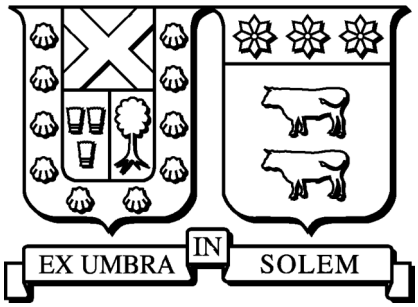
Universidad Técnica Federico Santa María
Departamento de Informática
Valparaíso - Chile



Una técnica de resolución para el Bus Evacuation Problem. Aplicación a la región de Valparaíso.

Javiera Pilar Loyola Vitali
javiera.loyola@alumnos.usm.cl

MEMORIA DE TITULACIÓN PARA OPTAR AL TÍTULO DE INGENIERO CIVIL EN INFORMÁTICA

Profesor Guía: Dra. María Cristina Riff Rojas
Profesor Correferente: Dra. Elizabeth Montero Ureta
Octubre 2016

Agradecimientos

Agradezco a mi familia, por estar siempre conmigo y apoyarme en todo momento. En especial a mis padres por, con mucho esfuerzo, haberme guiado en la dirección de la persona que soy. Ha sido duro, pero me han ayudado a crecer.

A Darío, por acompañarme todo este tiempo, y motivarme a ser mejor cada día, a no rendirme ante la adversidad y a luchar por lograr mis metas. Este trabajo es también suyo.

A la profesora María Cristina Riff por creer en mí, acogerme y enseñarme tantas cosas, no sólo en lo académico, sino también en lo personal. Además, a la profesora Elizabeth Montero y al equipo del laboratorio de IA, por todo el apoyo y desbordante carisma.

Finalmente, un agradecimiento a todos aquellos que, de una u otra forma, me ayudaron a llegar hasta aquí.

Resumen

El Bus Evacuation Problem (BEP) es un problema de planificación de rutas, dentro del contexto de una evacuación en una situación de emergencia. Considerando que se cuenta con transporte público para realizar la evacuación, el objetivo del problema es determinar las mejores rutas de cada uno de los vehículos, para llevar a todas las personas dentro de la zona de riesgo a refugios habilitados, de modo que la evacuación dure lo menos posible. En este trabajo, se presenta un estudio de los distintos enfoques que existen en la literatura para resolver el problema, para luego proponer un método basado en Greedy Randomized Adaptive Search Procedure (GRASP), con el objetivo de abordar un escenario real basado en el Gran Incendio de Valparaíso (Chile, 2014). En experimentos computacionales, se muestra que el algoritmo propuesto permite resolver instancias de tamaño real, además de superar en rendimiento a un solver comercial MIP.

Palabras Claves: Bus Evacuation Problem, Planificación, Control de desastres, Metaheurísticas.

Abstract

The Bus Evacuation Problem (BEP) is a route planning problem, in the context of an evacuation in an emergency situation. Considering that public transport is available to support the evacuation, the objective of the problem is to determine the best route for each one of the vehicles, to move all the people from a risk zone to available shelters located in safe zones, so that the evacuation time is minimized. In this work, a study of the different approaches used on the literature to solve the problem is presented, to later propose a method based on Greedy Randomized Adaptive Search Procedure (GRASP), in order to solve a real scenario based on the Great Fire of Valparaíso (Chile, 2014). In computational experiments, it is shown that the proposed algorithm is effective to solve real-world size problems, and able to outperform a commercial MIP solver.

Keywords: Bus Evacuation Problem, Evacuation planning, Disaster management, Metaheuristics.

Índice general

1 Introducción

Desde siempre la humanidad se ha visto enfrentada a las catástrofes naturales que ocurren en todo el mundo, como huracanes, tsunamis, así como a eventos de gran magnitud causados por el mismo hombre, como incendios masivos o bombardeos en zonas de guerra. Por este motivo, han surgido ramas de la Investigación Operativa que se han dedicado a estudiar la evacuación de una gran cantidad de personas que se encuentren en una zona en peligro, con el objetivo de minimizar el número de víctimas o maximizar el número de personas rescatadas exitosamente. Acontecimientos recientes como el Huracán Katrina en el año 2005, han puesto en evidencia la necesidad de planificar con mayor detalle la evacuación de individuos, que no pueden escapar por su cuenta. El Problema de Evacuación en Buses (BEP) consiste en la planificación de la evacuación a gran escala de individuos que se encuentran en una región en la que ha ocurrido, o se sabe con muy poco tiempo de anticipación que ocurrirá una catástrofe. En este contexto, se busca trasladar a estas personas a refugios específicos, con la ayuda de transporte público disponible especialmente para la ocasión. Es un asunto de gran importancia, ya que considera la movilización de individuos que son "tránsito-dependientes", que no tienen la posibilidad de movilizarse por su cuenta con algún automóvil particular o medio rápido de traslado. Algunos casos son familias de bajos recursos que no poseen transporte propio, ancianos o enfermos con problemas de movilidad, y personas extranjeras cuyo manejo del idioma de la zona no es suficiente para poder movilizarse de manera independiente [2].

Este problema fue planteado formalmente por primera vez en el año 2011 en [2]. Anteriormente, el área de planificación de evacuaciones ha estudiado otros escenarios que incluyen la evacuación a gran escala mediante el uso de automóviles, el desalojo de estadios o edificios, entre otras aplicaciones. La aplicación de técnicas provenientes del área de la inteligencia artificial e investigación de operaciones para resolver este tipo de problemas es importante, ya que es necesario considerar numerosos factores y los resultados se requieren en el menor tiempo posible, en especial cuando la planificación se realiza en el mismo momento o instantes previos a la evacuación.

El BEP busca principalmente planificar las rutas que deben seguir los buses para evacuar a los individuos de cierta zona de riesgo. Específicamente, se considera que estos sujetos se reúnen en varios puntos de encuentro distribuidos dentro de la zona. Los buses se encuentran inicialmente en una o varias estaciones, desde donde deben salir a buscar a individuos para llevarlos a los refugios disponibles. Para esto se debe considerar adicionalmente que tanto los buses como los refugios tienen una capacidad limitada que no puede ser excedida. El objetivo es minimizar el tiempo de duración de la evacuación, a la vez que se traslade a todos

los evacuados a los refugios, respetando las restricciones de capacidad mencionadas. Este problema se puede ver como una variante del conocido problema de optimización Vehicle Routing Problem, aunque difiere en varios aspectos que añaden complejidad al problema. Esto hace que el problema sea NP-completo, tal como se señala en [11].

Este documento está organizado de la siguiente manera. Inicialmente, en la Sección 1 se estudiará el problema de BEP en profundidad, indicando cada una de las componentes y restricciones que se deben considerar. Luego, en la Sección 2 se estudiarán algunas de las técnicas que se han utilizado para abordar el problema, así como también los distintos enfoques que se le ha dado en la literatura, junto con los diferentes resultados obtenidos a partir de la aplicación de las técnicas propuestas y de tendencias actuales. En la Sección 3 se presenta el modelo considerado para la resolución del problema, junto con una descripción detallada de la técnica que se implementó, que corresponde a un algoritmo basado en la metaheurística Greedy Randomized Adaptive Search Procedure (GRASP). En la Sección 4 se describen las instancias generadas, tanto las generadas aleatoriamente como unas instancias diseñadas en base a un escenario real ubicado en la Región de Valparaíso. Además, se describen los experimentos realizados. En la Sección 5 se muestran los resultados obtenidos de las pruebas, y finalmente se presentan las conclusiones del trabajo realizado.

2 Definición del problema

El Problema de Evacuación en Buses (BEP) consiste en la planificación de la evacuación a gran escala de individuos que se encuentran en una región determinada. El contexto asociado corresponde a una situación de catástrofe, donde se hace necesario trasladar a estas personas a refugios específicos con la ayuda de transporte público disponible. Es un asunto de gran importancia, ya que considera la movilización de individuos que son "tránsito-dependientes", que se encuentran en alguna situación particular que les impide movilizarse por su cuenta con algún automóvil particular o medio rápido de traslado. Algunos casos son familias de bajos recursos que no poseen transporte propio, ancianos o enfermos con problemas médicos y/o de movilidad, y personas extranjeras cuyo manejo del idioma de la zona no es suficiente para poder manejar la situación por su cuenta [2].

Este problema fue planteado formalmente por primera vez en el año 2011 por Douglas R. Bish [2]. Anteriormente, el área de planificación de evacuaciones a gran escala se centró principalmente en la evacuación realizada en automóviles, pero acontecimientos recientes, como el Huracán Katrina que afectó el área de Nueva Orleans, EE.UU. en el año 2005, han puesto en evidencia la necesidad de considerar también a aquellos individuos que no pueden evacuar por su cuenta, que en aquella ocasión resultaron ser los más afectados con la catástrofe, al no ser capaces de abandonar la zona a tiempo por sus propios medios [2, 7].

Un escenario de evacuación se compone de los siguientes elementos [2, 11, 12]:

- **Evacuados:** conjunto de personas que requieren asistencia y movilización para abandonar una zona de peligro. En la propuesta de Bish el número de evacuados se asume conocido con anticipación.
- **Buses:** se considera que para realizar la evacuación se cuenta con una flota de buses que trasladarán a los evacuados a los refugios. Cada bus tiene una capacidad máxima de personas que puede trasladar, la que se considera igual para todas las máquinas.
- **Puntos de encuentro:** puntos distribuidos en toda la zona de riesgo, donde la población se reúne para esperar el arribo de los buses de evacuación.
- **Refugios:** se cuenta con una serie de refugios pre-establecidos a donde serán llevados los evacuados. Cada refugio tiene una capacidad limitada de personas que puede albergar.

- **Estaciones:** puntos donde se encuentran las flotas de buses antes de comenzar la evacuación.

Las estaciones, puntos de encuentro, refugios, y las distancias que hay entre ellos, son representados mediante un grafo, que muestra los nodos que están conectados entre sí. En la Figura 2.1 se puede ver el ejemplo de un escenario que considera una estación, tres refugios y tres puntos de encuentro.

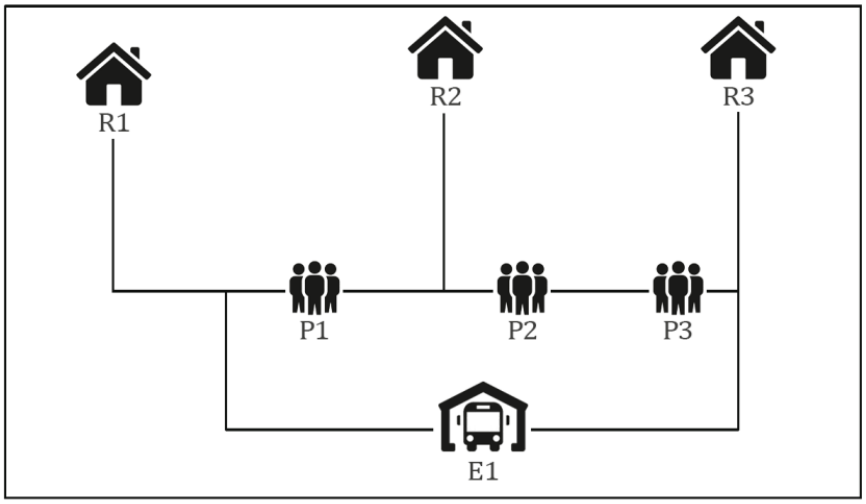


Figura 2.1: Ejemplo de un escenario simple de evacuación. Fuente: Elab. Propia.

Para comenzar la evacuación, las personas que se encuentren dentro de la zona de riesgo deben dirigirse al punto de encuentro más cercano, que también se denominan "*nodos de recogida*" o "*fuentes*" (*sources*) en la literatura [12]. Los buses, inicialmente ubicados en las estaciones, salen a recoger a los evacuados a los puntos donde se encuentran reunidos. Los buses deben realizar varias rondas para recoger a todos los individuos que estén en los puntos de encuentro y llevarlos a alguno de los refugios, conocidos en la literatura como "*sumideros*" (*sinks*), que aún tenga capacidad para albergar a más personas. El problema consiste en planificar de manera óptima la ruta que debe seguir cada bus cuando viaja de un punto de encuentro a un refugio, y luego de vuelta a otro punto de encuentro, de modo que la evacuación se lleve a cabo en el menor tiempo posible. El procedimiento completo se considera finalizado cuando todos los evacuados han sido trasladados a los refugios.

Formalmente, el BEP propuesto en [2] está dado por:

- **Parámetros de entrada:** El número de buses B, de puntos de encuentro P y de refugios S. Las distancias entre estaciones y puntos de encuentro, y entre estos últimos y los refugios. El número de personas en cada punto de encuentro, y las capacidades de cada refugio. Adicionalmente, se puede especificar la capacidad de los buses.

- **Objetivo:** Generar un recorrido para cada bus buscando minimizar la duración total de la evacuación. Esto se logra minimizando el tiempo total del bus que más se tarda en realizar todas sus rutas de evacuación.
- **Restricciones:** Todos los individuos deben ser evacuados y llevados a los refugios, además deben respetarse las restricciones asociadas a las capacidades máximas de buses y refugios.

Para describir un escenario se usará la notación $EiPjRkBv$, similar a la usada en [2], donde i, j, k y v, respectivamente, corresponden al número de estaciones, puntos de encuentro, refugios y buses. Usando un ejemplo de [12], donde se tiene una estación, tres puntos de encuentro, tres refugios y tres buses. La notación correspondiente sería $E1P3R3B3$. El número de personas en cada punto es $l = (1,3,3)$, y los refugios tienen capacidades de $u = (4,4,1)$. Las distancias del nodo estación a los puntos de encuentro son $d^{ini} = (7,4,9)$, y las distancias entre P (puntos de encuentro) y S (refugios) están dadas por la siguiente matriz:

$$d = \begin{pmatrix} 6 & 7 & 8 \\ 10 & 9 & 2 \\ 6 & 3 & 7 \end{pmatrix}.$$

La Figura 2.2 muestra la representación de la instancia mediante un grafo.

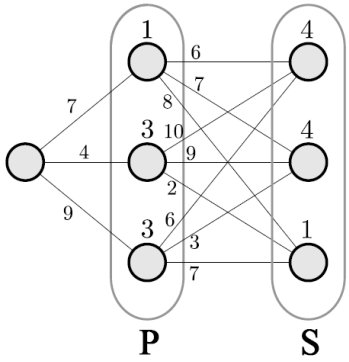


Figura 2.2: Instancia de ejemplo de BEP. Fuente [12].

