

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA
VALPARAÍSO - CHILE



“RESOLUCIÓN DEL BUS EVACUATION PROBLEM CON
ANT COLONY OPTIMIZATION PARA LA COMUNA DE
SANTO DOMINGO”

RENATO BURGOS HIDALGO

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN INFORMÁTICA

Profesor Guía: María Cristina Riff
Profesor Correferente: Nicolás Rojas Morales

Noviembre - 2024

DEDICATORIA

Dedicada a mi tata que en paz descansa.

AGRADECIMIENTOS

Quiero agradecer a mi familia por acompañarme durante este proceso, y por ayudarme a formarme como persona. Les agradezco todas las enseñanzas y lecciones que me dieron y que me seguirán dando.

Agradezco también a mis amigos, quienes han estado conmigo en momentos difíciles y con quienes siento dicha de compartir y reír.

Por último, agradezco a todas las personas que han sido parte de mi vida y de mi desarrollo, he llegado a estas instancias gracias a ustedes.

RESUMEN

Resumen— El *Bus Evacuation Problem* (BEP) consiste en encontrar las mejores rutas que permitan a un grupo de buses evacuar a todas las personas necesarias en medio de una catástrofe, de modo que se encuentren a salvo en refugios y minimizando los tiempos de transporte. Junto con el *Ant Colony Optimization* (ACO), el cual se basa en la comunicación entre hormigas por medio de feromonas, se encontrarán las mejores rutas posibles para la evacuación de personas en medio de un desastre. Este trabajo se enfocará en la resolución del problema para la comuna de Santo Domingo, Chile, y debido a la proximidad con el borde costero, las zonas de riesgo serán las posibles zonas afectadas por un tsunami.

Palabras Clave— Optimización computacional, *Bus Evacuation Problem*, *Ant Colony Optimization*.

ABSTRACT

Abstract— The *Bus Evacuation Problem* (BEP) consists in finding the best possible routes that allow a group of buses to evacuate every person amidst of a disaster, in order that they are safe in shelters with a minimum transportation time. Along with the *Ant Colony Optimization* (ACO), which is based on the communication between ants through pheromones, the best possible routes for people's evacuation amidst of a disaster will be found. This work will focus on solving the problem for the Santo Domingo commune, Chile, and due to the proximity to the coast, the danger zones will be the possible zones affected by a tsunami.

Keywords— Computational optimization, *Bus Evacuation Problem*, *Ant Colony Optimization*.

GLOSARIO

BEP: Bus Evacuation Problem

CEP: Casualty Evacuation Problem

DBEP: Dynamic Bus Evacuation Problem

IBEP: Integrated Bus Evacuation Problem

MOVRP: Multi-Objective Vehicle Routing Problem

RBEP: Robust Bus Evacuation Problem

SBRP: School Bus Routing Problem

SDMDVRPI: Split Delivery Multi-Depot Vehicle Routing Problem with Inter-Depot Routes

MPMDVRPTWHF: Multiple Pickup and Multiple Delivery Vehicle Routing Problem With Time Window and Heterogeneous Fleets

TSP: Travelling Salesman Problem

UAV: Unmanned Aerial Vehicles

VRP: Vehicle Routing Problem

ACO: Ant Colony Optimization

LSO: Link-based System Optimization

PSO: Path-based System Optimization

HC: Hill Climbing

SENAPRED: Servicio Nacional de Prevención y Respuesta ante Desastres

ÍNDICE DE CONTENIDOS

RESUMEN	III
ABSTRACT	III
GLOSARIO	IV
ÍNDICE DE FIGURAS	VII
ÍNDICE DE TABLAS	VII
INTRODUCCIÓN	1
CAPÍTULO 1: DEFINICIÓN DEL PROBLEMA	3
1.1 Elementos del problema	5
1.2 Ejemplo	5
1.3 Problemas Similares	7
1.4 Variantes del problema	8
CAPÍTULO 2: MARCO CONCEPTUAL	10
2.1 Aplicaciones del problema	10
2.2 Modelo Matemático	14
2.3 El Algoritmo ACO	17
2.3.1 Movimiento	17
2.3.2 Actualización de feromonas	18
CAPÍTULO 3: PROPUESTA DE SOLUCIÓN	20
3.1 Algoritmo	20
3.1.1 Representación	22
3.1.2 Etapas de Construcción	23
3.1.3 Cambios de Rutas	25
3.2 Instancias	26
3.2.1 Instancias Aleatorias	27
3.2.2 Instancias Santo Domingo	28
3.3 Parámetros	30
CAPÍTULO 4: VALIDACIÓN DE LA SOLUCIÓN	32
4.1 Entorno de desarrollo	32
4.2 Rendimiento ACO	32
4.2.1 Análisis de <i>Alpha</i> y <i>Beta</i>	32
4.2.2 Análisis de Iteraciones	33
4.2.3 Análisis de Feromonas	35
4.3 Resultados	36
4.3.1 Comparación con solver	36

4.3.2 Caso de estudio	40
CAPÍTULO 5: CONCLUSIONES	44
REFERENCIAS BIBLIOGRÁFICAS	47

ÍNDICE DE FIGURAS

1	Vista del problema	6
2	Planificación de buses en Kaiserslautern, Alemania	11
3	Plan de evacuación en una emergencia nuclear	11
4	PSO y LSO en una evacuación de tsunami	12
5	Evacuación a pie en Penco, Chile	14
6	Mejora de la solución	17
7	Movimiento de una hormiga	18
8	Diagrama de flujo para ACO y el BEP	21
9	Representación de la Solución	22
10	Mapa de Santo Domingo	28
11	Gráfico de calidad de soluciones con distintas iteraciones	34
12	Ruta del quinto bus	43
13	Evolución del camino más largo en cada iteración	44

ÍNDICE DE TABLAS

1	Planificación de buses	6
2	Parámetros de las instancias aleatorias	27
3	Refugios de Santo Domingo	29
4	Instancias de Santo Domingo	30
5	Resultados de ACO BEP con distintos α y β	33
6	Resultados de ACO BEP con distintas iteraciones	34
7	Resultados de ACO BEP con distintos valores de q	36
8	Resultados de ACO BEP y Solver desde S_1 a S_4	38
9	Resultados de ACO BEP y Solver desde S_5 a S_8	39
10	Resultados de ACO BEP y Solver de S_9	40
11	Resultados de ACO BEP y Solver para las instancias de Santo Domingo en minutos	41
12	Ruta para 20 buses	42

INTRODUCCIÓN

Las catástrofes, ya sean naturales o provocados por el hombre, suelen provocar tragedias tanto humanas como materiales. Aunque los daños materiales son casi inevitables, la evacuación para evitar pérdidas humanas es donde los recursos y esfuerzos son prioridad. Elaborar un plan de evacuación es por esto muy complejo, ya que todas las personas damnificadas por una catástrofe deben ser evacuadas lo más rápido posible. Para esto también se debe tener en cuenta la zona damnificada, el método de transporte en que las personas serán movilizadas y el lugar de destino en que las personas se encuentren a salvo. Para encontrar una planificación o ruta con tales condiciones, resulta algo muy difícil de optimizar computacionalmente, de modo que al introducir una nueva variable al problema se vuelve exponencialmente más complejo de optimizar.

Tras esto, en el año 2011 [Bish, 2011], se planteó un problema de optimización computacional que ocuparía técnicas de inteligencia artificial, llamado *Bus Evacuation Problem* (BEP). Es un problema en el que encontrar su óptimo se vuelve muy complejo al no haber un método que lo encuentre a partir de cualquier circunstancia. Es por esto que se debe realizar una búsqueda iterativa sobre las posibles soluciones, aproximando los resultados con a veces algoritmos no deterministas, o bien con soluciones que pueden ser subóptimas. Este problema se inspira en el *Vehicle Routing Problem*, el cual consiste en encontrar una ruta para que un vehículo recorra todos los objetivos de la forma más óptima posible; similar al VRP, el BEP debe encontrar una ruta en que se minimiza el tiempo de evacuación, pero se debe tener en cuenta la distancia a las zonas de peligro, las personas a evacuar en tales zonas y la distancia a cada punto de refugio. Se debe rescatar a todas las personas en una cantidad finita de buses, los cuales tienen una capacidad máxima de pasajeros. Los puntos de refugio también tienen una capacidad máxima, y si ese número se supera durante la evacuación, no se puede albergar más gente en ese punto y se debe ir a otro refugio.

Ciertos elementos de la evacuación pueden no estar disponibles, o bien podrían demorar por mucho tiempo la evacuación total. El tráfico que puede darse en tales catástrofes es muy relevante para ciertos problemas, y si no se toman precauciones puede conllevar a una evacuación más lenta de lo previsto. Debido a esto, la determinación de los refugios dependiendo de la catástrofe es muy relevante para el problema, omitiendo ciertos puntos de refugio y optando por otros más cercanos a las zonas de peligro.

Al ser un problema complejo de resolver, se han propuesto distintas soluciones a lo largo de los años, como la de [Bish, 2011], en la que se propone un algoritmo de búsqueda local para resolver el problema, o bien la de [Goerigk y Grün, 2012], en la que se propone un algoritmo de búsqueda tabú para resolver el problema. En este trabajo se propone una solución utilizando el *Ant Colony Optimization*, el cual es un algoritmo de optimización inspirado en la forma en que las hormigas se comunican entre ellas para encontrar la ruta más corta a un objetivo. En este algoritmo, las hormigas dejan rastros de feromonas en los caminos que recorren, para que en siguientes iteraciones las hormigas se guíen tanto por el rastro

de las feromonas como por la distancia entre caminos, y así encontrar la ruta más corta a un objetivo. Este algoritmo ha sido utilizado en problemas de optimización, especialmente para encontrar las rutas más cortas, asimilando el método de las hormigas. Desde su introducción en [Bullnheimer *et al.*, 1999] se ha combinado con distintos problemas similares, dejando buenos resultados en poca cantidad de iteraciones.

Además, debido al gran desafío de realizar una planificación correcta para zonas costeras en riesgo de ser afectadas por algún tsunami, y por las chances de sufrir una catástrofe de tales características en Chile, se estudiará en este documento una instancia a resolver de la comuna de Santo Domingo, Chile.

Este documento contiene 5 capítulos. El *Capítulo 1: Definición del problema*, profundiza las características más importantes del BEP y se analiza el caso de la comuna de Santo Domingo, Chile. Luego, en el *Capítulo 2: Marco conceptual*, se describen los aportes desarrollados hasta el momento por otros investigadores, y se discutirá cuáles son adecuados de incorporar en el contexto del problema al tratar. En el *Capítulo 3: Propuesta de solución* se presentará un modelo con el cual se busca llegar a una solución, así como la metodología a utilizar junto a las variables y parámetros para obtener resultados relevantes. Luego de haber implementado la propuesta, el *Capítulo 4: Validación de la solución*, hablará del por qué la solución obtenida es relevante y consistente con lo que se tenía planteado. Por último, el *Capítulo 5: Conclusiones*, contendrá las principales enseñanzas y reflexiones del documento, así como el posible trabajo a futuro que puede llegar a realizarse.

CAPÍTULO 1

DEFINICIÓN DEL PROBLEMA

El BEP es un problema muy relevante para las organizaciones encargadas de conllevar las evacuaciones de emergencia alrededor del mundo, ya que su principal objetivo es el de salvar vidas arriesgando muy poco en la práctica. Se deben evacuar a todas las personas que se encuentren en zonas críticas a través de buses, los que se encargan de transportar a estas personas a zonas de refugio, todo esto en el menor tiempo posible. Como hay una cantidad de buses finita, se deben aprovechar de buena manera para realizar una evacuación exitosa.

Se conoce que un tsunami puede ocurrir 15 minutos después de un movimiento telúrico, por lo que evacuar a una zona segura puede ser muy complejo en tan poco tiempo; por otro lado, una ola se conoce que puede alcanzar un máximo de 30 metros de altura. Estos casos extremos pueden ser catastróficos para gente que debe evacuar en muy poco tiempo, o bien en zonas menores a 30 metros sobre el nivel del mar. Por fortuna, la provincia de San Antonio y la comuna de Santo Domingo, así como muchas costas de Chile, se caracterizan por tener cerros cerca de las zonas costeras donde la evacuación puede ser muy rápida y expedita, ya que se puede evacuar a pie sin la necesidad de vehículos. Otro método es el de evacuar a edificios altos, los cuales puedan soportar una ola de gran magnitud. Ambos tipos de evacuación son aplicados en Chile y en el mundo:

- **Evacuación vertical:** el objetivo de esta evacuación es subir lo más alto posible de donde se encuentra el foco o el peligro. Es aplicado en edificios preparados y capacitados para soportar tornados, tsunamis, incendios, etc. En Chile se ocupa este tipo de evacuación como último recurso, pues es incierto si algunos edificios son capaces de soportar la ola de un tsunami.
- **Evacuación horizontal:** es el tipo de evacuación que recomienda el SENAPRED (Servicio Nacional de Prevención y Respuesta ante Desastres) para tener una evacuación exitosa. Consiste en alejarse lo más lejos posible del peligro, lo que en este caso de estudio sería lo más lejos de la costa. Esto implica evacuar a los cerros, para así estar muy alto sobre el nivel del mar, o bien evacuar a una zona muy lejana del mar.

En Chile la evacuación horizontal es la más utilizada, y en la que se enfocará este estudio. El tipo de evacuación horizontal más utilizada es la de evacuar a cerros (la que considera la evacuación a pie hasta llegar a una zona segura), ya que se aprovecha la altitud de un territorio para que el oleaje no alcance aquellas zonas. El segundo tipo de evacuación horizontal sin embargo, es más problemática porque mucha gente evacúa con sus vehículos, lo que en muchos casos ha provocado congestión vehicular, siendo aquellas personas alcanzadas por el oleaje o por el fuego. Ocurrió algo parecido en la comuna de Santo Domingo en Chile para el terremoto de 2010, y es que mucha gente quiso evacuar por el puente de Santo Domingo,

generando una gran congestión vehicular imposibilitando el paso de los mismos. Este puente se encuentra sobre la desembocadura del río Maipo y muy cercano del mar, por lo que la presencia de un tsunami pudo haber causado víctimas fatales.

Este tipo de evacuación se ha llevado a cabo en distintas ciudades de Chile, como es el caso de Lota [lot, 2011] y puede ser muy importante para personas con discapacidad o bien para grandes grupos de gente como escolares, aunque la evacuación a pie en muchos casos puede ser suficiente y más rápida de realizar. En aquellas zonas más planas y donde un tsunami alcance a muchas personas sería una buena opción evacuar de esta forma, aunque deben haber caminos despejados para que los vehículos de emergencia puedan pasar con normalidad, lo que implica que la gente no debe ocupar sus vehículos para evacuar. Esta es una suposición muy grande al ser una de las principales maneras de evacuación para muchas personas, aunque una de las más fatales también, como ocurrió en verano de 2024 en Viña del Mar [Townsend, 2024]. SENAPRED no recomienda evacuar en vehículos, y tampoco tiene contemplada la evacuación en buses ante un tsunami, pero es una manera que puede ser muy efectiva en otras partes del mundo o para otros contextos, como la evacuación ante erupciones volcánicas.

Circunstancias similares han ocurrido un par de veces en la provincia de San Antonio, Chile, la cual fue fuertemente damnificada por los terremotos de 1985 en Algarrobo [Araneda *et al.*, 2006] y el de 2010 en Concepción [Quezada *et al.*, 2012]. En el terremoto de 1985 fue el derrumbe del antiguo Puente Lo Gallardo, el cual conectaba Santo Domingo con San Antonio y que fue una gran pérdida para el transporte local. Mientras que para el terremoto de 2010, hubieron daños al nuevo Puente Lo Gallardo, haciendo muy difícil el traslado entre ambos territorios. Ambas situaciones fueron muy desafortunadas, y es posible que en otro terremoto ocurra una situación parecida. Por esto, el principal caso de estudio e inspiración para realizar este trabajo se basa en la comuna de Santo Domingo, donde se busca tener una evacuación exitosa en caso de un tsunami, y donde se contempla tanto la evacuación a pie como una vía válida de evacuación, así como la evacuación en buses para personas con discapacidad o bien para grandes grupos de gente.

Además de tratarse de un problema de gran envergadura al querer tener los mejores resultados posibles, es un problema computacionalmente muy complejo de optimizar, esto es al tratarse de uno *NP-Complete* como queda demostrado en [Goerigk y Grün, 2012], al igual que otros problemas similares como el VRP.

