new tabu search algorithm for the vehicle routing problem with backhauls*, European Journal of Operational Research 173 (2), pp. 540–555, 2006.

[Car88] Carpaneto,G., Martello, S., Toth, P., *Algorithms and codes for the assignment problem*, Annals of Operations Research 13 (1), pp. 191-223, 1988.

[Cho07] Choi, E., Tcha, D.-W., *A column generation approach to the heterogeneous fleet vehicle routing problem*, Computers and Operations Research 34, pp. 2080–2095, 2007.

[Chu95] Chu, P.C., Beasley, J.E., *A Genetic Algorithm for the Set Partitioning Problem*, The Management School, Imperial College, London, 1995.

[Coe07] Coello C., Carlos A., Lamont, G. B., Van V., D. A., *Evolutionary algorithms for solving multiobjective problems*, New York: Springer Verlag, 2nd ed., 2007.

[Cor01] Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C., *Introduction to Algorithms,* MIT Cambridge, Massachusetts, 2nd ed., 2001.

[Cor02] Corberán, A., Fernández, E., Laguna, M., Martí, R., *Heuristic solutions to the problem of routing school buses with multiple objectives*, Journal of the Operational Research Society 53, pp. 427–435, 2002.

[Cro58] Croes, G. A., *A Method for Solving Traveling-Salesman Problems*, Operations Research 6 (6), pp. 791-812, 1958.

[Cun97] Cung, Van-Dat, Mautor, Thierry, Michelon, Philippe, Tavares, Andréa, *A Scatter Search Based Approach for the Quadratic Assignment Problem.* In Proceeding of 1997 IEEE International Conference on Evolutionary Computation, pp. 165-169, 1997.

[Dan59] Dantzig, G. B., Ramser, J. H., *The Truck Dispatching Problem*, Management Science 6 (1), pp. 80-91, 1959.

[Deb01] Deb, K. *Multi-objective Optimization Using Evolutionary Algorithms*, New York: Wiley, 3rd reprinting, 2003.

[Deb02] Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., *A fast and elitist multiobjective genetic algorithm: NSGA-II*, IEEE Transactions on Evolutionary Computation 6 (2), pp.182 -197, 2002.

[Dei84] Deif, I., Bodin, J.H., *Extension of the Clarke and Wright algorithm for solving the vehicle routing problem with backhauling*. In Proceedings of the Babson Conference on Software Uses in Transportation and Logistic Management, pp.75-96, 1984.

[Del95] Dell'Amico, M., Maffioli, F., Varbrand, P., *On prize-collecting tours and the asymmetric travelling salesman problem*, International Transactions in Operational Research 2 (3), pp. 297–308, 1995.

[Don14] Dongarra, J. J., *Performance of various computers using standard linear equation software*, University of Tennessee, CS-89-85, 2014.

[Fei05] Feillet, D., Dejax, P., Gendreau, M., *Traveling Salesman Problems with Profits*, Transportation Science 39 (2), pp. 188-205, 2005.

[Fis81] Fisher, M. L., Jaikumar, R., *A generalized assignment heuristic for vehicle routing*, Networks 11 (2), pp. 109–124, 1981.

[Gaj09] Gajpal, Y., Abad, P. L., *Multi-ant colony system (MACS) for a vehicle routing problem with backhauls*, European Journal of Operational Research 196 (1), pp. 102–117, 2009.

[Gar10] García-Nájera, A., *Multi-Objective Evolutionary Algorithms for Vehicle Routing Problems*, Ph.D. thesis, School of Computer Science, College of Engineering and Physical Sciences, University of Birmingham, Birmingham, United Kingdom, 2010.

[Gar11] García-Nájera, A., Bullinaria, J. A., *An improved multi-objective evolutionary algorithm for the vehicle routing problem with time windows*, Computers and Operations Research 38 (1), pp. 287-300, 2011.

[Gar12] García-Nájera, A., *The vehicle routing problem with backhauls: a multi-objective evolutionary approach*, Evolutionary Computation in Combinatorial Optimization, Lecture Notes in Computer Science 7245, pp. 255-266, 2012.