Una solución factible y óptima para este ejemplo se presenta en la Tabla 2.1, compuesta por una serie de pares ordenados de la forma *(punto_encuentro,refugio)* que indica dónde se recogen los evacuados y hacia dónde son llevados en cada viaje. El primer bus viaja del punto de encuentro 1 al refugio 1, y luego del punto de encuentro 3 al refugio 2. El tiempo total que demora en completar la evacuación es de $d_1^{ini} + d_{11} + d_{31} + d_{32} = 7 + 6 + 6 + 3 = 22$. De

igual manera se calcula el tiempo de evacuación del bus 2 y del bus 3, resultando un valor de 23 en ambos casos. Finalmente, el tiempo total de evacuación es de 23.

Viaje n°	1	2	3
Bus 1	(1,1)	(3,2)	-
Bus 2	(2,1)	(3,2)	-
Bus 3	(2,3)	(2,2)	(3,2)

Tabla 2.1: Solución factible. Fuente [12].

2.1 Problemas relacionados

BEP se puede ver como una variante del Vehicle Routing Problem (VRP), aunque sus características únicas hacen necesario el uso de un modelo distinto que se adapte mejor a un escenario de evacuación. Según [2], la variante de VRP que más se aproxima al contexto de BEP es una mezcla de "Multi-Depot Vehicle Routing Problem" (MDVRP) con "Inter-Depot Routes" (MDVRPI) y "Split Delivery Vehicle Routing Problem" (SDVRP). A continuación, se explican en términos generales cada uno de ellos.

- **Vehicle Routing Problem (VRP):** Este problema está asociado a la administración del suministro de productos. Se tiene un grupo de clientes, cuyas demandas son conocidas, y uno o varios vehículos que llevan los suministros desde un depósito hasta los clientes. Se desea encontrar las rutas óptimas que deben seguir los vehículos para completar todas las entregas al menor costo posible [17]. Se consideran las siguientes restricciones [4]: (i) cada cliente es servido únicamente por uno de los vehículos, y en exactamente una visita; (ii) toda ruta comienza y termina en el depósito; (iii) la demanda total de los clientes de la ruta no excede la capacidad del vehículo asignado; (iv) la duración total de una ruta no debe exceder un valor pre-establecido.
- **Multi-Depot Vehicle Routing Problem (MDVRP):** Esta variante considera la existencia de varios depósitos. En su versión estándar, toda ruta debe comenzar y terminar en el mismo depósito. Cada cliente es asignado al depósito más cercano, por lo que cada cliente será visitado únicamente por vehículos que provengan del depósito asociado [4].
- **Multi-Depot Vehicle Routing Problem with Inter-Depot Routes (MDVRPI):** Además de considerar la existencia de varios depósitos, se incluye la posibilidad de que un vehículo pueda visitar un depósito cualquiera en su recorrido para recoger una nueva carga de suministros [4].

- **Split Delivery Vehicle Routing Problem (SDVRP):** Corresponde a una relajación del VRP en que se permite la "división de repartos", es decir, la demanda de un cliente puede ser satisfecha por más de un vehículo, removiendo así la restricción (i) mencionada anteriormente. Con esto, la demanda de un cliente puede ser mayor a la capacidad del vehículo, aunque se sigue manteniendo la restricción (iii) [6].

En [2], al problema resultante de la mezcla entre MDVRPI y SDVRP se le denomina "Split Delivery Multi-Depot Vehicle Routing Problem with Inter-Depot Routes" (SDMDVRPI), que según el autor no ha sido estudiado como tal. SDMDVRPI considera el siguiente escenario: se tiene un conjunto de clientes cuya demanda es conocida y requiere ser satisfecha por uno o más vehículos de igual capacidad. Éstos están situados en varios depósitos, y pueden satisfacer parcial o totalmente la demanda de un cliente, además de poder volver a cualquier otro depósito por más suministros.

Considerando ambos problemas, BEP y SDMDVRPI, es posible establecer un paralelo entre algunos de los componentes de cada uno [2]. Los clientes en el VRP tienen un rol similar a los nodos donde se recoge a los evacuados de BEP, siendo la demanda de cada cliente en VRP el equivalente al total de personas en el nodo que deben ser evacuadas. Los suministros con los que se satisface la demanda en VRP pueden asociarse a los cupos disponibles en cada bus en BEP, con los que se satisface la necesidad de evacuación. Los depósitos son las estaciones desde donde parten los buses (de hecho, a ambos se les denomina *depots* en la literatura), y a la vez los refugios son los lugares donde se deja a los evacuados.

BEP requiere de un modelo especial y distinto a los utilizados en las variantes de VRP, ya que tiene ciertas características específicas al problema detalladas en [2]:

- En VRP las rutas de los vehículos deben comenzar y terminar en el mismo depósito [17]. Esto puede entenderse como que un vehículo debe volver a su punto de partida al terminar el recorrido, lo que no se da en BEP, ya que en este último se considera que los buses se quedan en el último refugio que visitan. Esto se considera para asegurar el bienestar de conductores.
- En VRP, cuando se busca minimizar el tiempo total de entrega, ese tiempo se mide hasta que se satisface la última orden pendiente. Como los nodos de clientes se relacionan con los puntos de encuentro de BEP, lo anterior no consideraría el último traslado de evacuados hacia algún refugio. De esta forma, en BEP se incluye el costo de este último viaje para considerar la evacuación de todos los individuos.
- En VRP, cuando se busca minimizar el tiempo total de entrega de todos los vehículos, se considera la suma del tiempo que demora cada uno. Por otro lado, BEP busca minimizar la duración de la evacuación, es decir, se minimiza el tiempo de viaje del bus que más demora en completar la evacuación. Con esto, los tiempos de los otros

buses no se consideran, ya que ocurren en paralelo con respecto al tiempo del bus que más demora.

2.2 Complejidad del problema

En [11] se postula y demuestra que el BEP es de complejidad NP-completo, incluso si las distancias entre las estaciones y los puntos de encuentro es nula (es decir, si los buses se encuentran inicialmente en los mismos puntos donde se reúnen los evacuados), y las distancias entre los puntos de encuentro y un refugio específico son todas iguales entre sí (es decir, la distancia entre un punto de encuentro cualquiera i y un refugio j, es igual que la distancia de cualquier otro punto de encuentro i' al mismo refugio).

2.3 Variantes del problema

Si bien el problema es relativamente nuevo, se han llevado a cabo algunos estudios que proponen escenarios más complejos, incluyendo ciertos aspectos que puedan ayudar a generar modelos que se ajustan mejor a situaciones de la vida real. Algunos de ellos incorporan mejoras para poder planificar la evacuación en escenarios más ambiciosos.

En [11] se propone el Robust Bus Evacuation Problem (RBEP), una variante en la que el número de evacuados es desconocido y sólo se cuenta con ciertos escenarios probables. Una vez pasado un período de tiempo determinado, se conoce este valor con exactitud. En este problema, además de planificar las rutas y minimizar el tiempo de evacuación, se debe determinar en un momento específico si un bus debería partir con las personas que tiene hasta el momento o esperar a que lleguen más personas.

En [13] se presenta un "modelo integrado de localización y enrutamiento" para BEP, el Integrated Bus Evacuation Problem (IBEP). Además de planificar y minimizar las rutas que deberán seguir los buses, se incorpora la idea de que antes del proceso de evacuación es necesario definir la localización de los refugios a utilizar y de los puntos donde las personas serán recogidas. Para esto, se cuenta inicialmente con una lista de posibles refugios y puntos de encuentro disponibles, y el número total de evacuados se asume conocido. Si bien este modelo se adapta mejor a una situación real y ofrece un aporte en otro aspecto relacionado a la planificación de la evacuación, la complejidad y necesidad de procesamiento se incrementa.

Finalmente, en [10] se incorporan nuevos elementos al RBEP. Aquí se tiene que el número total de evacuados, los tiempos de traslado y el número de buses disponibles son inciertos al mismo tiempo. Esto se debe a que el escenario considera que la planificación de la evacuación

se hace a modo de preparación, mucho antes de que la misma sea necesaria, por lo que la información usada para los cálculos son sólo estimaciones. A causa de esto, la planificación puede o no adaptarse a la situación real en el momento en que se lleve a cabo la evacuación, y las rutas programadas podrían requerir de pequeñas modificaciones.

2.4 Resumen

En este capítulo se presentó la definición del Bus Evacuation Problem, junto con sus características y objetivos principales. Además, se presentó un ejemplo simple para ilustrar la estructura general del problema. Posteriormente, se hizo un análisis de las similitudes y diferencias con respecto al Vehicle Routing Problem que tienen características que, integrándolas, modelan una aproximación al BEP, y luego se comenta la complejidad del mismo. Finalmente, se presentó algunas de las variantes estudiadas desde la aparición del problema. Aquí se destaca la escasa investigación realizada hasta el momento de este tema específico, debido a que se trata de un problema prácticamente nuevo.

3 Estado del Arte

3.1 Trabajo relacionado

El problema de planificación de una evacuación mediante buses fue por primera vez planteado y definido formalmente por D. R. Bish en [2], en el año 2011. Aquí se discute la estrecha relación existente entre el BEP y una variante del Vehicle Routing Problem (VRP) denominada Split Delivery Multi-Depot Vehicle Routing Problem (SDMDVRPI), y se analizan las similitudes y diferencias entre ambos problemas. Los dos aspectos fundamentales en que difieren estos son: la función objetivo, ya que en el BEP generalmente se considera una estructura del tipo min-max, que involucra la minimización de la evacuación determinada por el bus con el mayor tiempo de viaje, mientras que en VRP se usa la función de costo; el otro aspecto es la estructura de la red formada, ya que los depósitos en el VRP se dividen en BEP en estaciones (desde donde salen los buses) y refugios (que reciben a los evacuados con capacidad limitada). Estos aspectos hacen que el BEP parezca más difícil de resolver, ya que es necesario generar las rutas, asignarlas a cada bus y seleccionar un refugio para cada una. Bish propone una representación matemática para el problema basada en programación entera mixta, y evalúa la utilización de la función min-max como función objetivo versus la función de costo usada en VRP, ambas adaptadas al BEP. Ambas funciones se definen de la siguiente manera:

- Función min-max: se minimiza la duración de la evacuación, que equivale a minimizar el costo de la ruta del bus con el mayor costo.
- Función de costo: se minimiza el costo total de ruta de cada vehículo, es decir, la suma de todas las rutas realizadas.

Para ejemplificar ambas funciones se considerará el escenario de la Figura 3.1. Se trata de un escenario con una estación, cuatro puntos de encuentro, un refugio y dos vehículos. En este caso se considera que ambos buses salen del refugio, que corresponde al nodo Y/S (estación y refugio a la vez). Los nodos P_i son los puntos de encuentro donde se recoge a los evacuados. Además, en cada nodo P_i hay un solo evacuado.

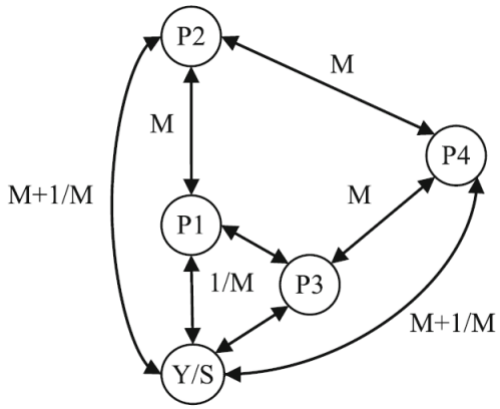


Figura 3.1: Grafo para ejemplificar las funciones min-max y de costo. Fuente [2].

Una solución óptima para la función de min-max según [2] sería asignar a un bus la ruta Y/S-P1-P2-Y/S y al segundo bus la ruta Y/S-P3-P4-Y/S, con una duración de evacuación (que es el valor optimizado) de $2M + 2/M$ y un costo total de $4M + 4/M$. Para calcular la duración de la evacuación se considera el recorrido de sólo uno de los buses (el que más demore, aunque en este caso ambos demoran lo mismo), y la función de costo considera la suma del tiempo de evacuación de todas las rutas de todos los buses. Por otro lado, una solución óptima usando el costo como función objetivo sería enviar a un bus por la ruta Y/S-P2-P4-Y/S y al otro a Y/S-P1-P3-Y/S, con un costo de $3M + 5/M$ (siendo éste el valor optimizado) y una duración de evacuación de $3M + 2/M$.

Para verificar el impacto de ambas funciones objetivo en los resultados se realizan pruebas a tres escenarios relativamente simples, E1P3R2B4, E1P4R2B4 y E1P5R3B6, usando el programa CPLEX [1]. Los resultados indicaron que independiente de la función utilizada, el tiempo de ejecución para resolver BEP aumenta significativamente al aumentar el tamaño del problema, y además sugieren que el objetivo min-max es mucho más complicado de resolver que el objetivo de costo, demorando un tiempo mucho mayor en los escenarios. El autor señala que para ambas funciones objetivo, cuando existen varias soluciones óptimas, algunas de ellas pueden incluir rutas innecesarias e indeseables (como buses que realizan rutas sin recoger pasajeros). Por este motivo, se propone una mejora para ambas funciones incorporando un ordenamiento lexicográfico. Con estas nuevas funciones se realizaron pruebas sobre los mismos escenarios utilizados anteriormente. La función objetivo de costo lexicográfica, en general, no disminuyó el tiempo de ejecución, a diferencia de la función de min-max lexicográfica, que permitió incluso resolver la instancia de mayor tamaño, que en la prueba anterior había cumplido con el límite de tiempo. Por otro lado, con esta última función se obtuvieron costos totales menores, aunque la duración de evacuación se mantuvo igual o mejoró muy poco.