Para llegar a obtener soluciones se debe explorar un espacio de búsqueda, lo cual se puede realizar con distintas técnicas de Inteligencia Artificial, para luego modificar aquellas soluciones que entreguen mejores resultados. Tal espacio de búsqueda puede crecer exponencialmente a medida que se introduce una nueva variable (como un nuevo refugio), por lo que llegar a encontrar la mejor solución dentro de todo ese espacio se hace muy complejo. Es por esto que es importante encontrarse con buenas soluciones iniciales que puedan ser mejoradas, pero soluciones que difícilmente lleven al óptimo no vale la pena mejorarlas; esto es conocido como un óptimo local, y evitar la exploración en tales puntos puede ahorrar mucho tiempo de ejecución.

1.1. Elementos del problema

En el problema se identifican los siguientes elementos, los cuales definirán la instancia del problema a resolver:

- **Evacuados:** el número de personas que se encuentran en peligro y que deben ser trasladadas.
- **Buses:** medios de transporte que tienen una capacidad máxima de pasajeros, con todos ellos teniendo la misma capacidad.
- **Estaciones:** puntos donde los buses parten y donde no se puede volver una vez parten su recorrido. Esto es porque el camino de vuelta puede ser peligroso para los choferes, junto que retrasa el tiempo de evacuación total.
- **Puntos de encuentro:** puntos donde se encuentran las personas a evacuar. Es una zona de riesgo por lo que las personas no se pueden quedar ahí, y se deben evacuar a todas ellas. Los buses tampoco deben quedarse en estos puntos, ya que pueden ser alcanzados por el peligro.
- **Refugios:** lugares seguros donde deben ser llevadas las personas evacuadas. Los refugios también tienen capacidad limitada y deben albergar a muchas personas. Posibles refugios son hospitales, gimnasios o escuelas.

Para el caso de la comuna de Santo Domingo, ya se conocen los 7 puntos de encuentro para que las personas ante una emergencia conozcan muy bien el procedimiento y el camino; se realizan simulacros constantemente para que los habitantes conozcan ambos elementos y así estar preparados para realizar una buena evacuación. Sin embargo, los refugios en este caso no son conocidos, siendo estos principalmente colegios, gimnasios, hospitales y espacios donde se puede albergar mucha gente. Al tener tantos refugios disponibles, es importante utilizar puntos cercanos a los puntos de encuentro, y siempre teniendo en cuenta la capacidad máxima de los mismos para no sobrepasar el aforo.

1.2. Ejemplo

Con las variables ya definidas, es posible crear un ejemplo que permita visualizar de mejor manera el modelo y las interacciones. En [Goerigk *et al.*, 2013] hay un ejemplo con los siguientes casos:

- Número de personas en cada punto de encuentro: $l = (1, 3, 3)$.
- Capacidades de cada refugio: $u = (4, 4, 1)$.

- Distancias de las estaciones a los puntos de encuentro: $d^{ini} = (7, 4, 9)$.

Por otro lado, la matriz de las distancias entre los puntos de encuentro y los refugios son:

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

Con todos estos datos se puede crear un grafo para visualizar de mejor manera la situación. Este grafo es dirigido desde su única estación hasta los puntos de encuentro, pero se vuelve uno no dirigido desde los puntos de encuentro a los refugios. Un ejemplo es visto en la Figura 1.

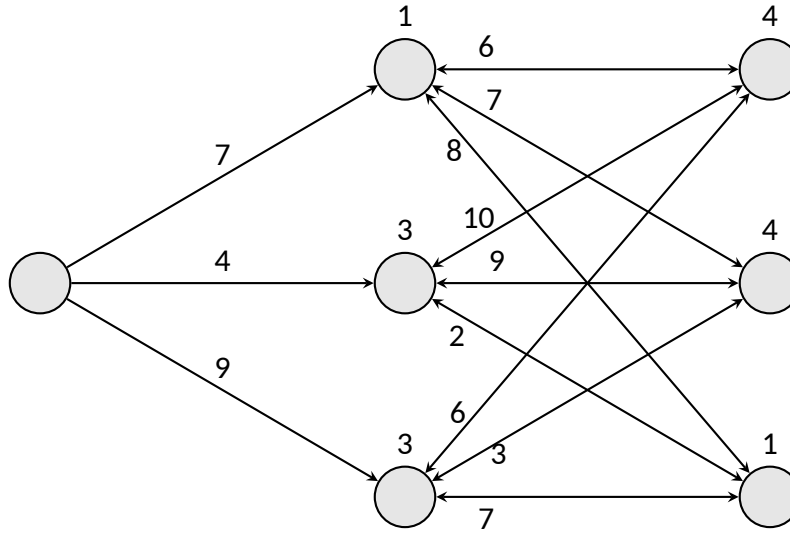


Figura 1: Vista del problema
Fuente: [Goerigk et al., 2013]

La solución óptima de esta instancia llevando a una persona por recorrido es la vista en la Tabla 1. En ella se tiene una serie de pares ordenados (*punto de encuentro, refugio*) con los viajes que realiza cada bus.

	Viaje 1	Viaje 2	Viaje 3
Bus 1	(1,1)	(3,2)	
Bus 2	(2,1)	(3,2)	
Bus 3	(2,3)	(2,2)	(3,2)

Tabla 1: Planificación de buses
Fuente: [Goerigk et al., 2013]

Se tiene además que dos buses tomaron el mismo camino y el tercero tomó otro, como es el caso para el primer bus que tomó el camino de 7 unidades (desde la primera estación hasta el primer punto de encuentro) y los demás tomaron el de 4 unidades (desde la primera estación hasta el segundo punto de encuentro). La decisión del primer bus es muy importante porque no tomó el camino más corto que sería el segundo con 4 unidades, sino que escogió el que más le conviene para tener la mejor solución. El primer bus por consiguiente fue desde el punto de encuentro 1 hasta el refugio 1 para luego ir desde el punto de encuentro 3 hasta el refugio 2, aunque primero tuvo que tomar el camino desde el refugio 1 hasta el punto de encuentro 3.

En total el primer bus demoró 22 unidades y trasladó a 2 personas. Los otros buses mientras tanto se demoraron 23 unidades; esta solución es ideal ya que los 3 buses tuvieron costos similares. Es importante mencionar que todos esos trayectos se pueden realizar de forma paralela, finalizando la evacuación en 23 unidades de tiempo. Es un caso contrario al de los 33 mineros en la mina de San José, Chile [Sougarret, 2020] donde se debía hacer solo un viaje al mismo tiempo y rescatar a solo una persona se tendría un tiempo total de 68, aunque claro que tal caso no aplica de la mejor manera al BEP.

1.3. Problemas Similares

Para la formulación del *Bus Evacuation Problem* se tomaron otros problemas como inspiración, pero difiere mucho de estos por distintas razones. Se identifican los más importantes a continuación:

- **Vehicle Routing Problem (VRP):** el Problema de Enrutamiento de Vehículos consiste en encontrar rutas óptimas para hacer entregas de la forma más rápida posible, como fue descrito en [Wenyi, 2023]. Es muy parecido al *Travelling Salesman Problem* (TSP), pero a diferencia de éste se tiene que lograr cumpliendo ciertas restricciones; tales son que un vehículo tiene una capacidad máxima de transporte, puede haber tráfico en el camino y el vehículo debe ir a una velocidad máxima. Se vuelve un problema tan variante incluso cuando ya se había encontrado una buena solución, ya que ésta puede cambiar mientras se realiza, volviéndolo un problema muy difícil de optimizar.
- **Split Delivery Multi-Depot Vehicle Routing Problem with Inter-Depot Routes (SDMDVRPI):** como su nombre lo indica, se encarga de entregar productos a compradores con vehículos que se encuentran en depósitos. Este problema se parece mucho al BEP, y es que se debe viajar desde ciertos puntos donde se recoge algo para luego entregarse a otros puntos, con los vehículos teniendo un límite de capacidad. La principal diferencia es que en SDMDVRPI se quiere minimizar costos mientras que en BEP se quiere minimizar el tiempo, dado que el primero es de un servicio de ventas y el segundo es de una emergencia donde se dispone de los costos de viaje suficientes. Este problema, descrito en [Bish, 2011] es uno que puede ser abarcado por servicios

de entrega de paquetes como Mercado Libre o Starken, o incluso por repartidores de comida.

- **Casualty Evacuation Problem (CEP):** este problema descrito en [Dikas y Minis, 2016] es muy similar al BEP, pero este consiste en el traslado de los damnificados que pueden ocurrir durante el accidente, a diferencia del *Bus Evacuation Problem* donde la evacuación puede ser para prevenir algún accidente. Esto lo vuelve un problema más complejo de optimizar porque se consideran los casos donde el número de personas a evacuar aumenta en el tiempo.
- **Multi-Objective Vehicle Routing Problem (MOVRP):** de acuerdo a [Molina et al., 2014], es un problema que consiste en encontrar la forma más efectiva de distribuir bienes a múltiples lugares, mientras que se logran tiempos mínimos de transporte y de entregas, así como la mínima emisión de los vehículos. Es importante mencionar este problema porque difiere en algunos elementos con el BEP, y es que en el MOVRP no se considera la cantidad de personas a trasladar, y el medio de transporte a utilizar es heterogéneo, es decir, puede haber diferencias en las características de los vehículos, algo que no siempre se toma en cuenta para el BEP. Por otro lado, en el BEP no se considera la emisión de los vehículos, algo que se podría tener en cuenta pero en el contexto de una catástrofe se puede evitar esta variante.
- **The multiple pickup and multiple delivery vehicle routing problem with time window and heterogeneous fleets (MPMDVRPTWHF):** este problema de logística, descrito en [Ky Phuc y Phuong Thao, 2021], consiste en atender distintos clientes más de una vez, con vehículos heterogéneos, los cuales tienen capacidad límite de transporte y con ventanas de tiempo para realizar las entregas. La principal diferencia con el BEP es que el MPMDVRPTWHF busca entregar pedidos específicos a clientes específicos en el momento exacto, aunque de todos modos minimizando los tiempos de transporte. En [Ky Phuc y Phuong Thao, 2021] se propone una solución implementando *Ant Colony Optimization* (ACO) para encontrar las mejores rutas posibles, por lo que su implementación resulta muy relevante para este estudio.

1.4. Variantes del problema

Para hacer el *Bus Evacuation Problem* un problema más específico para las ocasiones necesarias se crearon variantes que lo complejizan aún más, pero que al fin y al cabo valen la pena analizar y resolver. Se identifican las siguientes variantes más importantes:

- **Integrated Bus Evacuation Problem (IBEP):** en [Goerigk et al., 2014] se introduce una variante de BEP, la cual consiste en que antes de la evacuación y de elegir la ruta que seguirán los buses, se escojan los puntos de encuentro y los refugios que serán usados durante la evacuación.

- ***Robust Bus Evacuation Problem (RBEP)***: en [Goerigk y Grün, 2012] se explica que este enfoque consiste en que en el BEP no siempre se conoce el número de personas a evacuar. Después de un tiempo este número se conoce y el problema se hace más fácil de resolver.
- ***Dynamic Bus Evacuation Problem (DBEP)***: quizá la variante más relevante es la presentada en [Zhu, 2017], ya que toma en cuenta el hecho de que haya evacuación en vehículos personales por gran parte de la población. Aquellas personas que no tengan vehículos personales o tengan alguna discapacidad tienen como única opción la evacuación en buses. Es por esto que esta variante del BEP considera los cambios a haber durante la evacuación, principalmente el tráfico.

CAPÍTULO 2

MARCO CONCEPTUAL

Este capítulo tiene como objetivo presentar los conceptos y definiciones necesarios para entender el problema a abordar, así como las herramientas y técnicas que se utilizarán para resolverlo. Se presentarán los conceptos más relevantes del problema, las aplicaciones que se han dado a lo largo del tiempo, el modelo de programación entera mixta y la técnica de optimización que se utilizará para encontrar la mejor solución.

2.1. Aplicaciones del problema

El *Bus Evacuation Problem* es un problema relativamente nuevo, al ser introducido en el año 2011 en [Bish, 2011], y tomando como inspiración las catástrofes de Katrina en 2005, donde de acuerdo a [Berg, 2013] hubieron más de 1000 personas fallecidas por el huracán y la seguida inundación. Muchas de esas muertes pudieron haber sido evitadas si se hubiese llevado a cabo un buen plan de evacuación. Esto conllevó a la creación del mismo BEP, así como varias aplicaciones de éste en distintas circunstancias. A continuación se verán las más importantes realizadas hasta el momento, teniendo así una mejor comprensión del problema que se quiere abordar.

Uno de los casos más relevantes para el problema abordado es el explicado en [Goerigk et al., 2014], el cual tiene en consideración los casos en que el tráfico puede hacer más lenta la evacuación. Es por esto que el método propuesto sugiere determinar el punto de encuentro y a cuál refugio deben ir, para minimizar la distancia en posibles situaciones desfavorables. Para esto, se tienen muchos puntos de encuentro y refugios posibles, pero se elegirán los que entreguen la mejor solución encontrada. Esto, además, implica que no todos los refugios ni los puntos de encuentro se vayan a ocupar. Se toma en cuenta que esta operación puede empeorar el tiempo de ejecución, pero que al fin y al cabo encontrará mejores soluciones. La implementación de este trabajo fue realizada en la ciudad de Kaiserslautern, Alemania, y como se puede ver en la imagen 2, el círculo rojo corresponde a la zona donde podía haber la explosión de una bomba; es por esto que se determinaron los puntos de encuentro, los que están representados como cuadrados, mientras que los refugios corresponden a los triángulos.

Otro caso estudiado, y el cual es bastante interesante al ocupar DBEP como principal problema a resolver, es el planteado en [Chen et al., 2022], el cual mide constantemente la situación en que las personas a evacuar se encuentran. Esto es porque el contexto del problema es el de una posible emergencia nuclear, en la cual la radiación puede irse expandiendo rápidamente, y pudiendo ser fatal para muchas de las personas a evacuar. El plan a seguir es el visto en la figura 3, el cual por medio de dispositivos llamados *Unmanned Aerial Vehicles* (UAV) se mide la radiación, para luego actualizar el plan de evacuación si las condiciones son

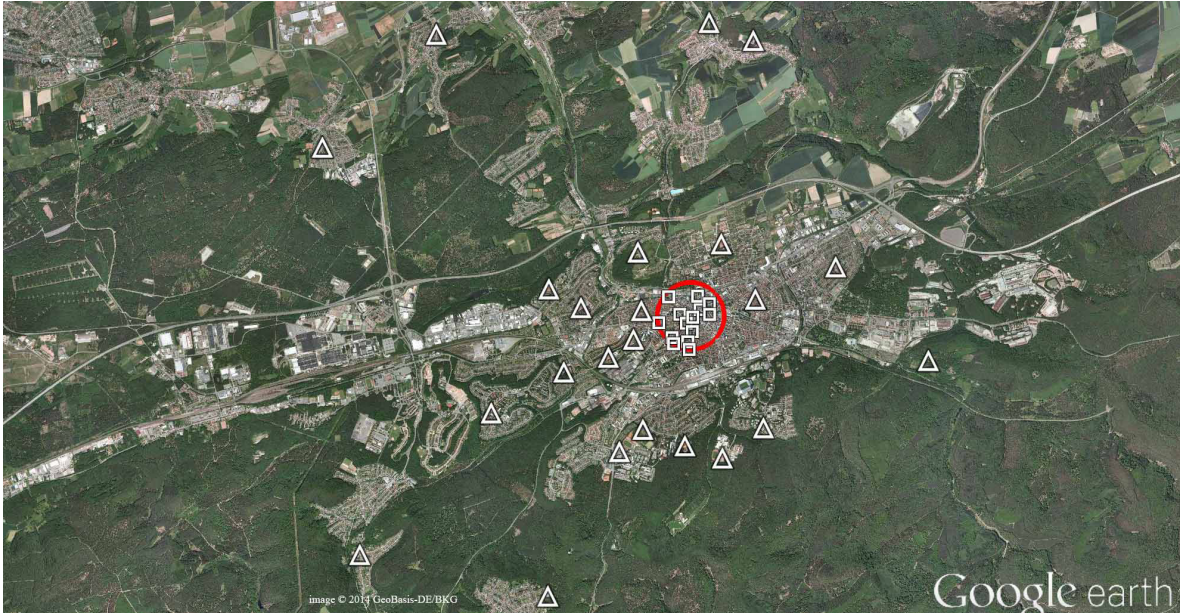


Figura 2: Planificación de buses en Kaiserslautern, Alemania
Fuente: [Goerigk et al., 2014]

muy críticas.