[Gar15] Garcia-Najera, A., Bullinaria, J. A., Gutiérrez-Andrade, M. A., *An evolutionary approach for multi-objective vehicle routing problems with backhauls*, Computers & Industrial Engineering 81, pp. 90–108, 2015.

[Gen99] Gendreau, M., Laporte, G., Musaraganyi, C., Taillard, E. D., *A tabu search heuristic for the heterogeneous fleet vehicle routing problem*, Computers & Operations Research 26 (12), pp. 1153–1173, 1999.

[Glo77] Glover, F., *Heuristics for integer programming using surrogate constraints*, Decision Science 8 (1), pp. 156–166, 1977.

[Goe89] Goetschalckx, M., Jacobs-Blecha, C., *The vehicle routing problem with backhauls*, European Journal of Operational Research 42 (1), pp. 39–51, 1989.

[Gol84] Golden, B., Assad, A., Levy, L., Gheysens, F., *The fleet size and mix vehicle routing problem*, Computers and Operations Research 11 (1), pp. 49-66, 1984.

[Gol85] B.Golden, E. Baker, J. Alfaro and J. Schaffer, *The vehicle routing problem with Backhauling: two approaches*. In R. Hammesfahr (Ed.) Proceedings of the XXI Annual Meeting of S. E. TIMS, pp. 90-92, 1985.

[Gol08] Golden, B., Raghavan, S., Wasil, E., editors, *The Vehicle Routing Problem: Last Advances and New Challenges*, New York: Springer, 2008.

[Hof10] Hoff, A., Andersson, H., Christiansen, M., Hasle, G., Lokketangen, A., *Industrial aspects and literature survey: fleet composition and routing*, Computers & Operations Research 37, pp. 2041-2061, 2010.

[Imr09] Imran, A., Salhi, S., Wassan, N. A., *A variable neighborhood-based heuristic for the heterogeneous fleet vehicle routing problem*, European Journal of Operational Research 197, pp. 509-518, 2009.

[Jac92] Jacobs-Blecha, C., Goetschalckx, M., *The vehicle routing problem with backhauls: properties and solution algorithms*, Technical Report, School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, Georgia. Presented at the National

Transportation Research Board, January 13-15, 1992, Washington D. C., 1992.

[Joz08] Jozefowiez, N., Semet, F., Talbi El-G., *Multi-objective vehicle routing problems*, European Journal of Operational Research 189 (2), pp. 293–309, 2008.

[Joz08b] Jozefowiez, N., Semet, F., Talbi El-G.,From Single-Objective to Multi-Objective Vehicle Routing Problems: Motivations, Case Studies, and Methods. In Golden, B. L., Raghavan, S., Wasil, E., editors, *The Vehicle Routing Problem: Last Advances and New Challenges*, New York: Springer, pp.445-471, 2008.

[Kat88] Kataoka, S., Morito, S., *An algorithm for the single constraint maximum collection problem*, Journal of the Operations Research Society of Japan, 31, pp. 515-530, 1988.

[Kel85] Keller, C.P., Multiobjective routing through space and time: The MVP and TDVRP problems. Ph. D. thesis, Department of Geography, University of Western Ontario, London, Ontario, Canada, 1985.

[Kel88] Keller, C.P., *The multiobjective vending problem: a generalization of the travelling salesman problem*, Environment and Planning B: Planning and Design 15 (4), pp. 447 – 460, 1988.

[Kno99] Knowles, J., Corne, D., *The Pareto archived evolution strategy: A new baseline algorithm for multiobjective optimization*, in Proceedings of the 1999 Congress on Evolutionary Computation, IEEE Press, pp. 98 - 105, 1999.

[Lap90] Laporte, G., Martello, S., *The selective travelling salesman problem*, Discrete Applied Mathematics 26 (2–3), pp. 193–207, 1990.

[Lap09] Laporte, G., *Fifty years of vehicle routing*, Transportation Science 43 (4), pp. 408-416, 2009.