[1] Software para la resolución de programación lineal, disponible en: http://www.cplex.com/

Posteriormente, en el mismo paper se propone una representación adicional similar a la anterior, pero en este caso se considera que las rutas ya han sido generadas y son datos de entrada para el modelo, en lugar de generar las rutas como lo hace el primer modelo. Además, en lugar de considerar los costos de los arcos, se consideran los costos de las rutas. Finalmente se proponen dos heurísticas para resolver instancias de tamaños más realistas. La primera heurística construye rápidamente una solución inicial factible, que luego se repara intercambiando y reasignando rutas. La segunda utiliza la solución de la primera heurística, que relajan el primer modelo matemático y el segundo modelo para conseguir mejores resultados. Ambas heurísticas se testean con los escenarios pequeños usados anteriormente, obteniendo muy buenos tiempos de ejecución y resultados óptimos o muy cercanos al óptimo. Con escenarios mucho mayores, siendo el más grande E2P32R5B18, no se lograron obtener óptimos con ninguna de las dos heurísticas, pero en general la segunda heurística entrega mejores resultados que la primera.

En [11], Mark Goerigk y Bob Grün consideran una versión extendida del BEP denominada Robust Bus Evacuation Problem (RBEP), en la cual se asume que se desconoce el número exacto de individuos dentro de la zona de riesgo. Sólo se cuenta con un conjunto discreto de escenarios probables, aunque después de un tiempo transcurrida la evacuación se sabe cuál fue el escenario que aconteció. El problema es decidir para cada bus, si es mejor evacuar inmediatamente considerando en número incierto de evacuados (buses *here-and-now*), o esperar el tiempo que sea necesario cada vez que está en un punto de encuentro, hasta que el número exacto de evacuados sea conocido (buses *wait-and-see*). El primer caso tiene como ventaja que el bus no necesariamente debe esperar hasta que lleguen todos los individuos al punto de encuentro. Además de lo anterior, se debe encontrar una planificación que minimice el máximo tiempo de viaje entre todos los buses, tal que todos los evacuados lleguen a los refugios. En este artículo, se propone una representación matemática basada en programación lineal entera mixta (MIP) para el RBEP.

Se usa como base el modelo propuesto en [2], y se añade el supuesto de que cada bus recoge exactamente una cantidad de personas igual a su capacidad cuando visita un punto de encuentro, lo que permite considerar que la demanda en los nodos de encuentro y la capacidad de los refugios son múltiplos de la capacidad de los buses. Posteriormente se adapta el modelo anterior, para que incluya las nuevas condiciones del problema acerca de la información incierta y si los buses deben irse de inmediato o esperar. Otros métodos propuestos para la resolución del problema son Búsqueda Lineal, para lo cual se propone un modelo MIP reducido basado en el modelo MIP para RBEP; y Tabu Search, para lo cual se propone una representación basada en tablas (matrices) que contengan listas de rutas, y se presenta un framework para aplicar esta metaheurística al problema. Además, se proponen dos métodos para encontrar límites inferiores (*lower bounds)* para la función objetivo. Finalmente, se comparan las metodologías propuestas usando instancias generadas de forma aleatoria, y un escenario real para simular una evacuación en la ciudad de Kaiserslautern, Alemania. Esta última considera cuatro puntos de encuentro, cinco refugios y una estación. Para cada instancia, inicialmente

se resolvió el modelo MIP usando CPLEX, se usa la búsqueda lineal con CPLEX, y finalmente se aplica Tabú Search. De los resultados obtenidos con las instancias aleatorias se pudo concluir que para instancias pequeñas todos los enfoques rinden casi igualmente bien, pero al ir aumentando la complejidad, con el modelo MIP no se logra resolver las instancias, y posteriormente tampoco lo hace la búsqueda lineal, demostrando que Tabu Search sería un enfoque adecuado para instancias más complejas debido a su buena escalabilidad. Para el escenario real se encontró un patrón similar, en donde Tabu Search y la búsqueda lineal con CPLEX funcionaron mejor que MIP con CPLEX.

En [12] se considera casi el mismo escenario descrito en [2], y se presentan varios enfoques para construir soluciones factibles, encontrando límites inferiores y superiores (*lower y upper bounds*) para el tiempo de evacuación. Para encontrar límites superiores, se proponen cuatro algoritmos greedy, tres de los cuales asumen que para cada bus ya se tiene una planificación de antemano. Por otro lado, para encontrar los límites inferiores se proponen tres algoritmos: el primero, $LB1$, subestima el costo de cada viaje realizado, el segundo, $LB2$, considera una versión de BEP donde el objetivo es minimizar la suma de los tiempos de viaje, y el tercero, $LB3$, se basa en escenarios reales, en donde los puntos de encuentro se encuentran en un entorno cerrado y los refugios están generalmente lejos, por lo que se contraen todos los puntos de encuentro en un único nodo.

Los límites encontrados con los distintos algoritmos propuestos, son integrados a un framework basado en Branch and Bound, que posteriormente se utiliza para encontrar soluciones óptimas. Para la construcción de este framework, los autores proponen algunas reglas de ramificación y técnicas de poda de nodos que permiten mejorar los resultados. Entre las técnicas de ramificación se tiene: $B1$, la regla de ramificado completo, creando un nodo para cada bus, punto de encuentro y refugio con demanda/capacidad disponible; $B2$, la regla de primeros buses primero, donde se ramifican primero los buses con menor índice; y $B3$, la regla del offset mínimo, que ramifica los buses con el menor offset, que corresponde al tiempo que un bus toma en recorrer los viajes ya asignados.

Se realizan varios experimentos para comparar los límites inferiores, los límites superiores y las reglas de ramificación propuestas. Se usa el framework programado, y el IP solver CPLEX para comparaciones, con 90 instancias generadas de manera aleatoria, donde las más pequeñas tienen dos refugios, dos puntos de encuentro y dos buses; y las más grandes tienen diez refugios y puntos de encuentro, y seis buses. En todas las instancias se considera una estación.

En los experimento para evaluar los límites inferiores, interesa analizar la relación calidad, tiempo y compromiso entre ambos, y compararlos con los tiempos obtenidos con CPLEX. En general, se tiene que los tres algoritmos tienen mejor rendimiento que el IP solver, resultando que $LB2$ es el que tuvo los mejores resultados, seguido por $LB1$, y $LB3$ en último lugar. En el segundo experimento se evalúan las diferencias entre los límites superiores propuestos. De los resultados obtenidos, se concluyó que el segundo algoritmo de límite superior, donde se

calcula un conjunto de rutas que luego se asignan a los buses considerando primero las de menor costo, tiende a producir mejores límites. En el último experimento, se comparan las reglas de ramificación y los enfoques de poda. Entre las reglas de ramificación, se encontró que la regla $B2$ permitió encontrar una mayor cantidad de soluciones óptimas que las otras dos, aunque en promedio la regla $B3$ resulta en mejores brechas (relación *límite_sup/límite_inf*) que la regla $B2$. Finalmente, al comparar las reglas de poda, los resultados no son lo suficientemente concluyentes como para determinar que una es mejor que la otra, ya que se nota que ambas generan un impacto en las soluciones. En general, los resultados muestran que los tiempos de respuesta de la herramienta propuesta mejora significativamente los resultados obtenidos por el solver comercial CPLEX, siendo capaz de resolver instancias de tamaño realista en tiempos más razonables.

En [13] se considera además de determinar las rutas de los buses, el problema adicional de determinar dónde estarán ubicados los puntos de encuentro y los refugios. Esta variante se denomina Integrated Bus Evacuation Problem (IBEP). Nuevamente, se presenta un modelo basado en programación lineal entera que se adapta a las condiciones del problema, y se propone un algoritmo Branch-Cut-and-Price que resuelve el modelo propuesto de forma exacta, a diferencia de otros enfoques que encuentran soluciones aproximadas mediante heurísticas. No se considera incertidumbre en este caso. Se generaron dos conjuntos de instancias, $\mathscr{I}_1$ con valores aleatorios y $\mathscr{I}_2$ con distancias euclidianas, donde se consideran cinco zonas concéntricas, y los puntos de encuentro se sitúan en la primera zona más cercana al centro y los refugios se distribuyen en las demás zonas. Para cada instancia se resolvió el modelo integrado usando CPLEX y se comparó con el enfoque propuesto, branch-cut-and-price (BCP).

Los resultados obtenidos mostraron que el BCP implementado logró encontrar un gap mucho menor a la obtenida con CPLEX, y resuelve una mayor cantidad de instancias de manera óptima. El rendimiento de BCP en ambos tipos de instancia $\mathscr{I}_1$ y $\mathscr{I}_2$ es prácticamente el mismo. Sin embargo, el enfoque secuencial resultó ser mejor en instancias de tipo $\mathscr{I}_2$. Posteriormente, se usó el mismo escenario real que el propuesto en [11] situado en la ciudad de Kaiserslautern. Se consideran en total 14 posibles puntos de encuentro, 23 posibles refugios y 3 buses. Con el algoritmo BCP se pudo calcular tiempos estimados de evacuación para escenarios con tres o cuatro puntos de encuentro, y de tres a seis refugios, del total disponible. Los tiempos de evacuación calculados en minutos se muestran en la Tabla 3.1, donde N^{RE} es el número de refugios, N^{PE} es el número de puntos de encuentro, y en la relación a/b, a es el límite inferior y b el límite superior de cada escenario. Por ejemplo, para $N^{PE} = 3$, si se habilitan 6 refugios en lugar de 3, se disminuiría en 10 minutos el tiempo de evacuación.

		N^{RE}			
		3	4	5	6
N^{PE}	3	51/51	47/47	42/42	41/41
	4	45/45	41/41	38/41	38/38

Tabla 3.1: Valores obtenidos para Kaiserslautern según N^{PE} y N^{RE}. Fuente [13].

Finalmente, se pudo concluir que los problemas de locación y enrutamiento integrados son efectivamente más difíciles de resolver, pero tienen el potencial de entregar soluciones con mejor valor objetivo que las encontradas por un enfoque secuencial. Los resultados mostraron que el algoritmo BCP propuesto es útil para resolver instancias de tamaño realista de manera óptima, y que supera por mucho al rendimiento del IP solver utilizado.

En el año 2014, en [9] se vuelven a incorporar más elementos al escenario de una evacuación en un área urbana con el objetivo de acercar más el problema a una situación más realista. En este caso se busca determinar qué refugios serán efectivamente usados para la evacuación, de un conjunto inicial de refugios posibles, planificar las rutas de evacuación para la evacuación de personas tránsito-dependientes mediante transporte público, y evaluar cómo interactúa el transporte público con el privado durante la evacuación. Además, se considera la seguridad de los evacuados durante la evacuación, como por ejemplo, priorizando rutas seguras sin riesgos de colapso de estructuras. Se incluye además la posibilidad de una evacuación usando vehículo particular, lo que involucra ciertas consideraciones adicionales. Por ejemplo, se considera que los refugios deben tener capacidad de estacionamiento para particulares, además de la capacidad para albergar evacuados. Además, se asume que los evacuados pueden comenzar su evacuación en momentos diferentes, que el encargado de la planificación tiene control total sobre el transporte público y particular, y que los vehículos pueden generar congestión. A esta variante del BEP se le denomina Comprehensive Evacuation Problem (CEP), y tiene complejidad NP-hard [9]. Para esto se propone un modelo de optimización multi-objetivo, en el que se busca minimizar el tiempo de evacuación, el riesgo de la evacuación y el número de refugios utilizados (que equivale a un menor costo resultante de la evacuación).

Para la resolución del CEP se desarrolla un algoritmo genético basado en un ordenamiento no-dominado (NSGA-II [16]) capaz de generar soluciones factibles para instancias de tamaño real. La solución obtenida mediante el algoritmo genético, posteriormente se mejora con una búsqueda local dentro del vecindario usando el mismo operador de mutación, que se ejecuta hasta encontrar la mejor solución o cumplir una cierta cantidad de iteraciones. Para evaluar el algoritmo se usan instancias que modelan evacuaciones reales: una corresponde a una evacuación circular, con la amenaza en el centro de la ciudad de Kaiserslautern, Alemania, y el riesgo inversamente proporcional a la distancia hasta el centro; la segunda es una evacuación por riesgo de tsunami en la ciudad de Nice, Francia. La primera instancia tiene un total de 13282 nodos y 32463 arcos, y la segunda un total de 6237 nodos y 13209 arcos. Se desea

saber qué complejidad debe tener una instancia para que pueda ser considerada un “escenario real”, o al menos pueda ser extendida a una instancia más completa. Para esto se evalúan distintas complejidades de ambas instancias, reduciendo la cantidad de nodos a 4000 y 8000 la primera, y 2000 y 4000 en la segunda. Los autores detallan cómo una instancia de CEP puede ser comprimida a una instancia de CEP reducida. Luego, una solución producida para una instancia comprimida puede ser extendida a la instancia más detallada. Esta técnica permite obtener resultados en menos tiempo y puede ayudar a mejorar su calidad. Se debe tomar en cuenta que reducir demasiado las instancias impide obtener resultados significativos.

En [5], se presenta un survey de las técnicas propuestas para resolver problemas de evacuación y que han sido modeladas como flujo sobre redes. Especialmente se evalúa cómo se interrelacionan los resultados de los modelos mencionados. En este caso, se consideran escenarios de evacuación a pequeña escala, como el desalojo de edificios y estadios, así como escenarios de gran escala como lo es la evacuación de áreas urbanas mediante transporte privado y público. El autor concluye que las características específicas de cada modelo hacen extremadamente difícil una unificación e implementación de una solución integrada. Por otro lado, los distintos modelos tienen ventajas y desventajas, por lo que no hay consenso respecto de cuál es mejor. Cabe destacar que en el contexto de evacuación a gran escala, se comenta que la mayoría de los modelos propuestos fallan en capturar el comportamiento de los evacuados, ya que asumen que la planificación puede llevarse a cabo con total eficiencia y control de las acciones de la población en riesgo, así como también se asume que los evacuados seguirán correctamente las instrucciones, consiguiendo el tiempo de evacuación óptimo. Por este motivo, aún no son aptos para aplicar correctamente en una situación real de emergencia. Finalmente, se menciona que se han propuesto varias heurísticas para abordar evacuaciones en grandes redes (como las de evacuaciones a gran escala) y poder estimar soluciones óptimas, pero hay pocas aproximaciones que estudien los errores resultantes. Así, queda una brecha para poder estudiar cómo estas heurísticas permiten mejorar las soluciones.