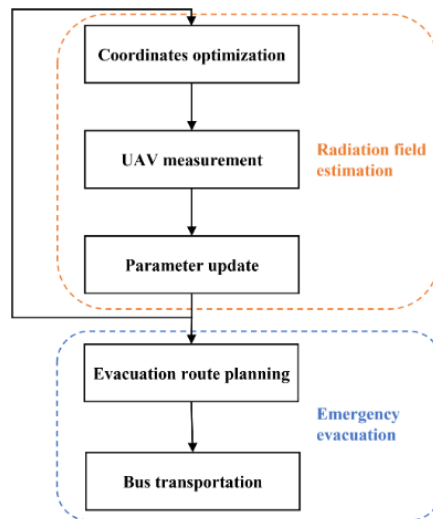


Figura 3: Plan de evacuación en una emergencia nuclear
Fuente: [Chen et al., 2022]

Un modelo de evacuación en dos etapas fue planteado por Zhenning Li en [Li et al., 2019], en el cual la primera etapa consiste en que las personas en peligro deben evacuar a pie a un refugio temporal, mientras que en la segunda etapa cuando la situación ya está más tranquila, las personas evacúan en grupo en automóviles a una zona segura. Cada etapa tiene un distinto tipo de optimización, aunque el de la primera etapa no es relevante para este estu-

dio; para la segunda etapa se tienen dos tipos de optimización, llamados *Link-based System Optimization* (LSO) y *Path-based System Optimization* (PSO). Ambos métodos consisten en la optimización del sistema como conjunto, ya que los vehículos influyen en el tráfico y, por lo tanto, en los tiempos de evacuación, a diferencia de la primera etapa donde las personas a pie no generan problemas entre sí y se puede tomar la ruta más corta para evacuar.

PSO se caracteriza por tener múltiples soluciones al haber distintos caminos para llegar al mismo objetivo, mientras que LSO tiene una solución única para llegar desde los refugios a las zonas seguras. Es importante remarcar que ambos métodos pueden tener la misma solución. Como PSO tiene en cuenta las calles y el tráfico que se va a formar en la evacuación, es más complejo modelar aquella situación, aunque es más aterrizada a la realidad. En la imagen 4 se puede ver la planificación de la evacuación en una ciudad.

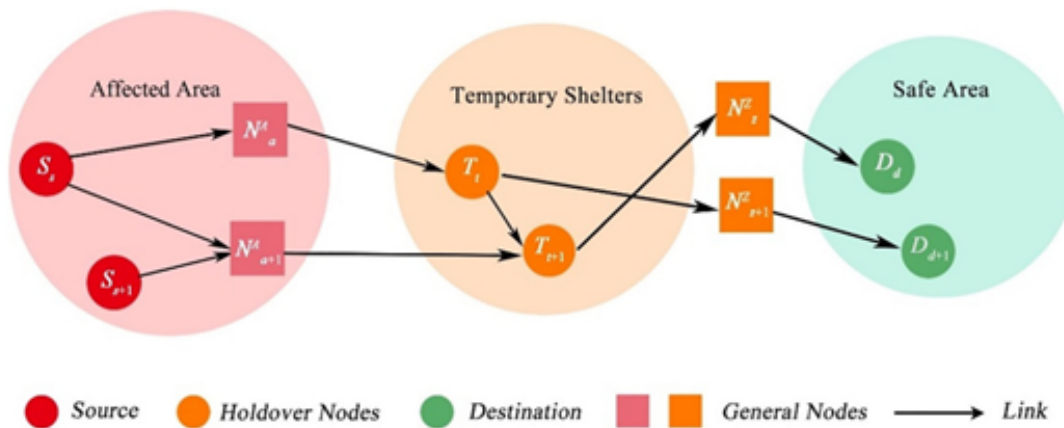


Figura 4: PSO y LSO en una evacuación de tsunami

Fuente: [Li et al., 2019]

Estos problemas son muy relevantes para entender el problema a abordar, siendo casos de estudio que pueden ser aplicados a distintas situaciones de emergencia. Además de estos casos, la técnica a estudiar en este trabajo, el *Ant Colony Optimization*, ha sido aplicada en distintos contextos y en distintos problemas de evacuación, como se verá en los casos siguientes.

Muy similar al caso posible a abordar, unos investigadores colombianos en [Arias-Rojas et al., 2012] investigaron sobre un caso de *School Bus Routing Problem* (SBRP) para una escuela de Colombia, en la cual se debía resolver tal problema y encontrar las mejores rutas para buses escolares. Tales buses recogían estudiantes en la mañana y los retornaban en la tarde. Aunque es muy parecido al BEP, este problema considera los gastos en realizar los trayectos, al igual que se plantea en un futuro tomar en cuenta las emisiones de CO2 de los buses, pero el BEP no considera tales factores en su planificación.

Por otro lado, la heurística utilizada para encontrar las rutas de los buses fue la *Ant Colony Optimization* (ACO), la cual entregó resultados positivos disminuyendo los tiempos de transporte en la mañana (8,3 %) y en la tarde (21,4 %). Las hormigas en este modelo equivalen

a los buses, las cuales se encargan de explorar distintas rutas en base al nivel de feromonas que se encuentran alrededor. Además, tienen movimientos a partir de probabilidades, para así explorar distintas rutas que aún no han sido exploradas. A pesar de que tienen un movimiento muy parecido al *greedy*, la red de feromonas que comparten las hormigas es la principal determinante de este modelo, ya que dependiendo de los resultados obtenidos se pueden guiar a caminos parecidos. Esto también permite tener soluciones en muy poco tiempo al no cambiar el estado de la ruta a medida que se crea.

Otro elemento importante de esta implementación es que los estudiantes son agrupados para así optimizar el número de viajes de los buses. De este modo, cuando un bus pase a recoger un grupo, éste se lleve a todos los estudiantes haciendo esto solo una vez. Básicamente, cada bus se encarga de recorrer el camino más corto que sería de resolver el TSP. Además, siempre habrá más buses que puntos a donde buscar a los estudiantes, algo que es posible en este caso al haber pocas personas por recoger.

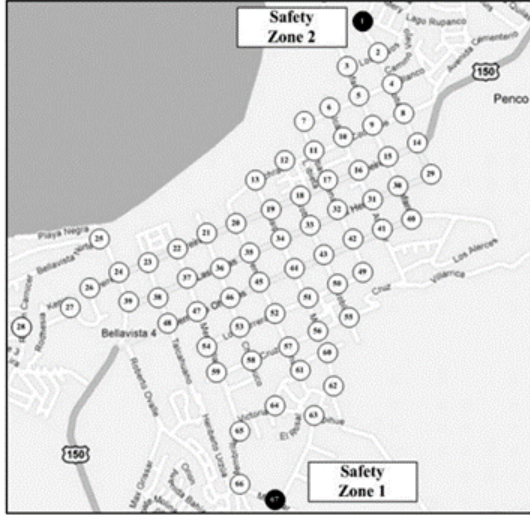
El último caso a revisar es uno aplicado en la ciudad de Penco, Chile. Tras el terremoto de 2010, investigadores en [Forcael et al., 2014] propusieron un modelo basado en la heurística de la colonia de hormigas para encontrar rutas de evacuación. Su implementación fue para la evacuación a pie hacia los cerros donde las personas evacuadas podrán encontrarse a salvo. Utilizaba “nodos” que se ubicarían en cada esquina de las calles para así analizar el número de nodos recorridos. Conociendo la distancia entre nodos se puede obtener la distancia total a recorrer por las personas que debían evacuar desde un nodo inicial, siendo en algunos casos más de 1500 metros.

Este estudio, además de ocupar las características más fundamentales descritas anteriormente, ocupa variantes muy interesantes para las feromonas y su influencia en la toma de decisiones. Las más relevantes son las matrices α y β ; la matriz α indica cuán seguro es un camino de i a j , mientras que la matriz β considera si un camino i a j es muy elevado o no. Para ambas matrices el valor 0.5 es un valor regular, donde en α al acercarse a 1 la condición del camino es peor o puede ser muy peligroso evacuar por ahí, mientras que en β un valor al acercarse a 0 indicará que la pendiente del camino es más empinada. Otras variables importantes son la cantidad de feromonas en un camino i a uno j , así como el coeficiente de evaporación de las feromonas, es decir, el radio en que las feromonas van evaporándose en los caminos. Estas variables pueden ser muy importantes para este estudio, y es que las feromonas pueden formar un rol importante en el movimiento de las hormigas, y los parámetros utilizados para la evacuación tales como las condiciones de los caminos (algo muy importante en un terremoto) resulta muy necesario para la evacuación a pie.

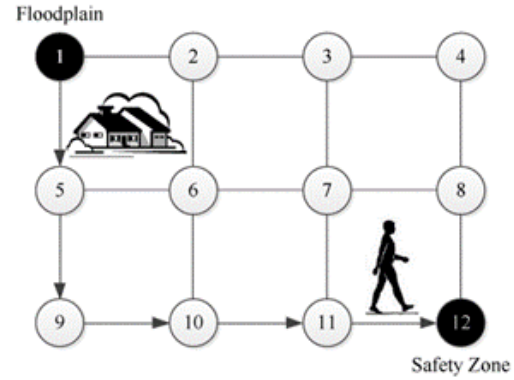
Este además fue un caso que fue aplicado con personas reales para testear las rutas, con resultados bastante interesantes. Se realizaron dos tipos de evacuación con 34 jóvenes: la primera fue una evacuación con ruta y la segunda fue sin ruta. Algunas personas conocían varios pasajes de la ciudad mientras que otras no, por lo que algunos de ellos tomaban caminos más cortos cuando no tenían ruta establecida, lo cual puede indicar que el algoritmo no entregó buenos resultados, pero tomaron caminos no indicados en la ruta para avanzar más rápido. Esto puede ser otro problema para este tipo de evacuación para personas con

movilidad reducida, algo que es contemplado por los autores para tener en cuenta a futuro.

En la imagen 5a se puede ver la planificación de la evacuación en Penco, Chile, junto con la figura 5b en donde se refleja el movimiento de las personas a pie en la evacuación.



(a) Mapa de Penco y sus zonas seguras



(b) Movimiento de peatones en la evacuación

Figura 5: Evacuación a pie en Penco, Chile

Fuente: [Forcael et al., 2014]

2.2. Modelo Matemático

Para empezar se deben definir los conjuntos con los que se trabajarán. Se tiene una red (N, A) , con N el conjunto de nodos y A los arcos. El conjunto N lo conforman las estaciones, los puntos de encuentro y los refugios. El último conjunto es el de los buses, V , donde todos tienen la misma capacidad Q . Ahora se deben plantear las variables, las restricciones que se deben cumplir dada la naturaleza del problema y el objetivo que se debe encontrar a partir de las soluciones factibles:

- **Variables:** corresponde a todo lo situacional del problema, como las estaciones Y , los puntos de encuentro P , y los refugios S . Es conveniente definir el número de personas ubicadas en los puntos de encuentro, D_j , con $j \in P$ y la capacidad de los refugios C_i , con $i \in S$. La capacidad Q de los buses es una variable opcional. También se tienen los costos asociados a recorrer cada camino definidos como d_{ij} . Cada arco $(i, j) \in P, S$ tiene un costo d_{ij} desde el nodo i al nodo j , es decir, el costo en recorrer desde el punto de encuentro i hasta el refugio j , así como desde un refugio i hasta un punto de encuentro j . Mientras que para los arcos (i, j) con $i \in Y$ y $j \in P$ se tienen costos

que no son bidireccionales, es decir, equivalen a los costos de ir solamente desde una estación a un punto de encuentro.

Se tienen, además, las siguientes variables de decisión:

- x_{ij}^{mt} : variable binaria que equivale a 1 cuando el bus m escoge el viaje t , recorriendo el arco (i, j) , de lo contrario 0, $\forall (i, j) \in A, m \in V$ y t un instante cualquiera de la evacuación, siendo un número entero mayor o igual a 0.
 - b_j^{mt} : número de personas evacuadas desde el punto de encuentro j asignando al bus m en el viaje t , $\forall j \in N, m \in V$ y t un instante cualquiera de la evacuación, también siendo un número entero mayor o igual a 0.
 - T_{evac} : la duración total de la evacuación.
- **Objetivo:** encontrar el recorrido más corto para los buses. Se deben evaluar las soluciones factibles y seleccionar aquella que minimice el tiempo necesario para evacuar, es decir, se debe minimizar T_{evac} .
 - **Restricciones:** todas las personas deben ser llevadas a los refugios. Se deben respetar las capacidades de los buses y de los refugios.

Para resolver este problema es importante considerar un modelo de programación entera mixta. Las restricciones más relevantes de este modelo, de acuerdo a [Thapa y Khadka, 2016] son:

- La función objetivo 1 consiste en minimizar el tiempo de evacuación total.

$$\text{Minimizar } T_{evac} \quad (1)$$

- En 2, el tiempo de evacuación total debe ser mayor o igual al costo máximo de los caminos recorridos.

$$T_{evac} \geq \sum_{(i,j) \in A} \sum_{t=1}^T d_{ij} x_{ij}^{mt}, \forall m \in V \quad (2)$$

- En 3, el bus que viaja al nodo j en el instante t deja tal nodo en el instante $t + 1$.

$$\sum_{i:(i,j) \in A} x_{ij}^{mt} = \sum_{k:(j,k) \in A} x_{jk}^{m(t+1)}, \forall j \in P, m \in V, t = 1, \dots, T - 1 \quad (3)$$

- En 4, los buses pueden quedarse en el refugio en vez de volver a un punto de encuentro.

$$\sum_{i:(i,j) \in A} x_{ij}^{mt} \geq \sum_{k:(j,k) \in A} x_{jk}^{m(t+1)}, \forall j \in S, m \in V, t = 1, \dots, T - 1 \quad (4)$$

- En 5, un bus no puede hacer dos viajes al mismo tiempo.

$$\sum_{(i,j) \in A} x_{ij}^{mt} \leq 1, \forall m \in V, t = 1, \dots, T \quad (5)$$

- En 6, todos los buses empiezan su recorrido en el instante $t = 1$.

$$x_{ij}^{m1} = 1, \forall i \in Y, j : (i, j) \in A, m \in V_i \quad (6)$$

- En 7, si un bus no empieza su recorrido al primer instante no lo puede empezar después.

$$x_{ij}^{mt} = 0, \forall i \in Y, j : (i, j) \in A, m \in V, t = 2, \dots, T \quad (7)$$

- En 8, el último viaje de un bus no puede terminar en un punto de encuentro.

$$x_{ij}^{mT} = 0, \forall j \in P, i : (i, j) \in A, m \in V \quad (8)$$

- En 9, un bus puede recoger personas de un nodo j si es que visitó tal nodo.

$$b_j^{mt} \leq \sum_{(i,j) \in A} Qx_{ij}^{mt}, \forall j \in N, m \in V, t = 1, \dots, T \quad (9)$$

- La restricción 10 representa la capacidad de los buses.

$$0 \leq \sum_{j \in P} \sum_{l=1}^t b_j^{ml} - \sum_{k \in S} \sum_{l=1}^t b_k^{ml} \leq Q, \forall m \in V, t = 1, \dots, T \quad (10)$$

- La restricción 11 representa la capacidad de los refugios.

$$\sum_{m \in V} \sum_{t=1}^T b_j^{mt} \leq C_j, \forall j \in S \quad (11)$$

- La restricción 12 asegura que todas las personas se subieron a un bus.

$$\sum_{m \in V} \sum_{t=1}^T b_j^{mt} = D_j, \forall j \in P \quad (12)$$

- La restricción 13 asegura que todas las personas llegaron a un refugio.

$$\sum_{j \in P} \sum_{t=1}^T b_j^{mt} = \sum_{k \in S} \sum_{t=1}^T b_k^{mt}, \forall m \in V \quad (13)$$

- La restricción 14 asegura que esta variable tenga solo valores binarios.

$$x_{ij}^{mt} \in \{0, 1\}, \forall (i, j) \in A, m \in V, t = 1, \dots, T \quad (14)$$

- Por último, la restricción 15 asegura que esta variable tenga valores enteros no negativos.

$$b_j^{mt} \geq 0, \forall j \in N, m \in V, t = 1, \dots, T \quad (15)$$

Con respecto a la función objetivo, el problema consiste en minimizar los tiempos de evacuación; para esto, se asume que todos los buses comienzan su recorrido al mismo tiempo, y el bus que más tiempo se tarda en terminar es el que debe ser minimizado.

2.3. El Algoritmo ACO

El algoritmo ACO es el principal método en la resolución del problema, el cual es uno de los métodos basados en *swarm intelligence* (inteligencia de enjambre). Consiste en un algoritmo probabilístico en que los agentes (hormigas) evalúan sus posibles movimientos, teniendo mayores probabilidades de elegir aquellos que tienen más feromonas depositadas. Las feromonas son rastros que dejan las hormigas al recorrer un camino, y que se van evaporando con el tiempo. La cantidad de feromonas en un camino es muy influyente en la elección de las hormigas, teniendo mayor probabilidad de ser elegidos aquellos caminos con más feromonas. Este movimiento estocástico es muy útil para poder escapar de mínimos locales y encontrar mejores soluciones, guiándose además por los caminos más cortos disponibles.