[Lee98] Lee, T-R, Ueng, J-H, *A study of vehicle routing problems with load-balancing*, International Journal of Physical Distribution & Logistics Management 29 (10), pp. 646 - 657, 1998.

[Len81] Lenstra, J. K., Kan, A. H. G. R., *Complexity of Vehicle Routing and Scheduling Problems,* Networks 11 (2), pp. 221-227, 1981.

[Lin65] Lin S., *Computer solutions of the travelling salesman problem*, Bell System Technical Journal 44, pp. 2245-2269, 1965.

[Mar06] Martí, R., Laguna, M., Glover, F., *Principles of scatter search*, European Journal of Operational Research 169, pp. 359–372, 2006.

[Mei11] Mei, Y., Tang. K., Yao, X., *Decomposition-based memetic algorithm for multiobjective capacitated arc routing problem*, IEEE Transactions on Evolutionary Computation 15 (2), pp. 151-165, 2011.

[Mel14] Melián-Batista, B., De Santiago, A., AngelBello, F., Alvarez, A., *A bi-objective vehicle routing problem with time windows: A real case in Tenerife*, Applied Soft Computing 17, pp. 140-152, 2014.

[Min99] Mingozzi, A., Giorgi, S., Baldacci, R., *An Exact Method for the Vehicle Routing Problem with Backhauls*, Transportation Science 33 (3), pp. 315-329, 1999.

[Och98] Ochi, L.S., Vianna D.S., Drummond, L.M.A., Victor, A.O., *A parallel evolutionary algorithm for the vehicle routing problem with heterogeneous fleet*, Future Generation Computer Systems 14, pp. 285-292, 1998.

[Osm02] Osman, I. H., Wassan, N. A., *A reactive tabu search meta-heuristic for the vehicle routing problem with back-hauls*, Journal of Scheduling 5 (4), pp. 263–285, 2002.

[Pac06] Pacheco, J., Martí, R., *Tabu Search for a Multi-Objective Routing Problem*, Journal of the Operational Research Society 57, pp. 29-37, 2006.

[Par12] Partyka, J., Hall, R., *Software Survey: Vehicle Routing,* ORMS-Today 39 (1), 2012.

[Par14] Partyka, J., Hall, R., *Vehicle Routing Software Survey: VR delivers the goods*, ORMS-Today 41 (1), 2014.

[Pen09] Peng, W., Zhang, Q., Li, H., Comparison between MOEA/D and NSGA-II on the multi-objective travelling salesman problem. In Goh CK., Ong YS., Tan K.C. (eds) *Multi-objective memetic algorithms*. Studies in Computational Intelligence, vol. 171, pp. 309-324. Springer, Berlin, Heidelberg, 2009.

[Pen17] Penna, P.H. V., Subramanian, A., Ochi, L. S., Vidal, T., Prins, C., *A hybrid heuristic for a broad class of vehicle routing problems with heterogeneous fleet*, Ann. Oper. Res., doi 10.1007/s10479-017-2642-9, 2017.

[Pes07] Pessoa, A., Poggi, M., Uchoa, E., *A robust branch-cut-and-price algorithm for the heterogeneous fleet vehicle routing problem.* En Demetrescu C. (eds). WEA 2007, Lecture Notes in Computer Science 4525, pp.150-160. Springer, Heidelberg, 2007.

[Pra13] Pradenas, L., Oportus, B., Parada, V., *Mitigation of greenhouse gas emissions in vehicle routing problems with backhauling*, Expert Systems with Applications 40 (8), pp. 2985-2991, 2013.

[Roc95] Rochat, Y., Taillard, E., *Probabilistic diversification and intensification in local search for vehicle routing*, Journal of Heuristics 1 (1), pp. 147-167, 1995.

[Rop06] Ropke, S., Pisinger, D., *A unified heuristic for a large class of vehicle routing problems with backhauls*, European Journal of Operational Research 171 (3), pp. 750-775, 2006.

[Rus06] Russell, R. A., Chiang, W.-C., *Scatter search for the vehicle routing problem with time windows,* European Journal of Operational Research 169 (2), pp. 606-622, 2006.