3.2 Resumen

En este capítulo se realizó una revisión de los avances más importantes que se han logrado en la resolución de BEP. Como se trata de un problema reciente, aún se encuentra poco desarrollado. Las investigaciones se han enfocado en resolver principalmente las variantes del problema, que incorporan factores adicionales para poder adaptar mejor el problema a una situación real. Esto genera cada vez escenarios más complejos, lo que motiva la proposición de nuevos modelos, heurísticas y diversas técnicas para poder resolver el problema. En general, se puede apreciar que las heurísticas que se han propuesto en las diversas investigaciones han permitido mejorar considerablemente los resultados, independiente de la técnica utilizada y del modelo estudiado. La escasa investigación en esta área deja abiertas las posibilidades de futuras aproximaciones, ya sea estudiando el problema en su versión más simple, o alguna de las diversas variantes más cercanas a la realidad, incluyendo la posibilidad de incluir nuevos

factores tales como el comportamiento humano.

En el siguiente capítulo, se muestra la versión del problema de evacuación que se considera en esta memoria, junto con una descripción detallada del algoritmo basado en GRASP implementado para su resolución.

4 Descripción del Algoritmo

En este capítulo se presenta el modelo considerado para abordar el Bus Evacuation Problem. Además, se propone un algoritmo basado en GRASP para su resolución, se describe la estructura general del algoritmo, y, posteriormente, se presentan cada una de sus componentes con mayor detalle, explicando además las consideraciones de diseño tomadas en cuenta.

4.1 Modelo matemático

Tal como ya se ha señalado, el problema ha sido estudiado considerando diversos factores y características adicionales, por lo que cada variante requiere, en general, el uso de un modelo específico para cada situación. El problema abordado en esta memoria corresponde a una simplificación del modelo original propuesto en [2].

4.1.1 Modelo de Programación Entera Mixta para BEP [2]

A continuación se presenta el modelo matemático propuesto para el problema original BEP en [2], considerando la siguiente notación:

Se considera una red (N,A), donde N y A denotan respectivamente el conjunto de nodos y arcos. N se compone de tres subconjuntos de nodos:

- Y, un conjunto de nodos de estaciones donde los buses se encuentran al comenzar la evacuación. Son los puntos de partida.
- P, un conjunto de nodos de demanda, cada uno de ellos representa un punto de encuentro que sirve para reunir a los evacuados, desde donde serán recogidos por los buses.
- S, un conjunto de nodos que representa a los refugios.

V representa el conjunto de buses disponibles, teniendo cada uno una capacidad Q. Los buses se dividen en subconjuntos V_y, $y \in Y$, donde cada bus $v \in V_y$ es inicialmente ubicado en la estación y.

El nodo p tiene una demanda (cantidad de personas esperando a ser evacuadas) D_p, $p \in P$, y el refugio s tiene un capacidad C_s, $s \in S$.

Cada arco (i, j) tiene un costo de viaje positivo de τ_{ij} con $(i, j) \in A$. Para el conjunto de arcos, se adoptan tres supuestos comúnmente realizados en la literatura VRP:

1. El costo de viaje es proporcional a la duración del viaje y la distancia; por lo tanto, se usan los términos "costo", "tiempo" y "distancia" indistintamente.
2. Todos los costos de la red son simétricos, es decir, $\tau_{ij} = \tau_{ji}$
3. Todos los costos de viaje satisfacen la desigualdad triangular:

   $\tau_{ij} \leq \tau_{ik} + \tau_{kj}, \forall (i, j), (i, k), (k, j) \in A$.
4. El objetivo es programar la ruta de los buses, para reducir al mínimo la duración de la evacuación, al mismo tiempo de satisfacer toda la demanda y sin violar restricciones tanto de refugio como de capacidad de buses.

Otra consideración es que los buses al terminar la evacuación no deben volver a las estaciones, considerando el riesgo que esto significaría para el conductor.

- Variables de decisión:
  - $x_{ij}^{vt} = \begin{cases} 1 & \text{Si en el viaje } t \text{ el bus } v \text{ pasa por el arco } (i, j) \\ 0 & \text{En caso contrario} \end{cases}$
  - b_j^{vt} = Número de evacuados del nodo j que serán transportados por el bus v después el viaje t, $\forall j \in N, v \in V, t = 1, ..., T$ (si j es un refugio, el bus queda liberado de pasajeros).
  - T_{evac} = La duración de la evacuación.
- Función Objetivo:
  - Mín T_{evac} = Minimizar el tiempo de la evacuación. Se minimiza el tiempo del bus que más demora en completar la evacuación (función Min-Max).
- Restricciones:
  1. El tiempo de evacuación resultante debe ser mayor o igual al costo máximo de cualquier viaje.

     $$T_{evac} \geq \sum_{(i,j) \in A} \sum_{t=1}^{T} \tau_{ij} x_{ij}^{vt}, \forall v \in V$$

     Observación: Para el objetivo Min-Max hay un tamaño óptimo flota umbral. Aumentar el tamaño de la flota más allá de este umbral no afecta la optimización.
  2. Flujo de balance para nodos de demanda: se asegura que un bus que viaja al nodo j en el viaje t deja el nodo j en el viaje $t + 1$.

     $$\sum_{i:(i,p) \in A} x_{ip}^{vt} = \sum_{k:(p,k) \in A} x_{pk}^{v(t+1)}, \forall p \in P, v \in V, t = 1, ..., T - 1$$

3. Flujo de balance para nodo de refugio: no se requiere que el bus deje el refugio s , es decir, en su último viaje el bus puede quedarse en el refugio debido al peligro que corre el conductor de volver a la zona de riesgo.

$$\sum_{s:(s,j)\in A} x_{sj}^{vt} \geq \sum_{k:(s,k)\in A} x_{sk}^{v(t+1)}, \forall s \in S, v \in V, t = 1, ..., T-1$$

4. Un bus solamente puede hacer un viaje a la vez.

$$\sum_{(i,j)\in A} x_{ij}^{vt} \leq 1, \forall v \in V, t = 1, ..., T$$

5. Cada bus debe partir desde la estación en su primer viaje.

$$x_{yj}^{v1} = 1, \forall y \in Y,\ j : (y, j) \in A, v \in V_i$$

6. Si un bus no sale de su estación al comienzo de la evacuación, no sale tampoco en viajes posteriores.

$$x_{yj}^{vt} = 0, \forall y \in Y, j : (y, j) \in A,\ v \in V, t = 2, ..., T.$$

7. No se permite que el último viaje de un bus termine en un nodo de demanda.

$$x_{ip}^{vT} = 0, \forall p \in P, i : (i, p) \in A, v \in V$$

8. Un bus sólo puede recoger a los evacuados desde el nodo j si ha viajado al nodo j.

$$b_j^{vt} \leq \sum Q x_{ij}^{vt}, \forall j \in N,\ v \in V, t = 1, ..., T$$

9. Capacidad del bus.

$$0 \leq \sum_{j\in P} \sum_{l=1}^{t} b_j^{vl} - \sum_{k\in S} \sum_{l=1}^{t} b_k^{vl} \leq Q, \forall v \in V, t = 1, ..., T$$

10. Capacidad del refugio.

$$\sum_{m\in V} \sum_{t=1}^{T} b_j^{mt} \leq C_j, \forall j \in S$$

11. Todos los evacuados deben ser recogidos de los nodos de demanda.

$$\sum_{m \in V} \sum_{t=1}^{T} b_j^{mt} = D_j, \forall j \in P$$

12. Todos los evacuados deben ser llevados a los refugios.

$$\sum_{p \in P} \sum_{t=1}^{T} b_p^{vt} = \sum_{s \in S} \sum_{t=1}^{T} b_s^{vt}, \forall v \in V$$

13. Esta variable tiene sólo valores binarios.

$$x_{ij}^{vt} \in \{0,1\}, \forall (i,j) \in A,\ v \in V, t = 1, ..., T$$

14. Esta variable tiene sólo valores positivos.

$$b_j^{vt} \geq 0, \forall j \in N,\ v \in V, t = 1, ..., T$$

4.2 GRASP

Para la resolución del Bus Evacuation Problem se implementó el algoritmo Greedy Randomized Adaptive Search Procedure (GRASP). Éste corresponde a una metaheurística iterativa y con múltiples repeticiones, propuesta por Feo y Resende [8]. Consiste principalmente en dos fases, una de construcción y otra de post-procesamiento. Además, es posible agregar antes de la construcción una fase de preprocesamiento, para fijar subestructuras que, de antemano, se sepa pertenecen a soluciones óptimas.

En cada iteración de GRASP se construye una solución factible, usando un algoritmo Greedy aleatorizado. Esto se refiere a que, para construir la solución mediante la función miope, ésta considera varias opciones en orden decreciente respecto al beneficio que aportan a la solución, en lugar de considerar sólo la opción de mayor beneficio como lo hace un algoritmo Greedy común. Una vez generada la solución completa, en la fase de post-procesamiento se le aplica algún procedimiento de búsqueda local, con el objetivo de encontrar una solución óptima local.

El Algoritmo 1 muestra una visión general de GRASP para un problema de minimización.

Cabe señalar que como en este problema no se encontraron sub-estructuras que pudieran ser parte de una solución óptima, en la implementación propuesta no se efectúa la fase de preprocesamiento.

Algoritmo 1: Pseudocódigo general para un GRASP

```
Inicializar();
MejorSolucion ← ∅;
MejorCosto ← ∞;
while iteracion < maxNumIteraciones do
    Preprocesamiento();
    SolucionActual ← AlgoritmoConstructivo();
    SolucionActual ← AlgoritmoBusquedaLocal(SolucionActual);
    if Costo(SolucionActual)< MejorCosto then
        MejorSolucion ← SolucionActual;
    end
end
return MejorSolucion
```

4.3 Algoritmo GRASP Propuesto

A continuación, se presentan los componentes del algoritmo implementado. Inicialmente se describe la representación utilizada para una solución, y luego se describen los aspectos considerados para el desarrollo de cada una de las fases que conforman el algoritmo GRASP.

4.3.1 Representación

En este problema, se busca conocer la ruta completa que debe realizar cada bus, desde que comienza la evacuación en su estación hasta que termina en un refugio cualquiera. Por este motivo, la solución se representa usando un arreglo, donde cada índice corresponde a uno de los buses de la evacuación. Cada casilla apunta a una lista de viajes, que compone la ruta total que realiza el bus respectivo.

Por otro lado, cada viaje se representa como un par (i, j), donde i corresponde al nodo de origen del viaje, y j es el nodo de llegada. Para facilitar la manipulación de la solución, la representación se compone de dos tipos de viajes: un viaje inicial, que va desde una estación a un punto de encuentro y donde no ocurre traslado de pasajeros; y uno o varios viajes de traslado, que parten en un punto de encuentro y llegan a un refugio. En una sub-ruta del tipo $(PE_1, R_1), (PE_2, R_2)$, donde PE y R son puntos de encuentro y refugios respectivamente, se tiene de manera implícita que el bus realizó un recorrido adicional para retornar desde el refugio R_2 al punto de encuentro PE_2. A este último se le denominará "viaje de retorno", aunque no tiene la misma estructura que los dos primeros. Sólo se presenta implicitamente al sumar su distancia asociada a la distancia total recorrida por un bus.

En la Figura4.1 se puede ver de forma gráfica la representación de la solución para una instancia con 3 buses, 1 estación, 4 puntos de encuentro y 2 refugios.

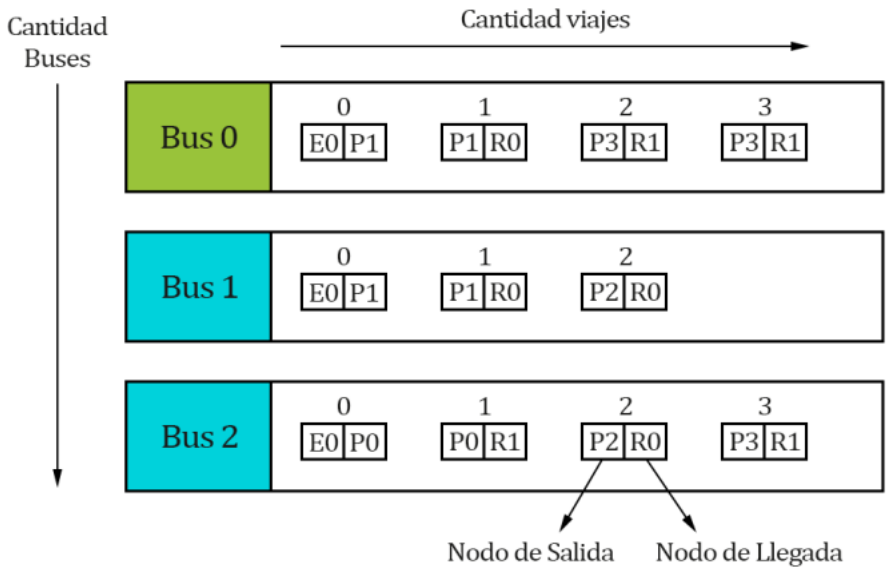


Figura 4.1: Representación de una solución. Fuente: Elab. Propia.

Esta representación permite conocer y evaluar rápidamente el recorrido asignado a un bus, así como manipular con mayor facilidad los viajes para poder hacer reparaciones a soluciones candidatas. Además, solo trabaja con soluciones factibles, incluyéndose en el diseño varias de las restricciones asociadas al problema.

4.3.2 Inicialización

Como se mencionó anteriormente, encontrar una solución para este problema consiste, básicamente, en encontrar un recorrido para cada bus, cumpliendo con una serie de restricciones asociadas a la evacuación. Un recorrido está compuesto de partes denominadas "viajes" o "trips", que corresponde a un viaje de un punto a otro. Así, generar el recorrido de un bus consiste en realizar estos viajes cortos iterativamente, lo que se traduce en ir añadiendo viajes a una lista con el recorrido completo del bus.

La fase de inicialización se realiza una sola vez luego de leer los datos de la instancia, y consiste principalmente en agregar la mayor cantidad de información posible a las estructuras del problema. Primero se identifican los viajes posibles a partir de cada nodo, es decir, las conexiones existentes en el grafo, y se guarda un objeto Trip por cada conexión. Se hace distinción entre un Trip inicial, que va desde una estación a un punto de encuentro, y un Trip que puede ir desde un punto de encuentro a un refugio. Luego, a cada nodo se le asigna una lista de los viajes que es posible realizar desde este nodo, ordenada según la distancia asociada al viaje. A un nodo refugio también se le agrega esta lista, pero con trips que llegan a él.