La mejora de la solución se puede apreciar en la figura 6, donde en (a) muchas hormigas toman el camino más largo ($A \rightarrow D$), mientras que en (b) más hormigas van escogiendo el mejor camino ($A \rightarrow C$), haciendo que hayan más feromonas depositadas en él, para que finalmente en (c) sea evidente para las hormigas cuál es el camino más corto.

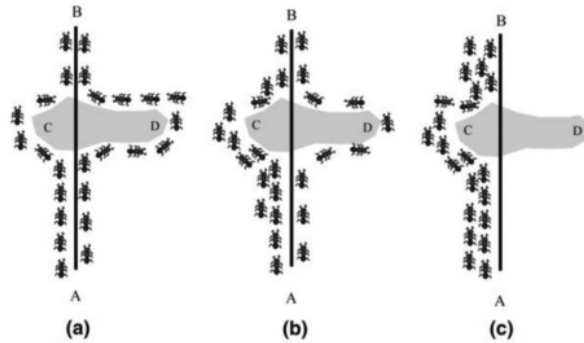


Figura 6: Mejora de la solución
Fuente: [Forcael et al., 2014]

De acuerdo al trabajo de [Forcael et al., 2014], el algoritmo ACO se puede dividir en las etapas de movimiento y actualización de feromonas, las cuales se describen en las siguientes secciones.

2.3.1. Movimiento

Debido a la naturaleza estocástica del algoritmo, se tiene que las hormigas no siguen un camino fijo, sino que eligen su camino basándose en la cantidad de feromonas depositadas en los caminos. Este movimiento se puede observar en la figura 7, donde la hormiga k en el nodo i puede elegir el camino j , g o l dependiendo del número de feromonas que se encuentren en los caminos.

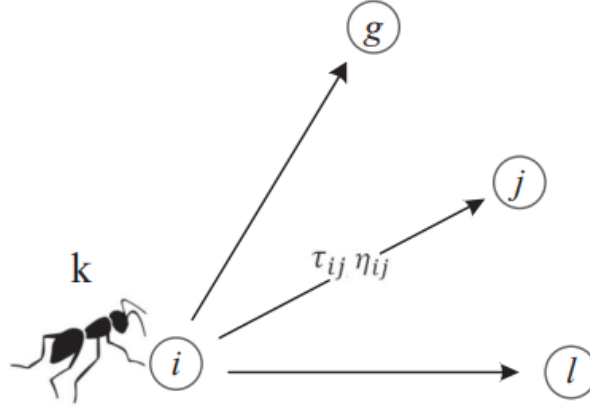


Figura 7: Movimiento de una hormiga

Fuente: [Forcael et al., 2014]

La probabilidad de que la hormiga k en el nodo i vaya a otro nodo está dada por la ecuación 16.

$$p_{ij}^k = \frac{(\tau_{ij})^\alpha (\eta_{ij})^\beta}{\sum_{l \in N} (\tau_{il})^\alpha (\eta_{il})^\beta} \quad (16)$$

En la ecuación 16, τ_{ij} es la cantidad de feromonas en el camino (i, j) , η_{ij} es el valor inverso de la distancia entre los nodos i y j ($1/d_{ij}$), siendo un valor heurístico, mientras que α y β son parámetros que controlan cuán influyentes son τ_{ij} y η_{ij} , respectivamente. Estos dos últimos parámetros son muy importantes, y es que cuando β es mucho mayor a α , significa que las feromonas no son relevantes, por lo que el movimiento sería elegido solamente por la distancia entre los nodos. Por otro lado, si α es mucho mayor a β , entonces la distancia no es relevante, y el movimiento sería elegido solamente por las feromonas depositadas. Es por esto que se deben escoger valores que sirvan tanto para explotar distintas soluciones como para explorar nuevas soluciones. Una visualización de los posibles movimientos de una hormiga se puede ver en la figura 7.

2.3.2. Actualización de feromonas

A medida que el algoritmo avanza, las feromonas depositadas en los caminos se van actualizando, de manera que los caminos más cortos tengan más feromonas depositadas, y así sean más atractivos para las hormigas. Para esto, debe haber tanto evaporación de las feromonas como una actualización de las mismas. La actualización de las feromonas, que ocurre cuando las hormigas terminan su trayecto, se puede ver en la ecuación 17, donde τ_{ij}^{old} corresponde a las feromonas antes de la actualización y τ_{ij}^{new} el nuevo valor de las feromonas.

$$\tau_{ij}^{new} = (1 - \rho)\tau_{ij}^{old} + \Delta\tau_{ij} \quad (17)$$

En la ecuación 17, ρ ($0 \leq \rho \leq 1$) es la tasa de evaporación de las feromonas. Cuando ρ es cercano a 0, las feromonas se evaporan lentamente, mientras que si es cercano a 1, las feromonas se evaporan muy rápido. Por otro lado, $\Delta\tau_{ij}$ es la cantidad de feromonas que se depositan en el camino (i, j) . Esto es formalizado en la ecuación 18.

$$\Delta\tau_{ij} = \begin{cases} q/L_k, & \text{si la hormiga } k \text{ usa el camino } (i, j) \\ 0, & \text{en caso contrario} \end{cases} \quad (18)$$

En la ecuación 18, q es una variable que representa la cantidad de feromonas depositadas por cada hormiga, mientras que L_k es la longitud del camino recorrido por la hormiga k . Si la hormiga k no recorre el camino (i, j) , entonces no se depositan feromonas en él.

Algunas fuentes sugieren distintos valores para los distintos parámetros mencionados, como en [Ma et al., 2017], pero difieren dependiendo en cada contexto.

CAPÍTULO 3

PROPUESTA DE SOLUCIÓN

Tras ya conocer el problema y entender su complejidad, es necesario invocar los métodos más importantes para llegar a sus soluciones. Se introduce en este capítulo el algoritmo basado en *Ant Colony Optimization* y las instancias que se utilizarán para la experimentación y su resolución.

3.1. Algoritmo

Para resolver el *Bus Evacuation Problem* se propone un algoritmo basado en *Ant Colony Optimization*, donde en cada iteración se guardan las rutas creadas por las hormigas para luego ser modificadas y escoger la mejor solución. El algoritmo se detiene cuando se alcanza un número máximo de iteraciones, y en cada iteración se reinician las hormigas y se realiza la evaporación y el depósito de las feromonas discutidas en el capítulo anterior. El diagrama de flujo para resolver el BEP con ACO está representado en la figura 8.

En 1 se puede ver el pseudocódigo del algoritmo principal, donde ocurre la construcción de las soluciones y la modificación de las rutas para obtener la solución final.

Algorithm 1 Main Algorithm

```
while  $iteration < maxIterations$  do
     $result \leftarrow GenAllPaths()$ 
     $pheromones \leftarrow (1 - decay) * pheromones + SpreadPheromones()$ 
     $results \leftarrow results + result$ 
end while
for  $result$  in  $results$  do
     $ShiftMovement(result)$ 
end for
 $bestSolution \leftarrow max(results.longestPath)$ 
return  $bestSolution$ 
```

La función `SpreadPheromones()` es la encargada de depositar las feromonas en los caminos, y se realiza en cada iteración del algoritmo al ya haber generado las rutas de evacuación. Para tener un buen desempeño en las siguientes iteraciones del algoritmo, se depositan feromonas en todos los caminos recorridos por las hormigas. Esto es porque se busca que en cada iteración se añadan más feromonas de las que se evaporan, y realizar un elitismo en que sólo la mejor hormiga contribuya a la actualización de las feromonas no es lo ideal. Además, se deben depositar feromonas solo en los caminos de puntos de encuentro a refugios y viceversa, ya que como se discutirá más adelante, estos son los caminos que permiten explo-

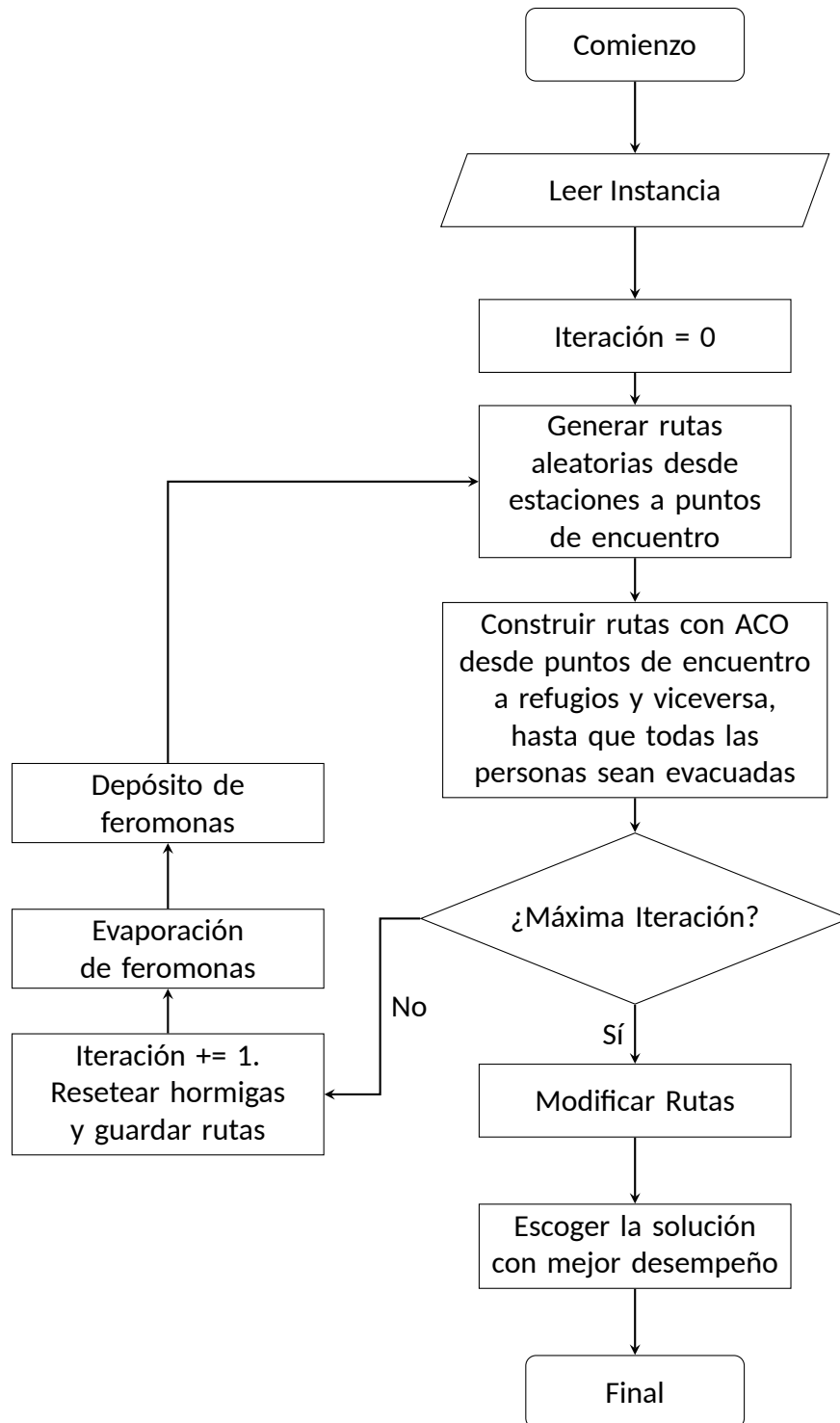


Figura 8: Diagrama de flujo para ACO y el BEP
Fuente: Elaboración Propia.

rar mejores soluciones. Como se mencionó en el capítulo anterior, la cantidad de feromonas depositadas en los caminos es de acuerdo a la ecuación 17.

Hay ciertas consideraciones que fueron muy relevantes en la implementación del algoritmo que se diferencian de otras implementaciones de ACO, las cuales serán explicadas en las siguientes secciones.

3.1.1. Representación

La representación de la solución es un aspecto muy importante en la implementación del algoritmo, ya que debe ser legible y fácil de modificar. En esta representación inspirada por [Loyola, 2016], cada fila representa la ruta de un bus, donde cada viaje V_m es representado como un par de nodos (i, j) , con i el nodo de partida y j el nodo de llegada. En 9 se muestra la representación de la solución, para una instancia de n buses con 3 estaciones, 4 puntos de encuentro y 5 refugios.

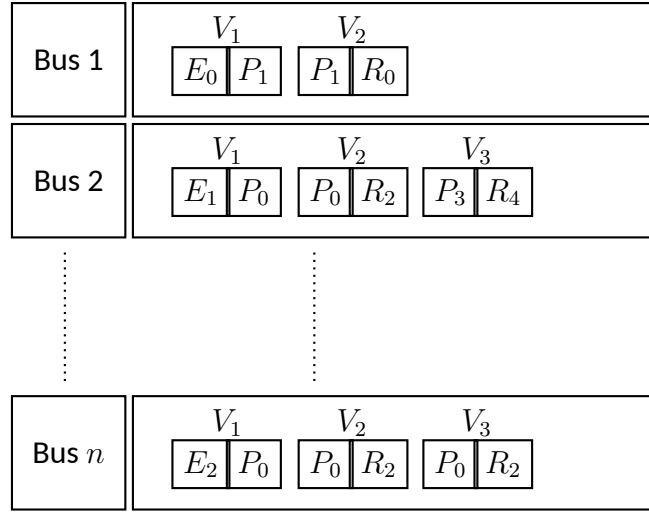


Figura 9: Representación de la Solución
Fuente: Elaboración Propia.

El primer viaje V_1 de cada bus es desde una estación E_i hasta un punto de encuentro P_j , lo que es representado como un viaje (E_i, P_j) . El siguiente viaje V_2 ocurre siempre desde el mismo punto de encuentro P_j hasta un refugio R_k , lo que es representado como (P_j, R_k) . Ya para los siguientes viajes V_m , también se tienen viajes desde puntos de encuentro a refugios, y se asume que hubo un viaje de vuelta desde el refugio anterior al siguiente punto de encuentro. Esto es porque ya construidas las rutas de los buses, es posible modificarlas para obtener mejores soluciones sin romper las restricciones del problema.

Esta modificación es un movimiento *shift* que se profundizará más adelante, y permite que siempre desde un recorrido (P_j, R_k) se rescaten las personas en el nodo j y sean alojadas en el refugio k , cambiando solamente el bus que realiza el viaje y el instante en que se ejecuta. La distancia recorrida cambiará sin embargo, ya que al modificar la ruta se deben calcular nuevas distancias entre refugios y puntos de encuentro.

3.1.2. Etapas de Construcción

Para la primera etapa de construcción, se generan rutas desde las estaciones a los puntos de encuentro utilizando un movimiento completamente aleatorio en cada iteración, es decir, sin depender de las feromonas. Esto es porque en esta primera etapa se condicionaría mucho el resultado final, y el objetivo es que las hormigas no caigan en mínimos locales de forma tan temprana. Explorar distintas rutas en esta etapa es importante para que las hormigas no escojan siempre caminos subóptimos, sino que exploren distintas rutas que serán influyentes en la siguiente etapa. El número de hormigas en esta etapa es igual al número de buses disponibles, por lo que las hormigas cumplen el rol de buses en la evacuación.

En la segunda etapa de construcción, donde se generan las rutas desde los puntos de encuentro a los refugios, se utilizará el algoritmo ACO para que influyan tanto las feromonas como la aleatoriedad. En esta etapa, las hormigas se guiarán por las feromonas depositadas en los caminos, pero también influirá el comportamiento estocástico para que las hormigas exploren rutas aleatorias, lo que permitirá que no se queden en mínimos locales y exploren rutas que no necesariamente parezcan las mejores en un principio.

Un elemento muy importante del algoritmo es que hasta que todas las personas no sean evacuadas no se detendrá la construcción de rutas. Para lograr esto, todas las hormigas primero deben completar sus viajes desde un punto de encuentro a un refugio, y si no hay más personas por evacuar se detiene el algoritmo. Luego, si aún quedan personas por evacuar, entonces solo las hormigas necesarias harán viajes a evacuar personas; esto es porque los buses no pueden terminar su recorrido en puntos de encuentro y no deben realizar viajes innecesarios.

En el algoritmo 2 se muestra la función que ocurre en cada iteración para crear las rutas de evacuación, donde ocurren ambas etapas de construcción.

Las 3 funciones que determinan el movimiento de las hormigas son `PickMoveRandomly()`, `PickMoveShelter()` y `PickMoveMeetingPoint()`, las cuales tienen distinto comportamiento y son fundamentales para la construcción de las rutas.