[Sal13] Salhi, S., Wassan, N., Hajarat, M., *The Fleet Size and Mix Vehicle Routing Problem with Backhauls: Formulation and Set Partitioning-based Heuristics*, Transportation Research Part E: Logistics and Transportation Review 56, pp. 22-35, 2013.

[Seg11] Segura Pérez, E., *Particularidades geométricas que mejoran el desempeño de los algoritmos de búsqueda local que resuelven el problema del agente viajero euclideano,* tesis para obtener el grado de Doctor en Ingeniería de Sistemas, Facultad de Ingeniería, Universidad Nacional Autónoma de México, 2011.

[Sil13] Silva, M. A. C., Klein, C. E., Mariani, V. C., Coelho, L. S., *Multiobjective scatter search approach with new combination scheme applied to solve environmental/economic dispatch problem*, Energy 53, pp. 14-21, 2013.

[Sub12] Subramanian, A., Penna, P.H.V., Uchoa, E., Ochi, L.S., *A hybrid algorithm for the Heterogeneous Fleet Vehicle Routing Problem*, European Journal of Operational Research 221, pp. 285-295, 2012.

[Tai99] Taillard, E., *A heuristic column generation method for the heterogeneous fleet vrp*, RAIRO - Operations Research - Recherche Opérationnelle 33 (1), pp.1-14, 1999.

[Tan10] Tang, J., Zhang, J., Pan, Z., *A scatter search algorithm for solving vehicle routing problem with loading cost*, Expert Systems with Applications 37 (6), pp. 4073-4083, 2010.

[Tot97] Toth, P., Vigo, D., *An Exact Algorithm for the Vehicle Routing Problem with Backhauls*, Transportation Science 31 (4), pp. 372-385, 1997.

[Tot99] Toth, P., Vigo, D., *A heuristic algorithm for the symmetric and asymmetric vehicle routing problems with backhauls*, European Journal of Operational Research 113 (3), pp. 528–543, 1999.

[Tot01] Toth, P., Vigo, D., VRP with backhauls. In Toth, P., Vigo, D., editors, *The vehicle routing problem*, SIAM, pp. 195-224, 2001.

[Tot14] Irnich, S., Schneider, M., Vigo, D., Four Variants of the Vehicle Routing Problem. En Toth, P., D. Vigo, D. editors, *Vehicle routing: problems, methods, and applications*, SIAM, pp. 241-271, 2014.

[Tut10] Tütüncü, G.Y., *An interactive GRAMPS algorithm for the heterogeneous fixed fleet vehicle routing problem with and without backhauls*, European Journal of Operational Research 201, pp. 593-600, 2010.

[Van99] Van Veldhuizen, D. A., *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*, Ph. D. thesis, Graduate School of Engineering, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, 1999.

[Was02] Wassan, N.A., Osman, I.H., *Tabu search variants for the mix fleet vehicle routing problem,* Journal of the Operational Research Society 53, pp. 768-782, 2002.

[Zha07] Zhang, Q., Li, H., *MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition*, IEEE Transactions on Evolutionary Computation 11 (6), pp. 712-731, 2007

[Zha12] Zhang, T., Chaovalitwongse, W. A., Zhang, Y., *Scatter search for the stochastic travel-time vehicle routing problem with simultaneous pick-ups and deliveries*, Computers & Operations Research 39 (10), pp. 2277-2290, 2012.

[Zit98] Zitzler, E., Thiele, L., "Multiobjective optimization using evolutionary algorithms - A comparative case study", Parallel Problem Solving from Nature - PPSN V, Series: Lecture Notes in Computer Science 1498, pp 292-301, Springer, 1998.

[Zit99] Zitzler, E., *Evolutionary algorithms for multiobjective optimization: Methods and applications*, Doctoral dissertation ETH 13398, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, 1999.

[Zit00] Zitzler, E., Deb, K., Thiele, L., *Comparison of Multiobjective Evolutionary Algorithms: Empirical Results*, Evolutionary Computation 8 (2), pp. 173-195, 2000.