4.3.3 Fase Constructiva

La fase constructiva del algoritmo corresponde a un Greedy con aleatorización. Como la función objetivo corresponde a minimizar la distancia total recorrida por el bus con la mayor ruta (minimización de la ruta máxima), la función miope utilizada consiste en el vecino más cercano, es decir, elegir el viaje que sea más corto de realizar desde el nodo actual. Esto apunta en la dirección de reducir el largo de las rutas de todos los buses, y obtener así un mejor valor de la función objetivo. Para añadir diversidad a la solución inicial, en lugar de elegir el viaje más corto, se elige un viaje desde una porción de la lista ordenada de viajes generada en la fase de inicialización, y que corresponde a la lista restringida de candidatos LRC del algoritmo GRASP. El Algoritmo2 muestra el pseudocódigo de la fase de construcción.

Al comenzar una iteración de GRASP, todos los buses se encuentran en sus respectivas estaciones. Todos los evacuados se encuentran en los puntos de encuentro, y los refugios se encuentran vacíos. Para realizar el primer viaje, se elige uno factible desde el nodo donde se encuentra el bus usando la LRC. Como la lista de viajes factibles se encuentra ordenada, la lista de candidatos contiene los k viajes más cortos que se pueden realizar desde ese nodo. Se elige un viaje al azar, se añade a la lista de viajes realizados y se actualiza la posición del bus al nodo de llegada. Una vez que el bus "llega" al punto de encuentro, la demanda de ese nodo disminuye en una cantidad igual a la capacidad del bus. Esto último se debe a que, en el modelo utilizado, se considera que la cantidad de personas en cada punto de encuentro es un múltiplo de la capacidad de los buses.

Cuando el bus se encuentra en un punto de encuentro, se elige nuevamente un viaje desde la lista de candidatos, y se aplica el viaje escogido. Se añade el viaje realizado al recorrido del bus, se actualiza su posición y se reduce la capacidad disponible del refugio de llegada. Finalmente, cuando el bus se encuentra en un refugio se elige un viaje factible desde la lista y se actualiza la posición del bus, pero no se agrega el viaje al recorrido del bus de manera directa.

Cada vez que un bus hace un viaje, la demanda/capacidad se actualiza de inmediato al llegar al punto de encuentro/refugio, lo que permite verificar inmediatamente si aún queda demanda o capacidad en el nodo. De esta manera, los buses a los que aún no se ha asignado un viaje, no considerarán los viajes que llegan a nodos sin demanda/capacidad dentro de su lista de candidatos.

Este procedimiento se repite hasta que ya no queden más personas en los puntos de encuentro que deban ser evacuadas.

4.3.4 Post-Procesamiento

Para mejorar la calidad de la solución generada en la fase de construcción se ha implementado un Hill Climbing con alguna mejora como algoritmo de búsqueda local.

Las componentes del Hill Climbing son las siguientes:

Algoritmo 2: Pseudocódigo Fase de Construcción GRASP

Input: Una instancia I del problema, tamaño LRC $\alpha \in [0,1]$

```
PerformInitialTrip();
while evacuacion no completada do
    foreach bus in V do
        nodoActual ← GetPosicionActual(bus);
        listaTrips ← GetListaTripsFactiblesDesdeNodo(nodoActual);
        largoRCL ← α∗Largo(listaTrips);
        trip ← SeleccionarTripAleatorio(listaTrips, 0, largoRCL);
        if nodoActual ∈ P then
            AñadirTrip(bus, trip);
            nodoActual ← NodoLlegada(trip);
            ActualizarPosicion(bus, nodoActual);
            capacidad ←DisminuirCapacidad(nodoActual);
            if capacidad == 0 then
                foreach p in P do
                    EliminarTripPorNodoLlegada(p, nodoActual);
                end
            end
        else                                              /*nodoActual ∈ S*/
            nodoActual ← NodoLlegada(trip);
            ActualizarPosicion(bus, nodoActual);
            demanda ←DisminuirDemanda(nodoActual);
            if demanda == 0 then
                foreach s in S do
                    EliminarTripPorNodoSalida(s, nodoActual);
                end
            end
        end
    end
end
```

- Representación de la solución: Se usa la misma que en la fase de construcción.
- Criterio de aceptación: Alguna mejora.
- Función de Evaluación: Seleccionar la solución vecina que permita reducir la distancia total del bus que se demora más en completar su recorrido.

Para generar los vecindarios se definió un movimiento *shift*, que se muestra en el Algoritmo 3.

El movimiento definido consta de los siguientes pasos:

Algoritmo 3: Pseudocódigo movimiento shift para Hill Climbing

Input: Una solución C del problema
Output: Valor booleano indicando éxito o fracaso en la operación

```
MejorValorFO ← ObtenerValorFO(C);
bMax ← ObtenerBusRutaMasLarga(C);
foreach t in ObtenerListaTripsRealizados(bMax) do
    EliminarTripDeBus(t, bMax);
    foreach b in V; b ≠ bMax; V ∈ C do
        foreach pos in ObtenerListaTripsRealizados(b) do
            InsertarTripEnPosicion(t, pos);
            if ObtenerValorFO(C) < MejorValorFO then
                return true
            else
                DeshacerInsercionTrip(t, pos);
            end
        end
    end
    DeshacerEliminacionTrip(t, bMax);
end
return false
```

1. Identificar el bus con la ruta más larga.
2. Tomar un viaje de traslado (de punto de encuentro a refugio) del bus con la ruta más larga, eliminarlo de la lista de viajes realizados por ese bus.
3. Insertar el viaje en el recorrido de otro bus.

Para generar el vecindario se debe tomar cada uno de los viajes de traslado del bus con la ruta más larga, e intentar insertarlos en todas las posiciones posibles en las rutas del resto de los buses. Este movimiento genera un vecindario factible, ya que si un bus deja de realizar un viaje desde un punto de encuentro A a un refugio B, las personas que se hayan trasladado en ese viaje simplemente serán trasladadas en el momento en que otro bus realice el mismo viaje.

Este procedimiento se repite hasta que se cumpla una cantidad de iteraciones especificadas, o hasta estancarse en un óptimo local.

4.4 Conclusiones

En este capítulo se propuso un algoritmo basado en Greedy Randomized Adaptive Search, para resolver el Bus Evacuation Problem. En primer lugar, se presentó el modelo matemático

considerado para representar el problema. Posteriormente, se presentan las distintas componentes del algoritmo, que corresponden a una fase de inicialización, una fase constructiva y una fase de post-procesamiento. Además, se describió la representación a usar para una solución del problema. Esta permite mantener de forma ordenada un registro de las rutas asignadas a cada bus, junto con la información asociada a cada recorrido (nodos de llegada, de salida, distancias entre nodos, etc.). Por otro lado, permite realizar movimientos para modificar la solución manteniendo siempre su factibilidad, con lo que se evita la necesidad de reparar una solución infactible. La fase constructiva corresponde a un algoritmo Greedy no determinista, en que se considera una lista restringida de candidatos que se va acortando a medida que se construye la solución, para una mayor eficiencia en la ejecución. Finalmente, el movimiento propuesto para la fase de post-procesamiento permite modificar la solución, con el objetivo de mejorar su calidad alivianando la carga del bus que tiene el recorrido más largo asignado.

En el siguiente capítulo, se presenta el detalle de las instancias que han sido diseñadas para la resolución del problema, incluyendo benchmark generado de forma aleatoria, así como uno basado en un caso real.

5 Instancias

Para la ejecución del algoritmo implementado, se consideraron dos grupos de instancias de distinto tipo:

- Grupo 1: Instancias generadas aleatoriamente
- Grupo 2: Instancias de tamaño real, generadas en base al Gran Incendio de Valparaíso.

Ambos tipos comparten las siguientes características fundamentales:

1. El número de personas en cada punto de encuentro es múltiplo de la capacidad de los buses [12].
2. Las capacidades de los refugios son múltiplos de la capacidad de los buses [12].
3. Los puntos de encuentro no están conectados entre sí.
4. Los refugios no están conectados entre sí.
5. Las distancias de un nodo a otro son las mismas de ida y de vuelta. Es decir, la distancia del punto de encuentro i al refugio j es igual a la distancia del refugio j al punto de encuentro i.
6. La capacidad total de los refugios siempre es mayor o igual a la demanda total en los puntos de encuentro.
7. La cantidad de buses en cada estación se distribuye equitativamente.

5.1 Grupo 1: Instancias aleatorias

Las instancias generadas aleatoriamente se hicieron en base a [12]. Se generaron 9 sets de instancias de tamaño creciente, cada uno compuesto por 10 instancias. En total, se utilizaron 90 instancias generadas aleatoriamente. La capacidad de los buses en todas las instancias es de 1. Los valores de las distancias entre nodos y las capacidades de los refugios corresponden a valores en el rango $\{1,...,10\}$, y las demandas en los puntos de encuentro están en el rango $\{1,...,5\}$.

En la Tabla5.1 se muestra el detalle de cada una, donde E es el número de estaciones, P el número de puntos de encuentro, R el número de refugios y B el número de buses disponibles.

Set	E	P	R	B
S_2	1	2	2	2
S_3	1	3	3	2
S_4	1	4	4	3
S_5	1	5	5	3
S_6	1	6	6	4
S_7	1	7	7	4
S_8	1	8	8	5
S_9	1	9	9	5
S_{10}	1	10	10	6

Tabla 5.1: Caracerísticas de las instancias aleatorias.

5.2 Grupo 2: Instancias Incendio Valparaíso

Las instancias del segundo grupo están basadas en el Gran Incendio de Valparaíso. Para esto, se consideraron aspectos de la región, como características de los buses de transporte, topología de la zona de evacuación, así como aspectos específicos asociados al incendio del 2014.

5.2.1 Antecedentes del Gran Incendio de Valparaíso

- **Zona de riesgo**

  El Gran Incendio de Valparaíso tuvo lugar en la comuna de Valparaíso, principalmente en Cerro Mariposa, El Vergel, La Cruz, Cerro El Litre, Cerro Las Cañas, Cerro Miguel Ángel y Mercedes. De acuerdo a las cifras oficiales de ONEMI, el incendio afectó aproximadamente un total de 965,2 hectáreas, de las cuales 28,8 se encontraban habitadas [3].

  Durante el suceso, el sitio de Google de respuestas ante crisis publicó un mapa donde se detallaba la zona afectada por el incendio [14]. Para la generación del escenario, esta misma se utilizó como zona de riesgo, que es desde donde se debe llevar a cabo la evacuación.

- **Refugios**

  La información acerca de los refugios que fueron utilizados durante el incendio se obtuvo de diversas fuentes, principalmente de reportes de la ONEMI, además de artículos y noticias de diarios digitales. Esto debido a que no existe información oficial acerca de

todos los refugios que fueron utilizados. Por otro lado, refugios que se usaron durante los primeros días del incendio, posteriormente quedaron inmersos dentro de la zona de evacuación, por lo que no fueron considerados para la generación de las instancias. Cada uno de los refugios utilizados que fueron reportados por los medios fueron ubicados en Google Maps para obtener sus coordenadas, y poder calcular las distancias posteriormente.

La capacidad de cada refugio se estimó de acuerdo a la cantidad máxima de personas albergadas informada por los medios de comunicación, estableciendo un número levemente superior, múltiplo de la capacidad de los buses. Los establecimientos considerados como refugios corresponden a consultorios, escuelas y otros centros comunitarios. Los datos considerados para la generación de las instancias se pueden ver en la Tabla5.2. En total, los 12 refugios considerados tienen espacio para albergar a 2.250 personas.

Nombre	Capacidad aproximada
Centro Comunitario Las Cañas	60
Consultorio Marcelo Mena	120
Consultorio Reina Isabel	120
Escuela Alemania	210
Escuela Gaspar Cabrales	210
Escuela Grecia	510
Escuela Pablo Neruda	120
Estadio O'Higgins	210
Fortín Prat	210
Iglesia Juan Bosco	300
Liceo Técnico Femenino	120
Liceo Eduardo de la Barra	60

Tabla5.2: Detalle de los refugios considerados.

- **Número total de evacuados**

  Según fuentes oficiales, el incendio desplazó un total aproximado de 12.500 personas [15], entre dirigidos y auto evacuados. La máxima cantidad de personas albergadas fue aproximadamente 2.000, ya que gran parte de los evacuados se quedaron en casas de amigos, familiares o conocidos. Por este motivo, para generar el escenario se considera que se desea evacuar específicamente a las personas que se quedarán en los refugios, estableciendo un número total de evacuados igual a la capacidad total de los refugios. Es decir, se debe evacuar a 2.250 personas.

- **Puntos de encuentro**

Para definir las ubicaciones donde se reunen los evacuados a esperar los buses, se consideraron paraderos de buses y zonas amplias donde sea posible reunir un gran número de personas. Usando paraderos se asegura que los buses puedan transitar por las zonas seleccionadas, ya que debido a la topología de la zona de evacuación existen numerosas áreas donde sólo es posible transitar a pie. Los puntos de encuentro se distribuyeron dentro del área residencial de la zona de riesgo con la ayuda de Google Maps, considerando aproximadamente la densidad poblacional de cada sector. Es decir, las zonas más pobladas tienen mayor cantidad de puntos de encuentro. Esto tiene sentido, nuevamente, debido a la forma de la zona de evacuación, que corresponde en mayor parte a cerros y zonas de difícil acceso. La cantidad total de puntos de encuentro definidos es de 52.

La cantidad de personas en cada punto de encuentro, se estima dividiendo la zona de riesgo en seis áreas distintas. Las áreas con mayor densidad poblacional tienen una mayor cantidad de personas que aquellas áreas con menor población, asignándose un porcentaje específico a cada área para poder determinar la demanda en cada punto.

- **Estaciones**

  Para las estaciones se usaron las garitas que están cercanas a la zona de riesgo, que corresponden a 5 en total: Garita Galvarino, Garita José Serey, Garita Abelardo Núñez, Garita Camino La Pólvora y Garita Ramaditas. De acuerdo a la planificación de las salidas de los buses disponible en Google Maps asociada a cada garita, se estima que cada una tiene una capacidad para 25 buses.

- **Buses**

  Se consideraron los buses de transporte urbano de la región, que tienen una capacidad de aproximadamente 30 personas. Como simplificación del problema se asume que todos los buses necesarios están disponibles en sus garitas respectivas, para cuando es necesario hacer la evacuación.