- `PickMoveRandomly()`: Esta función elige un movimiento aleatorio para las hormigas. Tiene prioridad solamente la distancia entre nodos, y no se consideran las feromonas depositadas en los caminos. Además, solo puede ir a aquellos puntos de encuentro que tengan personas por evacuar, lo que es posible aún en etapas tempranas de la construcción.
- `PickMoveShelter()`: Esta función elige un movimiento desde un punto de encuentro a un refugio, y es utilizada en la segunda etapa de construcción de rutas. Ocupa las feromonas depositadas en los caminos para guiar a las hormigas, usando la ecuación 16 para escoger el siguiente nodo. También debe escoger refugios que tengan capacidad para recibir a las personas evacuadas.

Algorithm 2 GenAllPaths()

```
for station in stations do                                ▷ Desde una Estación a un Punto de Encuentro
  for ant in busesInStation do
    move  $\leftarrow$  PickMoveRandomly()
    antPath  $\leftarrow$  antPath + move
  end for
end for
while evacuation not complete do
  for ant in ants do                                    ▷ Desde un Punto de Encuentro a un Refugio
    move  $\leftarrow$  PickMoveShelter()
    antPath  $\leftarrow$  antPath + move
  end for
  for ant in ants do                                    ▷ Desde un Refugio a un Punto de Encuentro
    move  $\leftarrow$  PickMoveMeetingPoint()
    if move == NULL then
      for antsLeft in ants do                            ▷ Hormigas que no han llegado a un refugio
        move  $\leftarrow$  PickMoveShelter()
        antPath  $\leftarrow$  antPath + move
      end for
      return ants
    end if
    antPath  $\leftarrow$  antPath + move
  end for
end while
return ants
```

- **PickMoveMeetingPoint()**: Esta función elige un movimiento desde un refugio a un punto de encuentro, y es muy similar a **PickMoveShelter()**. Igualmente ocupa las feromonas depositadas en los caminos para determinar su siguiente movimiento y debe escoger aquellos puntos que no estén vacíos.

Un aspecto importante de la construcción de rutas es que habrán instantes en que hayan caminos obligatorios por tomar, ya que pueden haber refugios que no tengan capacidad para recibir a las personas evacuadas o puntos de encuentro sin personas por evacuar. En esas circunstancias el movimiento basado en feromonas y probabilidades no será relevante, provocando que las hormigas depositen feromonas en caminos que no son muy transcurridos. Esto es muy importante en la construcción de las soluciones, ya que puede que haya soluciones distintas en cada iteración. De todos modos se obtendrán resultados con rutas más cortas que otras, y se espera que la función reparadora del algoritmo pueda mejorar estas soluciones.

3.1.3. Cambios de Rutas

La segunda implementación es la de modificar rutas creadas por las hormigas para mejorar la solución. A pesar de que se tienen buenas rutas de evacuación con la implementación del ACO, hay ciertas limitancias con el formato de las soluciones que alejan a las hormigas de generar la mejor solución. Por ejemplo, en un principio, todos los buses intentarán realizar la misma cantidad de viajes en su recorrido, lo que no se puede conocer con certeza si es el mejor acercamiento, como algunos buses podrían realizar más viajes que otros para tener una solución óptima. Como esto requiere de mucha aleatoriedad y no hay un método eficaz que implemente esta condición, se propone un cambio en las rutas de evacuación de los buses para mejorar la solución. Este movimiento *shift* no quiebra ninguna restricción al no cambiar la cantidad de personas evacuadas en cada punto de encuentro, así como la capacidad de los refugios del plan de evacuación.

Cada solución creada por el algoritmo es modificada, y se escoge la mejor solución de todas las modificaciones para ser la solución final. Esto además implica que no siempre la mejor ruta generada por el algoritmo ACO sea la mejor solución, ya que ciertos viajes pueden ser más eficientes en ciertas rutas de buses que en otras. La exploración del algoritmo ACO, generada tanto por la aleatoriedad como el depósito de feromonas, que permite a las hormigas encontrar buenas rutas, es complementada por el movimiento *shift* implementado.

En 3 se muestra el pseudocódigo del movimiento *shift*, que está basado en la heurística de *Hill Climbing* (HC) para encontrar mejores soluciones.

Este proceso ocurre por cada solución creada por el algoritmo ACO, y se repite hasta que no se encuentren mejores soluciones. Con este movimiento la ruta de los buses es modificada, y por lo tanto el tiempo de evacuación de los buses también sufrirá cambios. El objetivo es modificar la trayectoria del bus más ineficiente (el que demora más tiempo en evacuar) y revisar si hay buses que pueden realizar un viaje de este bus para mejorar su tiempo de evacuación. Se toma el último viaje del bus más ineficiente, y se intenta insertar en todas las posiciones posibles de las otras rutas, escogiendo la mejor solución de todas las modificaciones. Se realiza una búsqueda exhaustiva de todas las posibles combinaciones de viajes para encontrar una mejor solución, buscando en todos los buses y en todas las posiciones posibles de las rutas. Si la modificación de la ruta de un bus provoca que otro bus tenga una ruta más ineficiente, entonces esa modificación no se realiza y se sigue buscando una mejor solución para otro bus. Esto es porque tener una peor solución que la anterior no asegura que se encuentre una mejor solución en siguientes iteraciones, y el objetivo es mejorar constantemente la calidad de las soluciones.

Esta implementación está inspirada en la heurística de *Hill Climbing* con mejor mejora, y es que se evalúa la mejor solución de las modificaciones realizadas a cada bus. Si hay una mejora para un bus, entonces se actualizan las rutas y se vuelve a evaluar si hay una mejor solución para otro bus. Si no la hay, entonces se sigue buscando una mejor solución para otros buses, y se repite el proceso hasta que no se encuentren mejoras en las soluciones. Cuando no llega

Algorithm 3 ShiftMovement()

```
resetWhile  $\leftarrow$  True
while resetWhile do
    resetWhile  $\leftarrow$  False
    longestPath  $\leftarrow$  max(paths) ▷ Encontrar la ruta más larga
    lastTrip  $\leftarrow$  TakeLastTrip(longestPath)
    for path in paths, path  $\neq$  longestPath do
        while p < NumberOfTripsInPath(path) - 1 do
            newPath  $\leftarrow$  path ▷ Copiar camino
            InsertTripInNewPath(lastTrip, NumberOfTripsInPath(newPath) - p)
            newPaths  $\leftarrow$  newPaths + newPath
        end while
        bestPath  $\leftarrow$  BestPath(newPaths)
        if bestPath < longestPath then
            UpdatePaths()
            resetWhile  $\leftarrow$  True
            break
        end if
    end for
end while
return paths
```

a haber mejoras para ningún bus, entonces esta iteración se termina y se continúa con las siguientes iteraciones.

3.2. Instancias

Para la experimentación se utilizaron instancias aleatorias y reales de la comuna de Santo Domingo, con el fin de evaluar el desempeño del algoritmo propuesto en distintos escenarios. Cada instancia tiene parámetros que influyen en su creación, siendo estos el número de estaciones (E), el número de puntos de encuentro (P), el número de refugios (R), el número de buses disponibles (B) y la capacidad de los buses (C). Cada instancia pertenece a un set de instancias, en la cual comparten los mismos parámetros y se diferencian en la capacidad de los puntos de encuentro y refugios, así como el número de personas a evacuar y la distancia entre cada nodo. Las instancias se denotan de esta manera: $S_n - m$, donde n es el número del set y m es el identificador de la instancia.

3.2.1. Instancias Aleatorias

En la tabla 2 se muestran las características de las instancias aleatorias creadas para la experimentación.

Set	E	P	R	B	C
S_1	1	4	2	4	20
S_2	1	5	3	6	20
S_3	2	9	7	5	36
S_4	2	12	3	6	20
S_5	2	22	4	10	20
S_6	2	32	5	18	20
S_7	3	11	10	7	36
S_8	5	25	12	15	36
S_9	8	40	20	20	36

Tabla 2: Parámetros de las instancias aleatorias

Cada set tuvo 8 instancias aleatorias, por lo que fueron creadas 72 instancias en total. Las distancias entre las estaciones, puntos de encuentro y refugios también fueron generadas aleatoriamente, dentro de un rango de 1 a 10. Los buses en cada estación, así como el número de personas en los puntos de encuentro y la capacidad de los refugios también fueron generados aleatoriamente, siendo estos últimos múltiplos de C para simplificar la experimentación.

El número de personas a evacuar en cada instancia es determinante en el rendimiento del algoritmo, ya que la evacuación de muchas personas provoca que hayan más viajes por realizar y, por lo tanto, tomará más tiempo encontrar una solución. En las instancias pequeñas se debe evacuar alrededor de 100 personas, mientras que en las instancias del set 9 se deben evacuar cerca de 2000 personas. Estos son números que se asemejan a la cantidad de personas que se deben evacuar en zonas de riesgo, y la variación de este factor permite evaluar el desempeño del algoritmo en distintos escenarios.

Los 4 primeros sets tienen instancias pequeñas que son sencillas de resolver para el *solver*, ya que el bajo número de buses y estaciones vuelven a las instancias más fáciles de resolver. El resto de los sets tienen instancias más grandes, con más estaciones y buses disponibles, por lo que el problema se vuelve más complejo y se requiere de más tiempo para encontrar una solución. Las diferencias en las instancias de un mismo set pueden volver a algunas instancias más complejas que otras, por lo que las técnicas metaheurísticas como ACO pueden tener un mejor desempeño en la resolución de estas instancias.

3.2.2. Instancias Santo Domingo

Para el caso de estudio de la comuna de Santo Domingo, es necesario utilizar información real de la comuna para evaluar el desempeño del algoritmo propuesto. La municipalidad de Santo Domingo realiza constantemente simulacros de evacuación, tanto de tsunamis como de incendios forestales, por lo que se dispone de información para modelar un caso aplicable a esta comuna. Los puntos de encuentro son conocidos al ser parte del plan de evacuación de la comuna, al igual que las zonas de riesgo de tsunami.

En la figura 10 se muestra el mapa de las Rocas de Santo Domingo, donde se pueden ver las estaciones en color azul, los puntos de encuentro en color verde y los refugios en color rojo, mientras que la zona de riesgo de tsunami también es de color rojo.

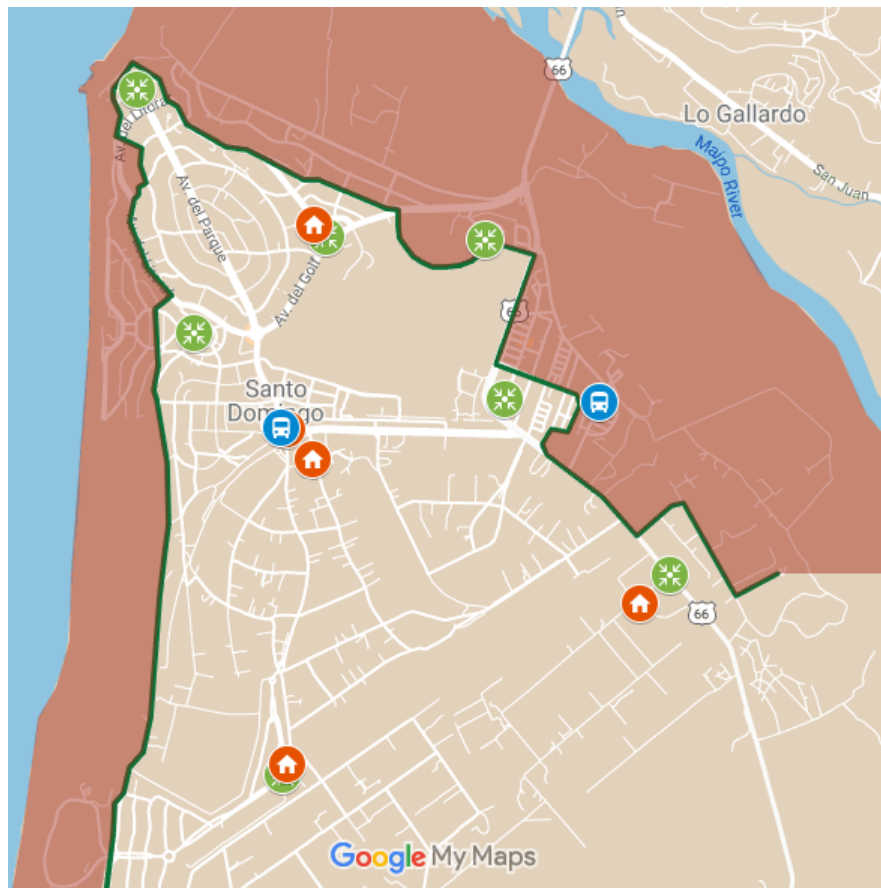


Figura 10: Mapa de Santo Domingo

Fuente: Elaboración Propia.

Las distancias entre los puntos fueron obtenidas a partir de la plataforma de Google My Maps, la que permite dejar marcadores en un mapa y obtener la distancia entre ellos. A partir de estas distancias se obtuvo el tiempo que demora un vehículo en recorrer sus trayectos, y se utilizaron estos tiempos para crear las instancias. A continuación se presentan los elementos de las instancias de Santo Domingo.

- **Zona de Riesgo:** la zona de riesgo de tsunami es determinada por SENAPRED, y corresponde a aquellas áreas donde hay menos de 30 metros sobre el nivel del mar. Abarca desde el borde costero hasta la desembocadura del Río Maipo, donde se encuentran muchos habitantes de la comuna y se tienen señalizaciones de evacuación a zonas seguras.
- **Estaciones:** ambas estaciones son importantes al haber buses de acercamiento en cada una de ellas. La primera, la municipalidad de Santo Domingo, es donde los buses de empresas privadas se reúnen para iniciar sus trayectos, mientras que la segunda estación es donde los buses de acercamiento comienzan sus recorridos en verano.
- **Puntos de Encuentro:** los puntos de encuentro de la comuna son 7, los cuales son de conocimiento público y establecidos por SENAPRED. Algunos puntos de encuentro son plazas, colegios o espacios abiertos en que las personas se encuentren a salvo de un tsunami.
- **Refugios:** se identifican 5 puntos comunitarios que podrían ser utilizados como refugios en caso de emergencia, y se encuentran en la tabla 3. Estos refugios tienen una capacidad máxima de personas que pueden albergar, siendo de 80 para las iglesias y capillas y de 120 para los colegios y escuelas.

Refugio	Capacidad
Colegio Helen Lee Lassen	120
Municipalidad de Santo Domingo	160
Escuela Waldorf	120
Iglesia Pentecostal de Nazareth	80
Capilla Santa María de la Paz	80

Tabla 3: Refugios de Santo Domingo

- **Personas Evacuadas:** aunque no hay información de cuántas personas se encuentran en las zonas de riesgo, se puede aproximar conociendo el número de personas que viven en la comuna. Según el censo de 2017, las Rocas de Santo Domingo tiene una población aproximada de 7238 personas con 876 hectáreas de superficie en zonas habitables. A partir de esto, y sabiendo que la zona de riesgo de tsunami ocupa aproximadamente 300 hectáreas, se puede estimar que hay alrededor de 2500 personas en la zona de riesgo.

Como varias de las 2500 personas puede que se queden en casas de conocidos o familiares, se estima que alrededor de 500 personas sean necesarias de evacuar con buses. Esto es un número aproximado, ya que esta comuna se caracteriza por tener más de 10000 personas en verano o en fines de semana largo, lo que aumenta la cantidad de personas a evacuar en caso de emergencia.

- **Buses:** el número de buses disponibles tampoco es conocido, por lo que ésta será una variante para evaluar el desempeño del algoritmo, y evaluar con qué número de buses se puede lograr evacuar a todas las personas en un tiempo razonable.

A partir de los elementos de las instancias de Santo Domingo, se crearon 4 instancias para evaluar el desempeño del algoritmo con distintos números de buses disponibles. En la tabla 4 se muestran las instancias de Santo Domingo creadas para la experimentación, con las 2 estaciones, 7 puntos de encuentro, 5 refugios y con buses de capacidad para 20 personas. Se realizarán experimentos con 5, 10, 15 y 20 buses disponibles.