5.2.2 Escenario de Evacuación

El escenario de evacuación se genera considerando todos los elementos anteriores. En la Figura5.1 se puede ver de manera gráfica la mayoría de los puntos definidos. Los puntos azules corresponden a las estaciones, los puntos verdes son los refugios, y los negros son los puntos de encuentro. La zona de riesgo está marcada con color rojo. Las distancias entre los nodos se obtuvieron utilizando la API de Google que proporciona la distancia entre dos puntos indicando las coordenadas de cada uno [1]. El mapa completo diseñado para representar el escenario se puede revisar en [1] con mayor detalle.

[1] Google Maps Distance Matrix API: https://developers.google.com/maps/documentation/distance-matrix/intro?hl=es

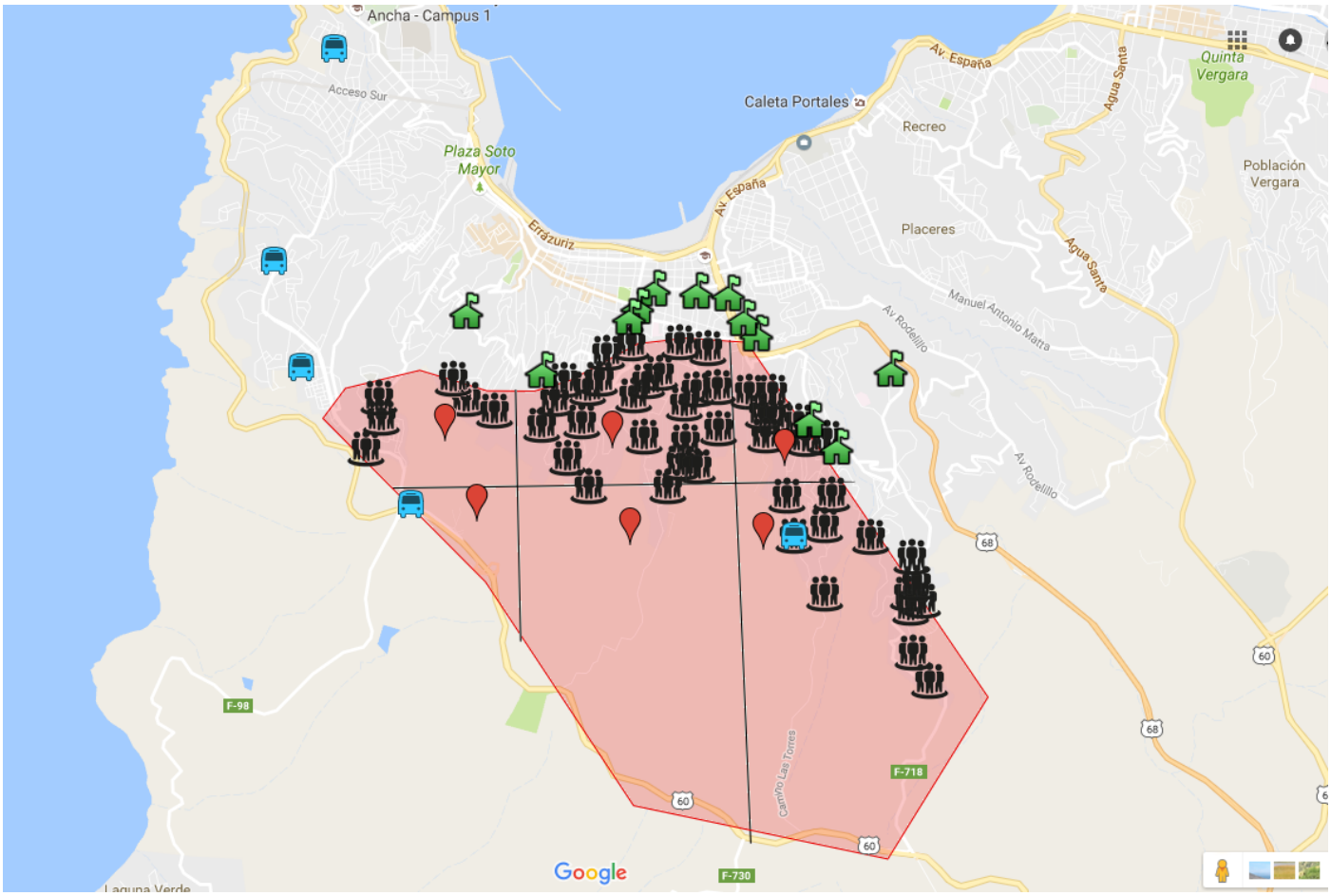


Figura 5.1: Escenario de evacuación basado en el Gran Incendio de Valparaíso. Fuente: Elab. Propia.

5.2.3 Características de las instancias generadas

En base a las características señaladas anteriormente, se generaron 5 instancias con la misma cantidad de nodos de cada tipo, variando la cantidad de buses en 10, 30, 50, 70 y 75. Las distancias entre los nodos se entregan en unidades de metros. En la Tabla 5.3 se muestra el detalle de las instancias, con E el número de estaciones, P el número de puntos de encuentro, R el número de refugios y B el número de buses disponibles.

N° Instancia	E	P	R	B
1	5	52	12	10
2	5	52	12	30
3	5	52	12	50
4	5	52	12	70
5	5	52	12	75

Tabla 5.3: Características de las instancias generadas.

5.3 Conclusiones

En este capítulo se describieron en detalle las instancias que serán utilizadas para resolver el BEP, las cuales permitirán evaluar el rendimiento del algoritmo propuesto. Se consideraron dos tipos de instancias: un primer conjunto, que fue construido con valores aleatorios para distancias entre nodos, capacidades de refugios y demandas en puntos de encuentro. El segundo conjunto, corresponde a un set de instancias que están basadas en el Gran Incendio de Valparaíso, tomando como base los refugios utilizados en esa ocasión, así como el área total donde tuvo lugar el incendio. Para este caso, se definieron además posibles puntos de encuentro dentro de la zona afectada, y se calcularon distancias entre estos y los refugios. Para elaborar estas instancias se consideraron varios supuestos, que acotan el escenario y permiten adaptarlo a una instancia del modelo propuesto.

En el siguiente capítulo, se mostrarán los resultados experimentales al aplicar el algoritmo propuesto en el Capítulo 4 para las instancias mencionadas, además de realizar análisis de rendimiento para sentar las bases para futuras investigaciones.

6 Experimentos y Resultados

En este capítulo se estudia el rendimiento del algoritmo implementado basado en la metaheurística GRASP. Para esto, se presenta una descripción de los experimentos, así como las métricas utilizadas para evaluar el rendimiento del algoritmo, como por ejemplo, la calidad de la función objetivo y el tiempo de ejecución. Además, se presentan los resultados de los experimentos realizados sobre las instancias aleatorias, para posteriormente pasar a los resultados obtenidos sobre las instancias generadas en base al escenario del incendio de Valparaíso.

Especificaciones del equipo y del entorno de trabajo

Las pruebas del algoritmo GRASP fueron realizadas en un equipo con procesador Intel core i3 2.53 GHz, memoria RAM 6.0 GB (5.74 utilizables), sobre el sistema operativo Ubuntu 16.04 LTS. Para comparar los resultados obtenidos para las instancias aleatorias, se usó el solver CPLEX v. 12.6. con el modelo de programación lineal entera mixta de [12].

6.1 Experimentos con instancias aleatorias

6.1.1 Análisis de parámetros

En primer lugar, se evalua el rendimiento del algoritmo sobre las instancias aleatorias, usando distintas combinaciones de los siguientes parámetros:

- Tamaño de la lista de candidatos: Tal como se señaló en la Sección 4, la lista de candidatos se construye para cada nodo (estación, punto de encuentro o refugio), y contiene los k viajes más cortos que se pueden realizar desde ese nodo. El parámetro para establecer el tamaño de la lista de candidatos corresponde al porcentaje de la lista completa de viajes factibles desde ese nodo. Por ejemplo, si desde una estación es posible realizar 10 viajes distintos y el parámetro de tamaño es 0.3, la lista de candidatos contiene los 3 viajes más cercanos. Se usan los valores 0.1, 0.3 y 0.5.
- Máximo número de iteraciones de Hill Climbing (fase de postprocesamiento de GRASP). Se usan 20, 50 y 100 iteraciones.

Además, para la generación de los números aleatorios se probaron los valores 4, 6, 8 y 10 como semillas. Estos experimentos se ejecutaron sobre las 10 instancias de los 9 sets, con un tiempo máximo de 30 segundos por ejecución. A continuación, se reportan los valores obtenidos para las instancias del set 10, ya que en el resto de instancias la influencia de los parámetros se hace cada vez menor, a medida que las instancias se van haciendo más pequeñas.

A pesar de que las instancias de un mismo set poseen las mismas características de magnitud (número de buses y nodos), los valores que son generados aleatoriamente (distancias, demandas y capacidades de los nodos) definen en gran medida el valor óptimo de la función objetivo. Por este motivo, en lugar de usar el valor de la función objetivo obtenido con los distintos parámetros, se establece una medida de puntaje, que permite normalizar los resultados de todas las instancias de un set. Este se calcula tomando el resultado obtenido en una ejecución del algoritmo, con una configuración específica de parámetros y para una instancia específica del set, y dividiéndolo por el mejor resultado obtenido para esa instancia, con cualquier combinación de parámetros.

El puntaje será igual a 1 si el resultado obtenido es el mejor encontrado para esa instancia, y mayor que 1 si es un resultado peor (mayor que el mejor valor encontrado, ya que el problema busca minimizar). De esta manera, para cada parámetro se busca el valor que permita obtener un puntaje promedio lo más cercano a 1 posible.

Tamaño LRC	Puntaje Promedio	Desv. Est. Puntaje
0.1	**1.003**	**0.008**
0.3	1.077	0.073
0.5	1.257	0.104

Tabla 6.1: Resultados obtenidos para distintos valores del tamaño de la Lista Restringida de Candidatos de GRASP.

Máx. Iteraciones HC	Puntaje Promedio	Desv. Est. Puntaje
20	1.112	0.130
50	1.112	0.130
100	1.112	0.130

Tabla 6.2: Resultados obtenidos para distintos valores del máximo de iteraciones de Hill Climbing.

Se observa en la Tabla 6.1 que los mejores resultados para las instancias del set 10 se obtuvieron con un menor tamaño de la lista de candidatos. A medida que se aumenta el tamaño de la lista, el valor promedio del puntaje aumenta, es decir, los resultados para las funciones

objetivo de cada ejecución son más grandes en relación al mejor valor encontrado para cada instancia. Un tamaño de la LRC 0.1 permitió, además, encontrar resultados más cercanos en general al menor valor posible, lo que se puede ver en el valor de la desviación estándar, que es menor respecto de los otros dos resultados. A medida que aumenta el tamaño de la LRC, los resultados obtenidos son más variables para una misma instancia.

Estos resultados se deben a que con pocos elementos en la lista de candidatos, la fase constructiva de GRASP se comporta de manera más elitista. En cambio, a medida que se aumenta el tamaño de la lista, la generación de la solución inicial se va haciendo más aleatoria, dando cada vez menos importancia a la calidad de la solución en pos de generar soluciones más diversas. En este caso, esto se traduce en un empeoramiento general de las soluciones obtenidas.

Un tamaño de 0.1 genera una lista con una buena cantidad de elementos para generar soluciones de buena calidad, a la vez que se añade un poco de diversidad a la solución, sobretodo en la fase más temprana de la construcción. En ésta, a medida que una solución se va completando, las listas se van haciendo cada vez más pequeñas, ya que sólo se van considerando los nodos donde aún hay demanda/capacidad disponible, por lo que el tamaño de la lista de candidatos va tendiendo a 1.

Por otro lado, en la Tabla6.2 se tiene que los valores utilizados como máximo de iteraciones de Hill Climbing, no tienen ninguna influencia en los resultados obtenidos. La situación en instancias de los otros sets fue la misma. Es posible que esto se deba a que el algoritmo de búsqueda local se queda estancado en óptimos locales con una menor cantidad de iteraciones que las utilizadas, por lo que en ninguna ocasión se alcanza el máximo de iteraciones definido.

6.1.2 Calidad de las soluciones obtenidas

Usando 0.1 para el largo de la LRC y un máximo de iteraciones de hill climbing de 20, se obtuvieron los resultados sobre todas las instancias de los 9 sets. El tiempo de ejecución es el mismo que anteriormente, de 30 segundos. Los resultados de cada instancia son comparados con los valores obtenidos con CPLEX, con un tiempo máximo de ejecución de 15 minutos. Si no se ha encontrado la solución óptima cuando se cumple el límite de tiempo, se entrega la mejor solución encontrada hasta el momento. Nuevamente, se normalizaron los resultados obtenidos, para poder obtener el rendimiento promedio sobre todas las instancias de cada set. El valor normalizado se obtiene dividiendo el resultado obtenido por cada técnica, por el mejor resultado obtenido entre los dos algoritmos. De igual manera, el valor 1 corresponde a la mejor solución.

En las Tablas 6.3 y 6.4 se presentan los resultados obtenidos sobre cada instancia. Los valores marcados con asterisco son aquellos valores óptimos encontrados por CPLEX. Los demás resultados son los mejores encontrados al finalizar la ejecución por límite de tiempo. Para las instancias más pequeñas S_2 y S_3, se tiene que CPLEX logra obtener 10 resultados óptimos de 10, y 8 de 10 respectivamente. Por otro lado, GRASP es capaz de encontrar todos los

resultados óptimos para S_2, y logra encontrar 2 de 10 en S_3. En este último set, el resto de los resultados son bastante cercanos a los de CPLEX, teniéndose que para la instancia S_3-02 se obtuvo una diferencia de 8 unidades, mientras que en el resto de las instancias las diferencias van desde 1 a 4 unidades.

Hasta el set S_5, CPLEX es capaz de encontrar casi siempre la mejor solución, a excepción de algunos casos en que ambos algoritmos encuentran el mismo valor. Cuando GRASP encuentra resultados peores, estos son siempre cercanos, con diferencias pequeñas de 0 a 5 unidades, en general.

Desde el set S_6 en adelante, GRASP comienza a encontrar algunos valores mejores que los obtenidos con CPLEX. A partir del set S_8 en adelante, GRASP se posiciona definitivamente como un mejor enfoque que CPLEX, logrando obtener 9 de los 10 mejores valores encontrados en cada instancia de S_8 y S_9. Para S_{10}, todos los resultados de GRASP son los mejores. En estos últimos 3 sets, no sólo se logró obtener en general mejores resultados que con CPLEX, sino que además se debe considerar que GRASP se ejecutó con un máximo de tiempo de 30 segundos, mientras que para CPLEX el límite fue de 15 minutos. Por otro lado, CPLEX consume una gran cantidad de memoria, teniéndose que en ocasiones fue necesario repetir experimentos, debido a que no quedaba memoria disponible. Para la instancia S_8-09, por ejemplo, no fue posible obtener resultados en ninguna ocasión.

	GRASP	CPLEX	GRASP Norm	CPLEX Norm
S_2-01	46*	46*	1.00	1.00
S_2-02	13*	13*	1.00	1.00
S_2-03	9*	9*	1.00	1.00
S_2-04	44*	44*	1.00	1.00
S_2-05	29*	29*	1.00	1.00
S_2-06	44*	44*	1.00	1.00
S_2-07	18*	18*	1.00	1.00
S_2-08	16*	16*	1.00	1.00
S_2-09	12*	12*	1.00	1.00
S_2-10	38*	38*	1.00	1.00
Media	**-**	**-**	**1.00**	**1.00**
S_3-01	35	31*	1.13	1.00
S_3-02	60	52	1.15	1.00
S_3-03	31	30*	1.03	1.00
S_3-04	60	60	1.00	1.00
S_3-05	10	8*	1.25	1.00
S_3-06	47	46*	1.02	1.00
S_3-07	16*	16*	1.00	1.00
S_3-08	24*	24*	1.00	1.00
S_3-09	24	20*	1.20	1.00
S_3-10	72	70*	1.03	1.00
Media	**-**	**-**	**1.08**	**1.00**
S_4-01	26	25	1.04	1.00
S_4-02	35	31	1.13	1.00
S_4-03	26	26	1.00	1.00
S_4-04	42	41	1.02	1.00
S_4-05	21	13	1.62	1.00
S_4-06	16	16*	1.00	1.00
S_4-07	30	21	1.43	1.00
S_4-08	21	20	1.05	1.00
S_4-09	21	17*	1.24	1.00
S_4-10	31	24*	1.29	1.00
Media	**-**	**-**	**1.18**	**1.00**
S_5-01	24	22	1.09	1.00
S_5-02	24	22	1.09	1.00
S_5-03	44	42	1.05	1.00
S_5-04	48	46	1.04	1.00
S_5-05	21	18	1.17	1.00
S_5-06	19	17	1.12	1.00
S_5-07	22	22	1.00	1.00
S_5-08	26	24	1.08	1.00
S_5-09	43	39	1.10	1.00
S_5-10	24	22	1.09	1.00
Media	**-**	**-**	**1.08**	**1.00**

Tabla 6.3: Comparación de resultados, instancias S_2 a S_5.

	GRASP	CPLEX	GRASP Norm	CPLEX Norm
S_6-01	28	23	1.22	1.00
S_6-02	19	15	1.27	1.00
S_6-03	23	23	1.00	1.00
S_6-04	30	29	1.03	1.00
S_6-05	13	11	1.18	1.00
S_6-06	25	26	1.00	1.04
S_6-07	18	17	1.06	1.00
S_6-08	27	22	1.23	1.00
S_6-09	26	30	1.00	1.15
S_6-10	29	24	1.21	1.00
Media	**-**	**-**	**1.12**	**1.02**
S_7-01	15	15	1.00	1.00
S_7-02	28	23	1.22	1.00
S_7-03	29	28	1.04	1.00
S_7-04	24	23	1.04	1.00
S_7-05	27	28	1.00	1.04
S_7-06	29	27	1.07	1.00
S_7-07	20	20	1.00	1.00
S_7-08	35	30	1.17	1.00
S_7-09	16	15	1.07	1.00
S_7-10	27	23	1.17	1.00
Media	**-**	**-**	**1.08**	**1.00**
S_8-01	26	26	1.00	1.00
S_8-02	20	24	1.00	1.20
S_8-03	16	17	1.00	1.06
S_8-04	22	22	1.00	1.00
S_8-05	29	31	1.00	1.07
S_8-06	29	27	1.07	1.00
S_8-07	21	21	1.00	1.00
S_8-08	26	26	1.00	1.00
S_8-09	24	-	1.00	-
S_8-10	19	22	1.00	1.16
Media	**-**	**-**	**1.01**	**1.05**
S_9-01	29	31	1.00	1.07
S_9-02	20	17	1.18	1.00
S_9-03	22	24	1.00	1.09
S_9-04	23	26	1.00	1.13
S_9-05	34	41	1.00	1.21
S_9-06	30	33	1.00	1.10
S_9-07	25	28	1.00	1.12
S_9-08	20	20	1.00	1.00
S_9-09	21	26	1.00	1.24
S_9-010	17	21	1.00	1.24
Media	**-**	**-**	**1.02**	**1.12**

Tabla 6.4: Comparación de resultados, instancias S_6 a S_9.

Un resumen de los resultados en función de las medias de los valores normalizados, se muestran en la Tabla6.5. Aquí se observa de manera más general la tendencia descrita anteriormente. Para las instancias más pequeñas CPLEX es capaz de encontrar soluciones óptimas, o mejores soluciones que GRASP; pero a medida que las instancias se van volviendo más grandes, CPLEX ya no es capaz de encontrar buenos resultados, teniéndose que GRASP lo supera en calidad y uso de los recursos. Para instancias aún mayores, CPLEX requiere una capacidad de procesamiento y memoria demasiado grande para poder encontrar alguna solución, por lo que usar una técnica incompleta se convierte en la única alternativa posible debido a su buena escalabilidad.

Set	Media GRASP	Media CPLEX
2	1.00	1.00
3	1.08	1.00
4	1.18	1.00
5	1.08	1.00
6	1.12	1.02
7	1.08	1.00
8	1.01	1.05
9	1.02	1.12
10	1.00	1.20

Tabla6.5: Comparación general de GRASP y CPLEX sobre instancias aleatorias, resultados normalizados.

6.2 Experimentos con instancias Escenario Valparaíso

En primer lugar, se evalúan los mejores parámetros para resolver una instancia del escenario de Valparaíso, considerando 5 estaciones, 52 puntos de encuentro, 12 refugios y 50 buses disponibles para la evacuación. Las semillas utilizadas fueron 2, 3, 5, 7, 9, 11, 13 y 15. En total, se realizaron 192 ejecuciones, considerando los siguientes valores de parámetros:

- Tamaño de la lista de candidatos: Al igual que en la fase de experimentación en instancias aleatorias, se usaron valores 0.1, 0.3 y 0.5 para la proporción del tamaño de la lista.
- Máximo número de iteraciones de Hill Climbing: Se usaron 20, 50, 100 y 500 iteraciones.
- Tiempo máximo de ejecución del algoritmo: Según [11], el tiempo del que se dispondría para poder generar una planificación apenas ocurra una situación de emergencia, sería de 3 minutos aproximadamente. Por supuesto, este tiempo dependerá mucho de la circunstancia en que se deba realizar la evacuación, pero para efectos de este

trabajo se considerarán 3 y 5 minutos como un tiempo límite para obtener una planificación.

Como en estas instancias las distancias entre nodos vienen dadas en unidades de metros, el valor de la función objetivo obtenido por el algoritmo tiene las mismas unidades, y en cada caso representa la distancia que recorre el bus con la ruta completa más larga. Por este motivo, en cada caso se busca el resultado con la menor distancia posible. Para obtener la duración de la evacuación, se usará una velocidad de 40 [km/hr], considerando que el límite máximo de velocidad en una zona urbana es de 60 [km/hr], y que gran parte de los recorridos se deben realizar en cerros, donde la movilidad es más reducida que en una zona más plana.

Por otro lado, en [11] se considera un escenario en que se evacúan 1750 personas en un área de 78.5 hectáreas, es decir, una densidad poblacional de 22.3 personas por hectárea. Considerando las características del escenario planteado, y el algoritmo Tabu Search implementado para su resolución, se obtuvo un tiempo de evacuación de 81 minutos. En el escenario considerado en el presente trabajo, se debe evacuar a 2250 personas en un área de 28.8 hectáreas, con una densidad de 78.1 personas por hectárea. Usando regla de tres, se estima que una evacuación que mantenga las proporciones debería durar aproximadamente 283.6 minutos, o 4.7 horas. Como el incendio duro 4 días, tiempo durante el cual pudo avanzar a lo largo de la zona afectada, se estima que una evacuación total del área que dure aproximadamente 4 horas es un buen valor referencial para medir la calidad de las soluciones obtenidas, ya que CPLEX no fue capaz de resolver estas instancias, debido a su magnitud.

A continuación, se reportan los resultados variando cada uno de los parámetros.

6.2.1 Tamaño de la lista de candidatos

Los resultados variando el tamaño de la LRC de GRASP se muestran en la Tabla6.6. Se observa que, a diferencia de los experimentos sobre instancias aleatorias, en este caso los mejores resultados se obtuvieron usando una proporción de 0.3. Este valor ofrece un buen equilibrio entre considerar la calidad de la solución construida, y la diversificación de la misma para explorar distintos espacios de búsqueda.

Tamaño LRC	Promedio FO [metros]	Desv. Est. FO	Mejor valor FO [metros]
0.1	12407	215.36	12086
0.3	**12286**	338.83	**11873**
0.5	12347	386.75	11879

Tabla6.6: Resultados obtenidos para distintos valores del tamaño de la Lista Restringida de Candidatos de GRASP

6.2.2 Máximo número de iteraciones de Hill Climbing

Los resultados en la Tabla6.7 muestran que con un máximo de iteraciones de 500 se consigue el mejor promedio para el valor de la función objetivo. A medida que aumenta el número de iteraciones, mejora el promedio de la función objetivo, a pesar de que con 50, 100 y 500 iteraciones se obtiene el mismo valor mínimo. Esto puede deberse a en el mejor de los casos, con estos parámetros el algoritmo es capaz de encontrar una solución muy cercana al valor óptimo, o bien, dada la naturaleza del movimiento aplicado en Hill Climbing, este valor corresponde a la mejor configuración que es posible generar, y el algoritmo Hill Climbing se detiene antes de alcanzar el máximo de iteraciones en gran parte de las ejecuciones promediadas. Se estima que una cantidad cercana a las 100 iteraciones es suficiente para obtener buenos resultados, por tener un promedio bastante cercano al obtenido con 500 iteraciones, además de ser capaz de encontrar el mejor valor con los parámetros adecuados.

Máx. Iteraciones HC	Promedio FO [metros]	Desv. Est. FO	Mejor valor FO [metros]
20	12799	250.60	12305
50	12221	156.70	**11873**
100	12185	170.95	**11873**
500	**12181**	168.77	**11873**

Tabla6.7: Resultados obtenidos para distintos valores del máximo de iteraciones de Hill Climbing.

6.2.3 Tiempo de ejecución de GRASP

Evaluando los tiempos de ejecución del algoritmo necesarios para generar una evacuación, se tiene que aumentar el tiempo máximo de ejecución permite mejorar levemente la calidad promedio de los resultados obtenidos, pero no permite mejorar el valor más pequeño encontrado, tal como se muestra en Tabla6.8. En ambos casos el mejor valor encontrado para la función objetivo fue de 11873 metros. La poca diferencia entre ambos tiempos de ejecución puede deberse a las razones señaladas anteriormente: el algoritmo es lo suficientemente bueno como para encontrar una solución muy cercana al óptimo (u óptima en el mejor de los casos), o esta es la mejor solución que es posible generar a partir del movimiento implementado para la búsqueda local.

Tiempo ejecución [s]	Promedio FO [metros]	Desv. Est. FO	Mejor valor FO [metros]
180	12382	331.54	11873
300	12312	314.00	11873

Tabla6.8: Resultados obtenidos para distintos valores del tiempo de ejecución.

6.2.4 Mejor solución para el escenario de evacuación

Considerando los parámetros determinados anteriormente, con un tamaño para la LRC de 0.3, un máximo de 100 iteraciones para Hill Climbing, y con un tiempo máximo de ejecución de 180 segundos (3 minutos), se obtuvo una planificación en la que el bus con la ruta más larga recorre 11.87 kilómetros. Considerando una velocidad de 40 [km/hr], esto corresponde a una evacuación con una duración de 17.8 minutos (sin considerar carga y descarga de pasajeros). Si bien la duración disponible para la evacuación depende mucho de la situación en que esta sea necesaria y las características de la emergencia, este valor se encuentra muy por debajo del límite máximo estimado al comienzo de la sección 6.2, por lo que el resultado es bastante satisfactorio.

Cabe destacar que en [11], la evacuación se lleva a cabo usando sólo 3 buses con capacidad de 80 cada uno (240 cupos en total), por lo que no es de extrañar que la evacuación sea mucho más rápida en este caso, usando 50 buses con capacidad de 30 personas cada uno (1500 cupos en total). De todas maneras, es necesario considerar que en un escenario real es posible que el uso de un bus tenga un costo asociado, por lo que una forma de mejorar este problema podría ser replantearlo con múltiples objetivos, buscando minimizar el tiempo de evacuación a la vez que se minimiza la cantidad de buses a utilizar, en base a los costos asociados.

Una parte de la planificación obtenida se muestra en la Tabla 6.9. La mayor cantidad de viajes que realiza un bus, es de 3 viajes de traslado de evacuados. La menor cantidad es de 1 viaje de traslado. Todos los buses realizan al menos dos viajes (1 de inicio y 1 de traslado de evacuados), debido a que el algoritmo usa todos los buses que se indican en la instancia de entrada.

Tiempo [min]		Viaje 0	Viaje 1	Viaje 2	Viaje 3
17.0	Bus 1	(D 0,AP 13)	(AP 13,S 2)	(AP 21,S 2)	(AP 6,S 6)
17.8	Bus 2	(D 0,AP 11)	(AP 11,S 1)	(AP 2,S 8)	
16.6	Bus 3	(D 0,AP 16)	(AP 16,S 4)	(AP 15,S 2)	(AP 3,S 3)
14.6	Bus 4	(D 0,AP 4)	(AP 4,S 6)		
17.8	Bus 5	(D 0,AP 18)	(AP 18,S 4)	(AP 16,S 5)	(AP 5,S 2)
...	...	...	...	...	...
6.9	Bus 35	(D 3,AP 45)	(AP 45,S 0)		
10.3	Bus 36	(D 3,AP 47)	(AP 47,S 9)		
17.8	Bus 37	(D 3,AP 39)	(AP 39,S 7)		
13.0	Bus 38	(D 3,AP 26)	(AP 26,S 5)		
...	...		...	...	...