Instancia	E	P	R	B	C
1	2	7	5	5	20
2	2	7	5	10	20
3	2	7	5	15	20
4	2	7	5	20	20

Tabla 4: Instancias de Santo Domingo

3.3. Parámetros

Una de las principales características de los algoritmos ACO es la cantidad de parámetros para su configuración, lo que permite obtener buenas soluciones en poco tiempo, así como también obtener soluciones óptimas en tiempos más largos. Es por esto que la configuración de los parámetros es un aspecto importante para el rendimiento del algoritmo, y sus ajustes no debiesen ser aleatorios. De acuerdo a [Ma *et al.*, 2017], los parámetros óptimos de su algoritmo ACO fueron $\alpha = 2$, $\beta = 5$, $q = 80$ y $\rho = 0,8$. Esta configuración resultó exitosa en la resolución de las instancias al dejar buenos resultados, aunque fueron modificadas para el presente trabajo en pos de mejorar el rendimiento. Para la ejecución del algoritmo ACO se utilizaron los siguientes parámetros:

- α : 2
- β : 4
- q : 4
- ρ : 0.1

Esto quiere decir que la influencia de las feromonas en la elección de los caminos es un poco menor que la influencia de la distancia entre nodos, como el valor de *alpha* es menor que el de *beta*. Esto permite a las hormigas explorar más rutas aleatorias y no depender tanto de las feromonas depositadas en los caminos. Se tiene además que el valor de las feromonas depositadas q no es tan alto, lo que permite que las feromonas depositadas en los caminos no sean tan grandes y no condicionen tanto la elección de los caminos. Por último, el valor de la evaporación de las feromonas ρ es muy cercano a 0, logrando que las feromonas decaigan lentamente en cada iteración, y por ende, se mantengan en los caminos por más tiempo.

El número de iteraciones por otro lado fue variando para cada instancia, ya que habían instancias que con muy pocas iteraciones se tenían resultados insastifactorios. Se limitó el número de iteraciones para que se entreguen resultados en tiempos acotados y razonables, de modo que se asemejaran a los tiempos en que el *solver* resolvió las instancias. En pequeñas instancias se usaron 400 iteraciones, mientras que para instancias más grandes se utilizaron 2000, 3000 y 4000 iteraciones.

CAPÍTULO 4

VALIDACIÓN DE LA SOLUCIÓN

Con el objetivo de validar la solución propuesta, se realizaron experimentos computacionales para comparar los resultados obtenidos con el algoritmo ACO con los de la literatura. Para ello, se utilizarán las instancias aleatorias para comparar el rendimiento del algoritmo ACO con los resultados obtenidos por un *solver* de programación entera mixta.

4.1. Entorno de desarrollo

Para la validación de la solución propuesta se utilizó un computador con un procesador Intel Core i5 2.90 GHz, memoria RAM 32.0 GB y sobre el entorno *Windows Subsystem for Linux* con Ubuntu 22.04.4 LTS. Los resultados fueron comparados con los obtenidos por el *solver* Gurobi a partir del modelo de programación entera mixta de [Thapa y Khadka, 2016]. Además, se utilizó Python para la implementación del algoritmo ACO, junto con librerías como *numpy* para el manejo de arreglos y operaciones matemáticas, y *matplotlib* para la visualización de los resultados.

4.2. Rendimiento ACO

A pesar de que las soluciones del algoritmo ACO son mejoradas con el método inspirado en *Hill Climbing* y pareciera que la calidad de la solución es enteramente dependiente de tal método reparador, lo cierto es que el algoritmo ACO, encargado de crear soluciones iniciales, es muy importante para obtener soluciones de calidad. Tanto la aleatoriedad como la determinación por escoger caminos más cortos del algoritmo ACO son fundamentales para obtener buenas soluciones, ya que un espacio de búsqueda puede ser muy grande para que un método reparador como el de *Hill Climbing* encuentre buenas soluciones. Los resultados obtenidos por el algoritmo propuesto dependen de ambas etapas, y los parámetros de la primera etapa son determinantes en la calidad de las soluciones obtenidas. Por lo tanto, se analizará el rendimiento del algoritmo con los parámetros propuestos en la sección 3.3.

4.2.1. Análisis de *Alpha* y *Beta*

Para analizar el rendimiento del algoritmo ACO se realizaron experimentos con distintos parámetros, por lo que primero se analizará el rendimiento con distintos valores de *alpha* (α) y *beta* (β). Se realizarán experimentos en 8 instancias distintas con 10 experimentos cada una, obteniendo así el promedio total para comparar los resultados obtenidos con otros

parámetros. Se realizaron 10 iteraciones para cada experimento, de modo que se aprecie el rendimiento del algoritmo para encontrar soluciones en poco tiempo. Los resultados se pueden ver en la tabla 5.

Instancia	α	β	α	β	α	β	α	β
	1	1	2	2	2	4	5	5
$S_2 - 7$	13.7	13.4	13.3	13.4				
$S_3 - 7$	21.0	19.2	18.5	17.7				
$S_4 - 7$	43.0	39.8	39.1	39.0				
$S_5 - 7$	35.8	34.3	32.4	32.2				
$S_6 - 7$	28.0	25.4	24.3	23.3				
$S_7 - 7$	20.3	17.6	15.9	16.5				
$S_8 - 7$	24.0	22.0	20.2	19.4				
$S_9 - 7$	25.3	21.3	18.7	18.5				

Tabla 5: Resultados de ACO BEP con distintos α y β

Como se puede apreciar en la tabla 5, los resultados van convergiendo a valores cada vez más bajos cuando se aumentan los valores de α y β . Se tiene que los mejores rendimientos son para los valores de $\alpha = 2$ y $\beta = 4$, así como para $\alpha = 5$ y $\beta = 5$, presentando valores muy cercanos entre sí, teniendo variaciones de una unidad aproximadamente. No se puede decir lo mismo para los valores de $\alpha = 1$ y $\beta = 1$ y para los valores de $\alpha = 2$ y $\beta = 2$, ya que presentan diferencias de promedios más altos que con los otros valores de α y β . Esto ocurre a pesar de que los valores de α y β son iguales, por lo que aumentar ambos valores termina siendo determinante para obtener mejores resultados en pocas iteraciones. Los dos primeros casos de α y β presentan valores más altos en promedio, por lo que son menos eficientes que los otros valores de α y β , necesitando más iteraciones para obtener soluciones de calidad. El cambio en estos parámetros tampoco afecta el tiempo de ejecución del algoritmo, ya que todos los experimentos se realizaron en menos de 1 segundo.

Se puede concluir que los valores de $\alpha = 2$ y $\beta = 4$ y $\alpha = 5$ y $\beta = 5$ son los mejores para obtener soluciones de calidad en pocas iteraciones, ya que presentan los mejores resultados en promedio. Además, es importante aumentar los valores de α y β periódicamente para analizar el comportamiento del algoritmo y obtener mejores resultados. La diferencia en promedio que estos valores presentan no es muy grande, por lo que ambos valores se deben tomar en cuenta para obtener buenos resultados. Estos resultados se obtuvieron con el método de *Hill Climbing* para mejorar las soluciones, por lo que se puede decir que el algoritmo ACO, junto con la decisión de estos parámetros, es fundamental para obtener este rendimiento.

4.2.2. Análisis de Iteraciones

Ya con los valores de $\alpha = 2$ y $\beta = 4$, se analizará el rendimiento del algoritmo variando el número de iteraciones. De igual forma que en el análisis de α y β , se realizaron experimentos en 8 instancias distintas con 10 experimentos cada una, obteniendo así el promedio total para comparar los resultados obtenidos con distinto número de iteraciones. Los resultados se pueden ver en la tabla 6.

Instancia	Iteraciones			
	100	500	1000	2000
$S_2 - 7$	13.0	13.0	13.0	13.0
$S_3 - 7$	16.5	15.6	15.5	15.3
$S_4 - 7$	37.2	36.4	35.7	35.6
$S_5 - 7$	30.4	29.3	28.9	28.6
$S_6 - 7$	21.8	20.6	20.6	20.5
$S_7 - 7$	13.4	12.6	11.4	10.8
$S_8 - 7$	18.1	17.0	16.6	16.4
$S_9 - 7$	17.0	16.1	16.0	15.1

Tabla 6: Resultados de ACO BEP con distintas iteraciones

Como se puede apreciar en la tabla 6, la mayoría de los resultados van convergiendo a valores cada vez más bajos cuando se aumenta el número de iteraciones. Esto se debe a que el algoritmo ACO, además de tener más soluciones de las que puede escoger, va mejorando sus soluciones a medida que se ejecutan las iteraciones. Aunque la diferencia entre soluciones no es tan grande, como es en el caso de $S_2 - 7$ que se obtiene la misma solución en todas las iteraciones, hay una tendencia a obtener soluciones más bajas a medida que se aumentan las iteraciones. En 11 se puede apreciar la tendencia de los resultados obtenidos con distintas iteraciones.

La principal desventaja de aumentar las iteraciones es que aumenta el tiempo de ejecución del algoritmo, tardando aproximadamente 40 segundos en obtener los resultados con 2000 iteraciones. A pesar de esto, el algoritmo ACO termina siendo muy eficiente en la obtención de soluciones de calidad, ya que para 100 iteraciones se obtienen soluciones muy cercanas a las obtenidas con más iteraciones, y en un tiempo menor a 3 segundos. Por lo tanto, aumentar el número de iteraciones es importante para obtener las mejores soluciones, pero su tiempo de ejecución puede ser muy alto para obtener soluciones similares a las obtenidas con menos iteraciones.

4.2.3. Análisis de Feromonas

Por último, se analizará el rendimiento del algoritmo ACO variando los parámetros q y ρ , los cuales son muy relevantes en la actualización de feromonas. En cada iteración del algoritmo,

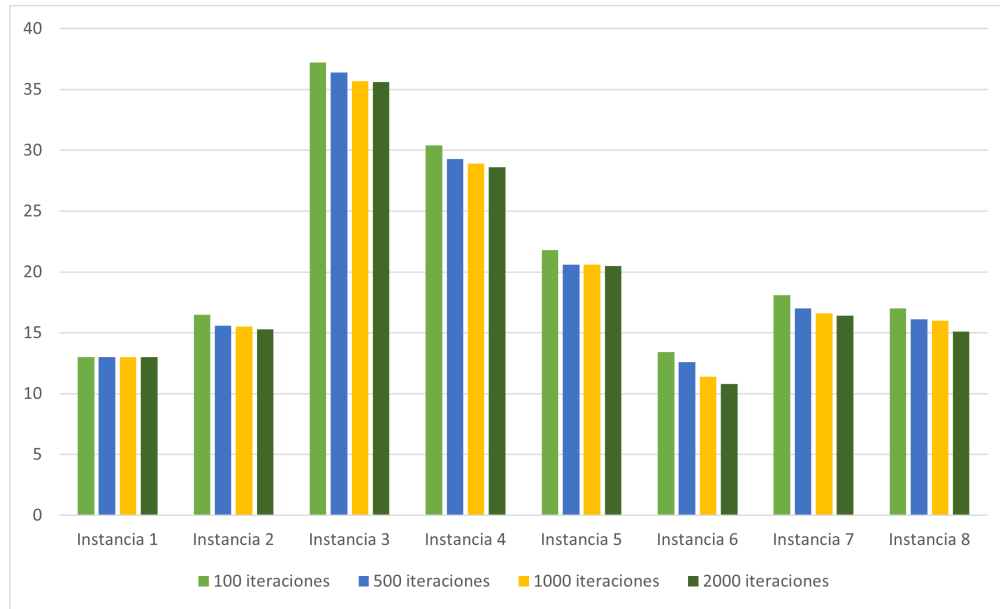


Figura 11: Gráfico de calidad de soluciones con distintas iteraciones

q representa la cantidad de feromonas que se depositan en los caminos recorridos por las hormigas, y ρ determina el radio en que las feromonas se evaporan en cada iteración.

Tras realizar experimentos, resultó que el valor de ρ es mucho más relevante que q para obtener soluciones de calidad, ya que es muy importante tener siempre feromonas en los caminos, más no solamente en aquellos caminos recorridos por las hormigas. Cuando $\rho \neq 1$ se obtienen soluciones de calidad, y el valor de q no termina siendo relevante para obtener tales soluciones, aunque debe ser mayor que 0; sin embargo, cuando $\rho = 1$ se obtienen soluciones muy bajas, ya que las feromonas se evaporan en cada iteración, y con $q = 0$ no habrán feromonas depositadas en los caminos, obteniendo las peores soluciones posibles.

Por lo tanto, se analizará la influencia de distintos valores de q cuando $\rho = 1$, ya que de esta forma se puede apreciar el comportamiento cuando hay pocas feromonas en los caminos. Así, en cada iteración se evaporan las feromonas y se depositan nuevas en aquellos caminos que recorren las hormigas. Tales feromonas serán influyentes en la siguiente elección de caminos, pero después de esto las feromonas vuelven a desaparecer. Para el análisis de q se utilizaron los valores $q = 0$, $q = 4$, $q = 100$ y $q = 10000$, y se realizaron 100 iteraciones por experimento con 8 instancias distintas. Los resultados se pueden ver en la tabla 7, donde se comparan los promedios de los experimentos realizados.

Como se puede apreciar, los resultados son bastante inconsistentes para las 3 primeras opciones, ya que hay instancias en que se obtienen soluciones mejores con $q = 0$ que con $q = 4$ y $q = 100$. No se puede decir lo mismo para $q = 10000$, dado que se obtienen soluciones mejores en todas las instancias. A pesar de esto, no se obtienen buenas soluciones con este método, a comparación de las soluciones obtenidas en la tabla 6. Por lo tanto, se puede concluir que el valor de q debe ser uno mayor a 0, y esta cantidad puede variar si se

Instancia	q			
	0	4	100	10000
$S_9 - 0$	17.4	16.5	17.5	15.1
$S_9 - 1$	16.9	16.2	16.9	15.9
$S_9 - 2$	16.3	15.1	15.3	14.7
$S_9 - 3$	14.2	14.9	14.5	13.5
$S_9 - 4$	16.8	16.0	15.8	14.6
$S_9 - 5$	16.5	17.0	16.0	14.3
$S_9 - 6$	16.9	17.1	17.9	15.1
$S_9 - 7$	20.9	21.5	21.6	18.9

Tabla 7: Resultados de ACO BEP con distintos valores de q

quieren depositar más feromonas en los caminos recorridos por las hormigas, ya que son determinantes para encontrar soluciones de calidad como demostrado con los resultados para $q = 10000$. El valor de ρ es mucho más relevante en la calidad de las soluciones obtenidas, aunque también necesita de q para que hayan tanto feromonas evaporadas como nuevas en cada iteración.

4.3. Resultados

A partir de las instancias generadas aleatoriamente y de las instancias de la comuna de Santo Domingo, se compararán los resultados obtenidos por el algoritmo ACO con el método de mejor mejora con los obtenidos por el solver Gurobi. Para todos los experimentos se utilizaron los parámetros de la sección 3.3 solamente variando el número de iteraciones por experimento, y se realizaron 10 experimentos para cada instancia.

4.3.1. Comparación con solver

A partir de las 72 instancias generadas aleatoriamente, de las cuales habían 9 sets distintos, se realizaron 10 experimentos para cada una de ellas. Para comparar el rendimiento del algoritmo se utilizó el solver Gurobi para obtener una solución con métodos exactos. Si el solver no entrega una solución óptima en 15 minutos, entonces se escoge la mejor solución encontrada hasta el momento. Este es un tiempo razonable para esperar un plan de evacuación ante tsunamis, ya que el tiempo mínimo para que un tsunami llegue a la costa es de 15 minutos aproximadamente. Esto es porque al tratarse de un problema *NP-Complete*, muchas instancias son sencillas de resolver, pero al introducir nuevas variables el tiempo de resolución crece de forma exponencial. Aquellas soluciones que no fueron resueltas por el solver en 15 minutos se destacarán con un asterico (*). Además, el tiempo máximo de ejecución del algoritmo ACO fue de 70 segundos, por lo que el número de iteraciones debe ser ajustado para obtener soluciones en menos tiempo.

A partir de los 10 experimentos se destacan los valores mínimos, máximos y su promedio, el cual este último se utilizó para comparar los resultados del *solver*. El *gap* toma en cuenta la diferencia entre el promedio de resultados obtenidos y el entregado por el *solver*, como se puede ver en la fórmula 19.

$$\text{Gap} = \frac{\text{Promedio ACO} - \text{Solución Solver}}{\text{Promedio ACO}} \times 100 \quad (19)$$

En la tabla 8 se presentan los resultados obtenidos por el algoritmo ACO y el *solver* para las instancias de los sets S_1 a S_4 .

Para los primeros 4 sets de instancias, las cuales corresponden a instancias pequeñas, fueron resueltas de forma óptima por el *solver* y en muy poco tiempo. La diferencia entre los resultados obtenidos por el algoritmo ACO y el *solver* es de un par de dígitos en la mayoría de los casos, aunque hay muchos otros donde se obtiene el óptimo de forma consistente. Para el set S_1 y S_2 se obtienen soluciones óptimas en la mayoría de las instancias, y en aquellas en que no se logra la solución óptima, el *gap* es menor al 15 %. En el set S_4 se tienen soluciones cerca del óptimo, aunque hay algunas instancias que presentan un *gap* mayor al 10 %. También aquellas instancias que logran el óptimo no lo hicieron en todos los experimentos, por lo que en tales instancias las soluciones no son consistentes.

En la tabla 9 se presentan los resultados obtenidos por el algoritmo ACO y el *solver* para las instancias de los sets S_5 a S_8 .