Tabla 6.9: Parte de una planificación para la instancia Valparaíso-5-52-12-50. Tiempo de evacuación: 17.8 minutos.

6.3 Estudio del número de buses usados en la evacuación

Dejando los mejores valores para los parámetros elegidos anteriormente, se evalúa cómo cambia el tiempo total de la evacuación según la cantidad de buses utilizados. Para esto se usan las instancias de Valparaíso, variando la cantidad de buses en 10 unidades, entre 10 y 70 buses, distribuidos de forma homogénea entre las 5 estaciones consideradas. Las demás características del escenario se mantienen constantes, y se usa una instancia de cada tipo.Además se añadió el caso de 75 buses, que es el más grande posible, ya que permite cubrir el 100% de la demanda en un solo viaje.

Este análisis se hace en base a los postulados de [2], donde se evalúa cómo varía el tiempo de evacuación en función de los vehículos utilizados, para así poder elegir una cantidad óptima. Esto debido a que utilizar buses u otro medio para la evacuación tiene un costo asociado.

Los resultados obtenidos corresponden a la ejecución del algoritmo usando los valores de los parámetros elegidos en la sección anterior, es decir, tiempo máximo de ejecución de 180 segundos, largo de la lista de candidatos de 0.3, y un máximo de 100 iteraciones de HC. La Tabla6.10 muestra los promedios obtenidos para la función objetivo, usando como semillas los valores 2, 3, 5 y 7. Posteriormente, en la Figura6.1 se muestran los mismos resultados de manera gráfica.

N° de Buses	Promedio FO [metros]	Duración Evacuación [min]
10	35730.5	53.6
20	19371.5	29.1
30	15092.25	22.6
40	13008.5	19.5
50	12157.25	18.2
60	11992.5	18.0
70	13463	20.2
75	18534	27.8

Tabla6.10: Resultados obtenidos para distintos valores del número de buses.

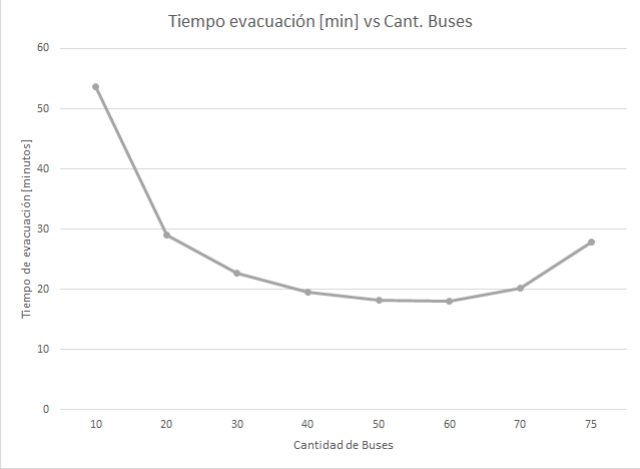


Figura 6.1: Gráfica de tiempos de evacuación según el número de buses.

Se observa que el valor de la función objetivo no disminuye linealmente con el aumento de buses. Esto concuerda con la Observación 3 señalada en [2].

Por otro lado, se tiene que el mejor valor se obtiene usando 60 buses para un mismo escenario. Usando más buses se tiene un leve empeoramiento de la función objetivo. Esto se debe a que a mayor cantidad de buses, en el caso más extremo se requerirá sólo un viaje para trasladar a todas las personas a los refugios. Con cada bus teniendo una lista tan limitada, con sólo dos viajes (el viaje inicial desde la estación, y el viaje de traslado de pasajeros), el valor de la función dependerá mucho de la solución inicial, y dada la naturaleza del movimiento no es posible mejorar lo suficiente la solución.

Cabe señalar que a partir de los 75 buses, el valor de la función objetivo se mantendrá constante a pesar de que se aumente el número de buses. Esto se debe a que los 75 buses cubren totalmente la demanda en todos los puntos de encuentro, ya que 75 buses con capacidad para 30 personas cada uno, pueden trasladar en total a 2250 personas, que es el total de evacuados. Por este motivo, si se envían más buses, éstos no tendrán personas que evacuar y finalmente no participan en la evacuación.

Ambas situaciones concuerdan con la Observación 2 de [2], que señala que existe un límite de optimalidad respecto de la cantidad de vehículos utilizados, y que incrementar la cantidad no genera mejora en la función objetivo.

En relación a la calidad de los resultados, se puede ver que en el caso en que se usen sólo 10 buses la duración de la evacuación será de aproximadamente 54 minutos, lo que se encuentra por debajo del límite máximo considerado anteriormente, de 4 horas. Se puede concluir que es posible llevar a cabo una evacuación en un tiempo aceptable usando sólo 10 buses, aunque como se mencionó anteriormente, dependerá de la situación evaluar el tiempo que se tiene disponible para evacuar. En una situación crítica, usando 60 buses es posible evacuar a la población en 18 minutos aproximadamente, considerando el escenario planteado. Como se

vió anteriormente, este sería la máxima cantidad de buses a usar en este caso, debido a que poner más buses sólo empeorará los tiempos de la evacuación.

6.4 Conclusiones

En este capítulo se mostraron los resultados de la aplicación de la metaheurística basada en GRASP a dos conjuntos de instancias, obteniendo resultados de buena calidad en tiempos pequeños. Para ambos tipos de instancias se realizó un análisis de parámetros para evaluar el comportamiento del algoritmo, con lo cual se pudo verificar su robustez al ser capaz de obtener resultados de calidad similar en todos los casos analizados.

Por otro lado, respecto a la calidad del algoritmo, en el caso de las instancias aleatorias se obtuvo que GRASP logró encontrar valores cercanos a los resultados obtenidos con CPLEX en las instancias más pequeñas, y que obtiene mejores valores que este para instancias de mayor tamaño. Para el caso de las instancias basadas en el incendio de Valparaíso, se obtuvieron tiempos de evacuación cercanos a los 18 minutos, lo que sugiere la factibilidad de aplicación del algoritmo propuesto a escenarios menos relajados y más realistas de forma satisfactoria.

7 Conclusiones

Con el trabajo realizado se logró resolver el Problema de Evacuación de Buses, usando un modelo simplificado basado en [2] y [12]. Para esto se implementó el algoritmo Greedy Randomized Adaptive Search Procedure (GRASP), que consta principalmente de una repetición iterativa de una fase de construcción de la solución, seguido de una fase de postprocesamiento para mejorar la solución construida. GRASP tiene la ventaja de que tiene una estructura fácil de implementar, además de requerir de pocos parámetros para su ejecución. Los parámetros necesarios para la ejecución del algoritmo se eligieron realizando varias pruebas con distintos valores para los parámetros de GRASP, además de los parámetros específicos del problema de evacuación.

Para evaluar la calidad del algoritmo se generaron dos tipos de instancias. Las primeras tienen valores aleatorios, y están basadas en las instancias utilizadas en [12]. Por otro lado, se confeccionó un escenario de evacuación basado en el Gran Incendio de Valparaíso, considerando elementos que se asemejaran lo más posible al evento real acontecido, así como considerando aspectos que entrarían en juego en caso de una evacuación real en el sector en que ocurrió el Incendio.

Por ejemplo, para las estaciones se consideró que sus ubicaciones no quedaran muy lejos de la zona de riesgo; además, se revisó la cantidad de buses que se estacionan en cada garita, para conocer el número máximo disponible.

Las ubicaciones de los puntos de encuentro también consideran varios aspectos como la amplitud de la zona para poder reunir a varias personas; las condiciones del camino, de modo que los buses sean capaces de recorrer la zona; la cercanía de los puntos a zonas habitadas, para asegurar que se pueda llegar a pie a algún punto cercano de evacuación; la cantidad de personas esperando en cada punto depende de la densidad poblacional (estimada) de la zona donde se encuentra el punto, entre otros.

Para efectos de la simplificación del problema se hace el supuesto de que las distancias de ida y vuelta entre dos nodos son iguales, siendo que en la realidad esto no es así. Una variante del problema podría considerar la generación de grafos con mayor cantidad de conexiones, por ejemplo, entre refugios o entre puntos de encuentro, así como arcos dirigidos para mantener las diferencias entre recorridos de ida y vuelta.

Con todos los datos recopilados, se generaron varias instancias buscando representar en medida de lo posible una situación real.

El algoritmo GRASP permitió encontrar resultados que cabría esperar para una evacuación en una situación de emergencia. El mejor valor obtenido fue de 11992 metros, que a una

velocidad aproximada de 40 km/h demora 18 minutos en completarse, considerando sólo los viajes, sin carga y descarga de pasajeros. Los resultados son satisfactorios, más aun considerando que sólo se ejecuta el algoritmo durante sólo 3 minutos, y sin la necesidad de una cantidad de memoria demasiado grande, como en el caso del modelo MIP resuelto con CPLEX.

Como trabajo futuro, es posible hacer varias mejoras. En primer lugar, las instancias generadas en base al incendio de Valparaíso contienen una gran cantidad de estimaciones, debido a la falta de información oficial disponible al respecto. Dos de los factores que se podrían mejorar son: la estimación de la capacidad de los refugios, calculándola en base al área utilizable de cada establecimiento; y la estimación de la densidad poblacional en cada zona. Esta última se podría calcular en base al Censo más reciente, conociendo el número de habitantes por sector.

Por otro lado, tal como se mencionó anteriormente, una opción es añadir nuevos elementos al modelo del problema para poder incorporar aspectos más realistas de un escenario de evacuación. Por ejemplo, se puede reformular el problema, y transformarlo a uno con múltiples objetivos, minimizando la duración de la evacuación a la vez que se minimizan los costos de la misma, eligiendo, por ejemplo, la cantidad óptima de buses a utilizar, la cantidad de puntos de encuentro y refugios disponibles, entre otras cosas. Otra idea sería complementar el problema planteado con un problema de cobertura de áreas, para determinar las mejores ubicaciones para habilitar los puntos de encuentro. Finalmente, se puede considerar que el escenario que se presenta en este trabajo, correspondiente a un incendio de gran magnitud, es más bien una amenaza que avanza a lo largo de una zona de manera dinámica y relativamente lenta (en comparación con otro tipo de catástrofes, como lo serían un tsunami o una explosión de una bomba). En este tipo de amenaza, la zona de riesgo se va definiendo dinámicamente a medida que el fuego avanza, por lo que sería un buen complemento plantear una versión del problema en que lo puntos de encuentro y refugios se vayan definiendo a medida que se vaya haciendo necesario. Esto, proyectando a la vez cómo irá variando el escenario en las próximas horas o en los próximos días, para evitar ubicar refugios en zonas que posteriormente tendrán que ser evacuadas por el avance del fuego.

Respecto a la implementación de GRASP, como mejora sería bueno probar con distintos movimientos para la fase de búsqueda local, de modo que instancias más complejas tuvieran mayor posibilidad de llegar a soluciones más cercanas al óptimo. Además, se pueden realizar más pruebas, por ejemplo, aumentando el tiempo de ejecución para ver si se logra alguna mejora, o evaluando más valores de los distintos parámetros, en especial la semilla para la generación de número aleatorios, que tiene un impacto bastante importante en los resultados obtenidos.

Bibliografía

[1] Escenario de evacuación para el incendio en valparaíso, Sept 2016. https://drive.google.com/open?id=12Hdh_y34PNlgIBop0qmCrigNI-s&usp=sharing.

[2] Douglas R Bish. Planning for a bus-based evacuation. *OR spectrum*, 33(3):629–654, 2011.

[3] ONEMI Chile. Gran incendio de valparaíso, Abril 2014. http://www.onemi.cl/incendio-en-valparaiso/ (Última visita Sept. 20, 2016).

[4] Benoit Crevier, Jean-François Cordeau, and Gilbert Laporte. The multi-depot vehicle routing problem with inter-depot routes. *European Journal of Operational Research*, 176(2):756–773, 2007.

[5] Tanka Nath Dhamala. A survey on models and algorithms for discrete evacuation planning network problems. *Journal of Industrial and Management Optimization*, 11(1):265–289, 2015.

[6] Moshe Dror, Gilbert Laporte, and Pierre Trudeau. Vehicle routing with split deliveries. *Discrete Applied Mathematics*, 50(3):239–254, 1994.

[7] James R Elliott and Jeremy Pais. Race, class, and hurricane katrina: Social differences in human responses to disaster. *Social Science Research*, 35(2):295–321, 2006.

[8] Thomas A Feo and Mauricio GC Resende. A probabilistic heuristic for a computationally difficult set covering problem. *Operations research letters*, 8(2):67–71, 1989.

[9] Marc Goerigk, Kaouthar Deghdak, and Philipp Heßler. A comprehensive evacuation planning model and genetic solution algorithm. *Transportation research part E: logistics and transportation review*, 71:82–97, 2014.

[10] Marc Goerigk, Kaouthar Deghdak, and Vincent T'Kindt. *A Two-Stage Robustness Approach to Evacuation Planning with Buses*. Technische Universität Kaiserslautern, Fachbereich Mathematik, 2013.

[11] Marc Goerigk and Bob Grün. The robust bus evacuation problem. Technical report, Fachbereich Mathematik, Technical University of Kaiserslautern, 2012.

[12] Marc Goerigk, Bob Grün, and Philipp Heßler. Branch and bound algorithms for the bus evacuation problem. *Computers & Operations Research*, 40(12):3010–3020, 2013.

[13] Marc Goerigk, Bob Grün, and Philipp Heßler. A branch-cut-and-price approach to the bus evacuation problem with integrated collection point and shelter decisions. Technical report, Technische Universität Kaiserslautern, Fachbereich Mathematik, 2013.

[14] Google Crisis Response. Incendio en valparaíso, Abril 2014. http://google.org/crisismap/2014-valparaiso-fire (Última visita Sept. 20, 2016).

[15] Pedro Reszka and Andrés Fuentes. The great valparaiso fire and fire safety management in chile. *Fire Technology*, 51(4):753–758, 2015.

[16] Nidamarthi Srinivas and Kalyanmoy Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary computation*, 2(3):221–248, 1994.

[17] Liong Choong Yeun, Wan Rosmania Ismail, Khairuddin Omar, and Mourad Zirour. Vehicle routing problem: models and solutions. *Journal of Quality Measurement and Analysis*, 4(1):205–218, 2008.