En estas instancias se tiene una baja del rendimiento, teniendo instancias donde el algoritmo propuesto no llega al resultado óptimo. Se logran obtener instancias que retornan siempre el óptimo, pero en muchas instancias la diferencia de resultados es mayor al 20 %. El *solver* para las últimas instancias de los sets S_6 y S_8 no logra obtener una solución óptima en el tiempo límite, por lo que se escogen las mejores soluciones encontradas hasta el momento. Esto es porque a pesar de tener los mismos parámetros en las demás instancias, varían las personas a evacuar, haciendo la evacuación más larga y compleja. En estas instancias se demuestra la importancia de ocupar un método heurístico a diferencia de un método matemático, ya que el tiempo de resolución puede ser mucho mayor con este último. Al ser instancias más complejas se necesita más tiempo de ejecución para resolverlas, por lo que se necesitan aproximadamente 2000 iteraciones para obtener estos resultados.

En la tabla 10 se presentan los resultados obtenidos por el algoritmo ACO y el *solver* para las instancias del set S_9 , así como la diferencia global obtenida por ámbos métodos.

Para el set 9, el cual tiene las instancias más grandes y difíciles de resolver, el *solver* no logró el resultado óptimo en el tiempo establecido en varias de ellas. En ninguna de las instancias con el algoritmo propuesto se logra tener un resultado mejor o igual al del *solver*, aunque tales resultados se demoraban aproximadamente 60 segundos en obtenerse. Se realizaron 4000 iteraciones por experimento, lo que tiene mucha influencia en el tiempo de ejecución del algoritmo. La diferencia entre el promedio y el resultado del *solver* es de un par de dígitos en

Instancia	ACO			Solver	Gap
	Min	Max	Promedio		
$S_1 - 0$	12	12	12.0	12	0.0 %
$S_1 - 1$	11	11	11.0	11	0.0 %
$S_1 - 2$	21	23	22.2	20	11.0 %
$S_1 - 3$	14	14	14.0	14	0.0 %
$S_1 - 4$	13	13	13.0	13	0.0 %
$S_1 - 5$	12	12	12.0	12	0.0 %
$S_1 - 6$	22	24	23.4	22	6.4 %
$S_1 - 7$	15	15	15.0	15	0.0 %
$S_2 - 0$	12	12	12.0	12	0.0 %
$S_2 - 1$	11	11	11.0	11	0.0 %
$S_2 - 2$	11	11	11.0	11	0.0 %
$S_2 - 3$	10	10	10.0	10	0.0 %
$S_2 - 4$	12	14	13.6	12	13.3 %
$S_2 - 5$	15	15	15.0	15	0.0 %
$S_2 - 6$	12	12	12.0	11	9.1 %
$S_2 - 7$	13	13	13.0	13	0.0 %
$S_3 - 0$	9	11	10.6	9	17.8 %
$S_3 - 1$	10	12	11.2	10	12.0 %
$S_3 - 2$	10	11	10.6	10	6.0 %
$S_3 - 3$	17	18	17.6	15	17.3 %
$S_3 - 4$	13	15	14.0	12	16.7 %
$S_3 - 5$	7	9	8.2	7	17.1 %
$S_3 - 6$	12	13	12.7	11	15.5 %
$S_3 - 7$	15	16	15.3	14	9.3 %
$S_4 - 0$	19	20	19.1	18	6.1 %
$S_4 - 1$	18	20	18.9	17	11.2 %
$S_4 - 2$	20	22	21.5	19	13.2 %
$S_4 - 3$	16	17	16.3	15	8.7 %
$S_4 - 4$	15	16	15.1	14	7.9 %
$S_4 - 5$	18	20	18.9	18	5.0 %
$S_4 - 6$	13	15	14.1	12	17.5 %
$S_4 - 7$	34	37	35.8	32	11.9 %

Tabla 8: Resultados de ACO BEP y Solver desde S_1 a S_4

Instancia	ACO			Solver	Gap
	Min	Max	Promedio		
$S_5 - 0$	12	14	13.5	11	22.7 %
$S_5 - 1$	12	13	12.2	11	10.9 %
$S_5 - 2$	13	15	14.0	11	27.3 %
$S_5 - 3$	14	15	14.5	12	20.8 %
$S_5 - 4$	14	14	14.0	11	27.3 %
$S_5 - 5$	14	16	14.9	12	24.2 %
$S_5 - 6$	12	13	12.8	11	16.4 %
$S_5 - 7$	28	28	28.0	23	21.7 %
$S_6 - 0$	10	11	10.5	10	5.0 %
$S_6 - 1$	13	15	14.1	12	17.5 %
$S_6 - 2$	12	13	12.8	11	16.4 %
$S_6 - 3$	12	13	12.2	10	22.0 %
$S_6 - 4$	15	15	15.0	15	0.0 %
$S_6 - 5$	12	12	12.0	12	0.0 %
$S_6 - 6$	12	12	12.0	12	0.0 %
$S_6 - 7$	19	21	20.1	16*	25.6 %
$S_7 - 0$	7	8	7.2	7	2.9 %
$S_7 - 1$	10	11	10.4	9	15.6 %
$S_7 - 2$	12	14	13.0	12	8.3 %
$S_7 - 3$	9	10	9.3	8	16.3 %
$S_7 - 4$	8	8	8.0	8	0.0 %
$S_7 - 5$	9	11	9.4	8	17.5 %
$S_7 - 6$	11	13	12.6	11	14.5 %
$S_7 - 7$	10	12	10.9	9	21.1 %
$S_8 - 0$	10	11	10.8	8	35.0 %
$S_8 - 1$	7	8	7.9	6	31.7 %
$S_8 - 2$	8	9	8.9	7	27.1 %
$S_8 - 3$	12	12	12.0	12	0.0 %
$S_8 - 4$	8	9	8.8	6	46.7 %
$S_8 - 5$	8	8	8.0	6	33.3 %
$S_8 - 6$	7	8	7.9	6	31.7 %
$S_8 - 7$	16	17	16.5	12*	37.5 %

Tabla 9: Resultados de ACO BEP y Solver desde S_5 a S_8

Instancia	ACO			Solver	Gap
	Min	Max	Promedio		
$S_9 - 0$	10	11	10.8	8*	35.0 %
$S_9 - 1$	11	12	11.8	8*	47.5 %
$S_9 - 2$	10	11	10.2	9*	13.3 %
$S_9 - 3$	9	10	9.9	7	41.4 %
$S_9 - 4$	10	11	10.9	8*	36.3 %
$S_9 - 5$	10	11	10.9	9	21.1 %
$S_9 - 6$	11	12	11.8	8*	47.5 %
$S_9 - 7$	14	16	15.3	12*	27.5 %
Promedio Global					15.3 %

Tabla 10: Resultados de ACO BEP y Solver de S_9

la mayoría de los casos, lo que se refleja en que algunos resultados presenten un *gap* mayor al 40 %.

Finalmente se tiene una diferencia global de 15.3 % entre los resultados obtenidos por el algoritmo ACO y el *solver* para todas las instancias.

En general, el algoritmo ACO obtiene soluciones cercanas al óptimo, aunque en instancias más grandes la diferencia entre los resultados obtenidos por el algoritmo y el *solver* es mayor. En la mayoría de los casos, el *solver* obtiene soluciones óptimas, aunque en instancias más grandes no logra obtener una solución óptima en el tiempo establecido. En estos casos, el algoritmo propuesto es más efectivo que el *solver* al obtener buenas soluciones en tiempos razonables. El *solver*, además de ser más lento en estas instancias, ocupa muchos recursos computacionales en su ejecución, lo que impacta negativamente en el rendimiento del computador y ralentizando otras de sus tareas.

En muchas instancias se logra obtener el resultado óptimo en el mismo tiempo de ejecución que el *solver*, por lo que tienen rendimientos parecidos en aquellas instancias. Sin embargo, aunque en instancias se logre obtener el resultado óptimo, no lo logra de forma consistente, teniendo que realizar varios experimentos para obtener la solución óptima, por lo que modificar los parámetros del algoritmo podría ser una buena opción para obtener mejores resultados. En resumen, el algoritmo ACO es muy importante para resolver instancias muy grandes, esto es al obtener buenas soluciones en mucho menos tiempo, siendo estas muy cercanas al óptimo en instancias pequeñas y medianas.

4.3.2. Caso de estudio

Para el caso de la comuna de Santo Domingo, se realizó un plan de evacuación ante tsunamis con el algoritmo ACO, teorizando posibles refugios y rutas de evacuación. Variando el número de buses para evacuar a las personas, y en pos de tener las mejores rutas de evacuación, se

realizaron experimentos con 5, 10, 15 y 20 buses para obtener soluciones menores a 15 minutos. Si se obtienen soluciones menores a 15 minutos es posible realizar una evacuación exitosa, aunque esto es en condiciones ideales y sin considerar factores externos que puedan afectar la evacuación.

En la tabla 11 se presentan los resultados obtenidos por el algoritmo ACO para la comuna de Santo Domingo y para distintos números de buses. Se utilizaron los mismos parámetros que en los experimentos anteriores, con 400 iteraciones por experimento y 10 experimentos por instancia.

Buses	ACO			Solver	Gap
	Min	Max	Promedio		
5	22	24	22.8	22	4.0 %
10	14	16	14.7	14	5.0 %
15	12	12	12.0	10	20.0 %
20	10	11	10.1	10	1.0 %
Promedio Global					7.0 %

Tabla 11: Resultados de ACO BEP y Solver para las instancias de Santo Domingo en minutos

Como se puede apreciar en la tabla 11, se obtuvieron soluciones óptimas en 3 de los 4 casos, aunque tales soluciones no fueron alcanzadas de forma consistente. Al no ser instancias tan grandes, tanto el algoritmo ACO como el *solver* logran obtener estas soluciones en menos de 1 minuto. Además, el *solver* obtiene estas soluciones en el tiempo establecido, tardando alrededor de 15 segundos para los experimentos con 10 buses y menos de 5 segundos para el resto de los experimentos.

El caso con peor rendimiento es el de 5 buses, donde no se cumple una evacuación exitosa en menos de 15 minutos. Las soluciones mejoran a medida que se aumenta el número de buses, ya que con 10 buses se obtienen soluciones óptimas de 14 minutos tanto para el *solver* como para el algoritmo propuesto. Esto, en condiciones ideales, es una evacuación exitosa, sin embargo, en una situación real donde la evacuación no sea expedita, se necesita más tiempo para evitar estas problemáticas. Aunque en los puntos de encuentro la mayoría de las personas se encuentren a salvo, hay que considerar que hay muchas personas que necesitan refugio lo antes posible por problemas de salud o por estar en zonas de riesgo, por lo que se necesita evacuar a todas las personas en el menor tiempo posible.

Para el caso con 15 buses se obtienen mejores resultados con una solución óptima de 10 minutos de evacuación, aunque el algoritmo propuesto alcanza resultados de 12 minutos en promedio. Esto ya es mucho más acorde a una evacuación exitosa, ya que se pueden contemplar márgenes de error con un tiempo de evacuación mucho menor a los 15 minutos. Aún así es deseable tener soluciones menores a los 10 minutos, dado que en situaciones de emergencia se da la alerta unos minutos después de generado el tsunami, así como el tiempo en que los choferes empiecen sus recorridos con los buses y las personas se suban a los buses es indeterminable.

El último caso, con 20 buses, es el que mejor rendimiento tiene con el algoritmo propuesto, obteniendo 10.1 minutos de evacuación en promedio versus los 10 minutos entregados por el *solver*. Como no se obtuvieron mejores soluciones aumentando la cantidad de buses, es seguro concluir que con 20 buses se obtienen soluciones satisfactorias para una evacuación rápida y exitosa. A pesar de esto, al ser una comuna tan pequeña y con solamente 2 estaciones, disponer de tantos buses de forma inmediata es muy complejo; teóricamente es posible evacuar a todas las personas en el tiempo establecido, pero en la práctica es muy difícil de lograr. El caso de 10 buses podría ser mejorado con mejores puntos de refugio, y así obtener mejores soluciones con menos recursos. Por otro lado, para el caso de 5 buses parece ser que es muy difícil evacuar a 500 personas en menos de 15 minutos, por lo que se debe disponer de más buses y mejores puntos de refugio para evacuar a todas las personas con 10 buses.

Ya con las rutas definidas, se puede ver en la tabla 12 la mejor ruta para 20 buses obtenida por el algoritmo propuesto, la cual tiene un tiempo de evacuación de 10 minutos. En esta tabla se pueden ver los buses y los viajes que realizan, así como el tiempo de evacuación individual de cada bus. Los viajes son representados de la misma forma que se hizo en la sección 3.1.1.

Buses	Viajes			Tiempo de evacuación
	1	2	3	
Bus 1	(E_1, P_1)	(P_1, R_5)		7
Bus 2	(E_1, P_5)	(P_5, R_4)		10
Bus 3	(E_1, P_7)	(P_7, R_3)	(P_6, R_2)	9
Bus 4	(E_1, P_7)	(P_7, R_3)		3
Bus 5	(E_1, P_6)	(P_6, R_2)	(P_4, R_1)	10
Bus 6	(E_1, P_1)	(P_1, R_5)		7
Bus 7	(E_1, P_6)	(P_6, R_2)		6
Bus 8	(E_1, P_1)	(P_1, R_5)		7
Bus 9	(E_1, P_7)	(P_7, R_3)		3
Bus 10	(E_1, P_1)	(P_1, R_5)		7
Bus 11	(E_1, P_5)	(P_5, R_4)		10
Bus 12	(E_1, P_2)	(P_2, R_2)		8
Bus 13	(E_2, P_3)	(P_3, R_1)		8
Bus 14	(E_2, P_2)	(P_2, R_2)	(P_4, R_1)	8
Bus 15	(E_2, P_2)	(P_2, R_2)	(P_4, R_2)	10
Bus 16	(E_2, P_1)	(P_1, R_3)		9
Bus 17	(E_2, P_3)	(P_3, R_1)		8
Bus 18	(E_2, P_5)	(P_5, R_4)		10
Bus 19	(E_2, P_4)	(P_4, R_1)	(P_4, R_2)	8
Bus 20	(E_2, P_3)	(P_3, R_1)		8
Promedio				7.8

Tabla 12: Ruta para 20 buses

La mayoría de los buses realizan 2 viajes, mientras que solamente 5 buses realizan 3 viajes, ya que con los 20 buses no es posible evacuar a todas las personas con solamente 2 viajes. Con esta ruta, quedan 2 refugios con capacidad disponible: el tercero tiene capacidad de 40 y el cuarto de 20 personas. Los viajes con rutas más cortas fueron aquellos que demoraron 3 minutos, y se obtuvo un promedio de 7.8 minutos de evacuación para los 20 buses. En la figura 12 se puede ver la ruta del quinto bus, el cual demora 10 minutos en realizar su ruta de 3 viajes.

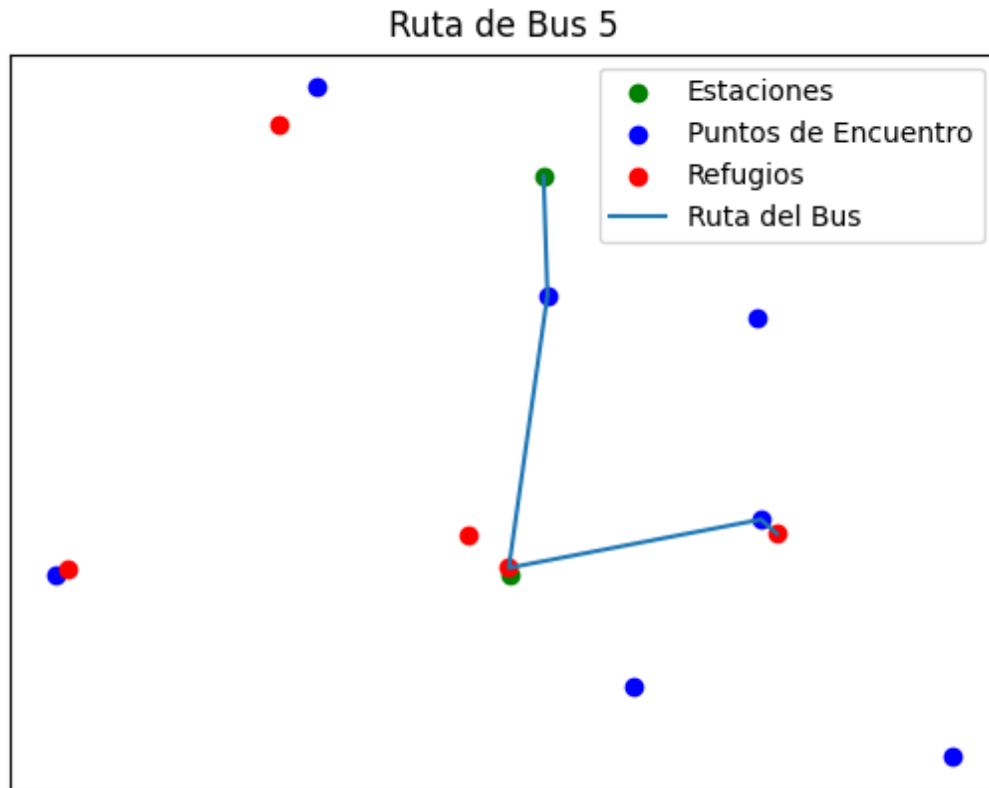


Figura 12: Ruta del quinto bus

A partir de las 400 iteraciones para el caso de 20 buses, se obtuvieron varias soluciones iniciales con el algoritmo ACO, que posteriormente fueron mejoradas con el método de mejor mejora. En la figura 13 se puede ver la evolución del camino más largo con el algoritmo ACO en cada iteración.

Como se puede apreciar en la figura 13, es cierto que la mejor solución se obtiene tras realizar varias iteraciones, pero en siguientes iteraciones se logran soluciones peores que la encontrada anteriormente. Esto es para poder explorar nuevas soluciones y no quedarse con una solución local, y al tratarse del bus con peor desempeño se pueden tener buses con mejores rutas y tiempos de evacuación. La correlación entre la cantidad de iteraciones y el tiempo de evacuación demuestra esto, ya que es muy cercana a 0, lo que indica que con este número de iteraciones no hay una relación directa entre ambos valores. A pesar de esto, es impor-

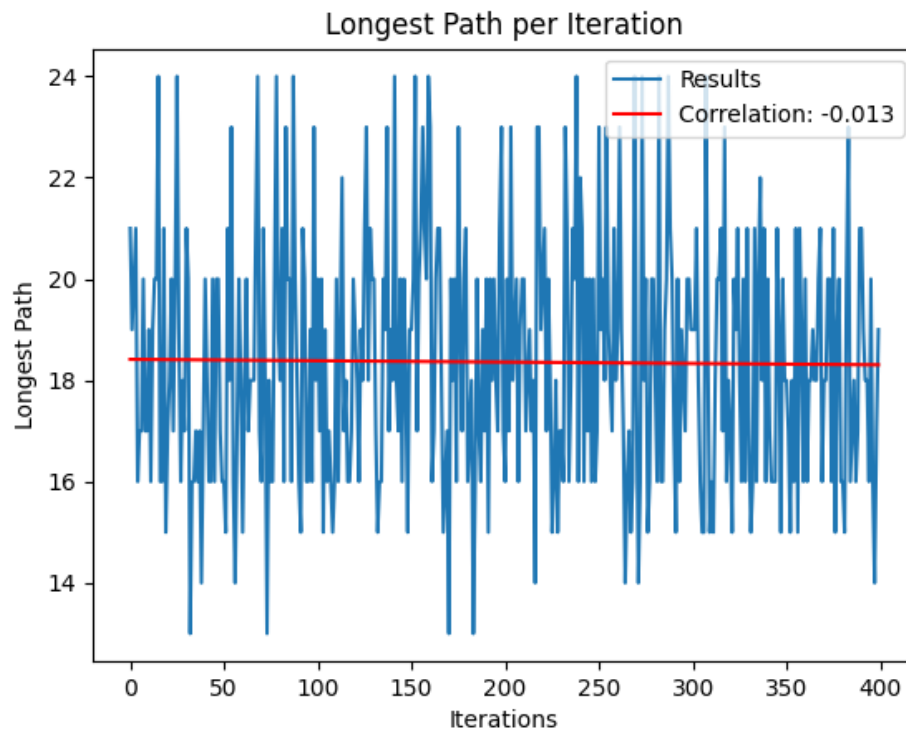


Figura 13: Evolución del camino más largo en cada iteración

tante realizar muchos experimentos de calidad para que el método reparador obtenga las soluciones óptimas, como quedó demostrado en experimentos anteriores donde se obtuvieron peores soluciones con una mala configuración de parámetros. En este caso, la mejor solución del algoritmo ACO es de 13 minutos, la cual puede ser mejorada con el método de mejor mejora a 10 minutos.

CAPÍTULO 5

CONCLUSIONES

En este documento se estudió el *Bus Evacuation Problem* y se propuso un algoritmo basado en *Ant Colony Optimization* para llegar a soluciones iniciales que luego con movimientos *shift* se modificarían para encontrar la mejor solución. El método basado en ACO ocupa una matriz de feromonas para guiar a otros agentes, las hormigas, a encontrar rutas tentativas para la evacuación. Se tomaron como principales inspiraciones a los estudios del algoritmo ACO realizados en [Ky Phuc y Phuong Thao, 2021] y los aportes sobre la resolución del BEP en [Loyola, 2016].

Se realizaron experimentos con instancias generadas aleatoriamente, así como con instancias reales de la comuna de Santo Domingo, Chile. Ambos experimentos mostraron que se obtienen soluciones cercanas al óptimo en tiempos razonables, y que la principal ventaja del algoritmo propuesto es la rapidez en encontrar soluciones en poco tiempo. Para esto se compararon los resultados obtenidos con los de un *solver* comercial de programación entera mixta, y se observó que el algoritmo propuesto es capaz de encontrar soluciones en tiempos mucho menores, aunque con una calidad ligeramente inferior. En el caso de las instancias generadas aleatoriamente, hubieron casos complejos que ambos métodos tenían dificultades para resolver, pero el algoritmo propuesto entregó soluciones en mucho menos tiempo que el *solver*. Mientras que en el caso de las instancias reales se obtuvo un buen desempeño del algoritmo propuesto, llegando a soluciones óptimas en la mayoría de los casos.

Se teorizaron posibles puntos de refugio para las personas evacuadas, utilizando espacios capaces de albergar grandes cantidades de personas, como escuelas e iglesias, y la ubicación de estos refugios puede ser un factor determinante en la eficiencia de la evacuación. Es posible trabajar con el gobierno local de la comuna para determinar puntos de refugio más cercanos a los puntos de encuentro, y así tener una planificación más eficiente y segura. El número de personas a evacuar también es un factor determinante en la eficiencia de la evacuación, lo cual también fue teorizado a partir del territorio de la comuna de Santo Domingo y su densidad poblacional. Esto es algo que también puede ser profundizado al trabajar con el gobierno local de la comuna. Se utilizaron distancias entre nodos otorgados por la plataforma Google My Maps, utilizando trayectos viables por los buses, yendo por caminos que no se encuentren en las zonas de riesgo.

Además, se analizó el desempeño con distinta cantidad de buses disponibles para la evacuación, y se observó que la evacuación más rápida se logra con 15 buses, pero que con 10 buses se obtienen resultados menores a los 15 minutos de evacuación. Sin embargo, es importante tener en cuenta que la cantidad de buses disponibles puede variar dependiendo de la temporada, ya que en verano es donde hay más personas habitando la comuna, y hay más buses de acercamiento que pueden ser utilizados para la evacuación. Estos factores son muy importantes al momento de planificar una evacuación, ya que es una forma de conocer

las limitancias en distintos contextos para que luego sean mejorados.

El método propuesto es una forma efectiva de encontrar soluciones en poco tiempo, y puede ser utilizado para evacuaciones de emergencia en la comuna de Santo Domingo, Chile. Sin embargo, hay instancias donde se tiene una diferencia muy grande con respecto al óptimo, lo que puede ser mejorado con un mejor método basado en ACO o bien un mejor método reparador. El algoritmo ACO propuesto es una buena forma de encontrar soluciones iniciales, y que terminó siendo determinante para que el método reparador encontrara mejores soluciones. Se observó que el algoritmo de reparación basado en *Hill Climbing* con mejor mejora puede ser modificado, ya que en ocasiones no logra encontrar ninguna mejora y se queda estancado en un óptimo local. Distintos métodos pueden ser implementados, tales como *Simulated Annealing* o *Tabu Search* para encontrar mejores soluciones y explorar exhaustivamente distintas soluciones.

Los parámetros del algoritmo ACO pueden ser modificados para obtener distintos resultados, los cuales son establecidos en 3.3; la cantidad de iteraciones terminó siendo determinante para encontrar soluciones en poco tiempo, habiendo una relación muy lineal entre el número de iteraciones y el tiempo de ejecución. Por otro lado, los valores de α y β influyeron bastante en la calidad de las soluciones obtenidas, siendo cercanos a 2 y 4 respectivamente los valores que mejor resultados entregan. Los valores de ρ y q , sin embargo, no influyeron en la calidad de las soluciones como se esperaba, ya que su variación no tuvo un impacto significativo en los resultados obtenidos. Esto fue contraproducente en un principio, porque se esperaba que en cada iteración la evaporación de las feromonas así como el depósito de estas impactaran de forma significativa en los resultados, permitiendo a las hormigas tomar mejores decisiones en cada iteración. Se teorizó depositar feromonas solamente en caminos que la mejor hormiga recorrió, pero esto dejaba a la matriz de feromonas muy vacía y no permitía a las hormigas explorar distintas alternativas. Por lo tanto, se decidió depositar feromonas en todos los caminos recorridos por las hormigas, lo que significó una mejora en la calidad de las soluciones obtenidas.

La aleatoriedad para la elección de los caminos a recorrer por las hormigas es un factor determinante en la exploración de distintas soluciones, y aunque en ocasiones se encuentren soluciones subóptimas, es necesario para explorar el gran espacio de búsqueda del problema. Al no utilizar métodos exactos se corre el riesgo de encontrar soluciones subóptimas, pero se obtienen soluciones de calidad en tiempos razonables, y utilizando menos recursos computacionales.

El método basado en ACO es una buena forma de encontrar soluciones iniciales para el BEP, y puede entregar buenas soluciones para las instancias aleatorias y reales de la comuna de Santo Domingo, Chile. Es en instancias medianas donde el método no logra encontrar soluciones óptimas, mientras que con métodos exactos se logra encontrar soluciones óptimas en tiempos similares. Debido a esto, junto con el comportamiento inesperado de la evolución de soluciones visto en la figura 13 y la influencia de los parámetros ρ y q en la calidad de las soluciones obtenidas, es probable que el método tenga limitaciones. La complejidad del problema recae en la ida y vueltas de los buses, la capacidad de los refugios y la cantidad de

personas a evacuar, por lo que hay caminos que no pueden ser recorridos por las hormigas, y las feromonas depositadas en aquellos caminos no son relevantes para la construcción de la solución.

El algoritmo propuesto es una forma efectiva de encontrar soluciones en poco tiempo, y a pesar de que las feromonas tienen un comportamiento inesperado, es un problema complejo de optimizar y los parámetros del algoritmo probaron ser determinantes en la calidad de las soluciones obtenidas. El objetivo de las hormigas, que es encontrar caminos cortos para la evacuación, resulta muy importante para la construcción de la solución, combinando la exploración de aquellos caminos que poseen más feromonas junto con la distancia entre los nodos.

Como trabajo futuro, es posible implementar un modelo que determine los mejores puntos de refugio para la evacuación, y que estos sean determinantes en la eficiencia de la evacuación. Muchos refugios pueden encontrarse muy lejos de los puntos de encuentro por lo que pueden ser omitidos, y optar por otros más cercanos a los puntos de encuentro. Además, es posible trabajar con otros gobiernos de las comunas chilenas para implementar este método en otro tipo de evacuaciones, como erupciones volcánicas. Muchas de las ideas propuestas en este trabajo pueden ser muy relevantes para la evacuación de personas en situaciones críticas, siguiendo siempre las recomendaciones de las instituciones del país.

REFERENCIAS BIBLIOGRÁFICAS

- [lot, 2011] (2011). Lota ejecutó su propio plan de emergencia con prontitud y eficacia. Último acceso: 27 Sep., 2024, en <https://resumen.cl/articulos/lota-ejecuto-su-propio-plan-de-emergencia-con-prontitud-y-eficacia>.
- [Araneda et al., 2006] Araneda, M., Avendaño, M., y Kausel, E. (2006). Observaciones gravimétricas y nivelaciones en proceso del terremoto del 3 de marzo de 1985, Chile. *Revista Geofísica*, (62):49–61.
- [Arias-Rojas et al., 2012] Arias-Rojas, J. S., Jiménez, J. F., y Montoya-Torres, J. R. (2012). Solving of school bus routing problem by ant colony optimization. *Revista EIA*, (17):193–208.
- [Berg, 2013] Berg, R. (2013). Tropical cyclone report hurricane isaac (al092012) 21 august–1 september 2012. *National Hurricane Center*, pp. 1–78.
- [Bish, 2011] Bish, D. R. (2011). Planning for a bus-based evacuation. *OR spectrum*, 33(3):629–654.
- [Bullnheimer et al., 1999] Bullnheimer, B., Hartl, R. F., y Strauss, C. (1999). Applying the ant system to the vehicle routing problem. *Meta-heuristics: Advances and trends in local search paradigms for optimization*, pp. 285–296.
- [Chen et al., 2022] Chen, B., Li, Z., y Yang, Z. (2022). Nuclear accident emergency response system: radiation field estimation and evacuation. *Sustainability*, 14(9):5663.
- [Dikas y Minis, 2016] Dikas, G. y Minis, I. (2016). Solving the bus evacuation problem and its variants. *Computers & Operations Research*, 70:75–86.
- [Forcael et al., 2014] Forcael, E., González, V., Orozco, F., Vargas, S., Pantoja, A., y Moscoso, P. (2014). Ant colony optimization model for tsunamis evacuation routes. *Computer-Aided Civil and Infrastructure Engineering*, 29(10):723–737.
- [Goerigk y Grün, 2012] Goerigk, M. y Grün, B. (2012). The robust bus evacuation problem.
- [Goerigk et al., 2013] Goerigk, M., Grün, B., y Heßler, P. (2013). Branch and bound algorithms for the bus evacuation problem. *Computers & Operations Research*, 40(12):3010–3020.
- [Goerigk et al., 2014] Goerigk, M., Grün, B., y Heßler, P. (2014). Combining bus evacuation with location decisions: A branch-and-price approach. *Transportation Research Procedia*, 2:783–791.
- [Ky Phuc y Phuong Thao, 2021] Ky Phuc, P. N. y Phuong Thao, N. L. (2021). Ant colony optimization for multiple pickup and multiple delivery vehicle routing problem with time window and heterogeneous fleets. *Logistics*, 5(2):28.

- [Li et al., 2019] Li, Z., Yu, H., Chen, X., Zhang, G., y Ma, D. (2019). Tsunami-induced traffic evacuation strategy optimization. *Transportation research part D: transport and environment*, 77:535–559.
- [Loyola, 2016] Loyola, J. (2016). *Una técnica de resolución para el bus evacuation problem. Aplicación a la región de Valparaíso*.
- [Ma et al., 2017] Ma, Y., Han, J., Kang, K., y Yan, F. (2017). An improved aco for the multi-depot vehicle routing problem with time windows. En *Proceedings of the Tenth International Conference on Management Science and Engineering Management*, pp. 1181–1189. Springer.
- [Molina et al., 2014] Molina, J. C., Eguia, I., Racero, J., y Guerrero, F. (2014). Multi-objective vehicle routing problem with cost and emission functions. *Procedia-Social and Behavioral Sciences*, 160:254–263.
- [Quezada et al., 2012] Quezada, J., Jaque, E., Fernández, A., y Vásquez, D. (2012). Cambios en el relieve generados como consecuencia del terremoto mw= 8, 8 del 27 de febrero de 2010 en el centro-sur de chile. *Revista de Geografía Norte Grande*, (53):35–55.
- [Sougarret, 2020] Sougarret, A. (2020). Recordando los hitos en el rescate de san José. Último acceso: 22 Nov., 2023, en <https://www.mch.cl/2020/10/15/recordando-los-hitos-en-el-rescate-de-san-jose/>.
- [Thapa y Khadka, 2016] Thapa, B. B. y Khadka, S. R. (2016). A brief note on bus based evacuation planning. *Journal of Institute of Science and Technology*, 21(1):28–34.
- [Townsend, 2024] Townsend, N. (2024). Tres hileras de autos que no pudieron escapar del fuego: la historia detrás de la imagen más impactante del incendio en viña del mar. Último acceso: 17 Oct., 2024, en <https://www.theclinic.cl/2024/02/10/historia-detras-foto-mas-impactante-incendio-vina-del-mar-achupallas/>.
- [Wenyi, 2023] Wenyi, L. (2023). Research on vehicle routing problem and application scenarios. En *Handbook of Mobility Data Mining*, pp. 63–88. Elsevier.
- [Zhu, 2017] Zhu, X. (2017). Dynamic bus routing problem for evacuation. Tesis de m^áster, State University of New York at Buffalo.