Estado del Arte: Bus Evacuation Problem (BEP)

Lorna Mella

July 11, 2025

Abstract

Durante las últimas décadas, la necesidad de evacuar grandes grupos de personas ante catástrofes naturales ha impulsado el desarrollo del **Bus Evacuation Problem (BEP)**. Este problema logístico consiste en trasladar personas desde zonas de riesgo hacia refugios seguros utilizando una flota limitada de buses, con el objetivo de minimizar el tiempo total de evacuación (*makespan*). En este informe se modela formalmente el problema y se analiza su complejidad, así como sus principales formulaciones matemáticas. Además, se revisa el estado del arte, incluyendo enfoques exactos, heurísticos e híbridos. Finalmente, se propone e implementa una solución heurística basada en un algoritmo *Greedy* combinado con *Hill Climbing*, evaluando su rendimiento en nueve instancias representativas. Los resultados muestran que la variante con **mejor mejora** logra un equilibrio efectivo entre calidad de solución y eficiencia computacional, validando su aplicabilidad en contextos de evacuación reales donde el tiempo de respuesta es crítico.

1 Introducción

En el contexto de catástrofes naturales, como terremotos, huracanes o incendios forestales, resulta esencial evacuar a grandes grupos de personas en el menor tiempo posible. La planificación eficiente de estos procesos es un desafío logístico que ha motivado el estudio del *Bus Evacuation Problem* (BEP), una especialización del *Vehicle Routing Problem* (VRP), centrada en trasladar personas desde zonas de riesgo hacia refugios utilizando una flota limitada de autobuses [1].

Este informe tiene como propósito analizar y modelar formalmente el BEP, describiendo sus variantes fundamentales y explorando estrategias de resolución. La estructura del documento es la siguiente: primero se presenta una definición general del problema y sus principales elementos; luego, se exponen los modelos matemáticos más representativos; posteriormente se revisa el estado del arte, incluyendo enfoques exactos, heurísticos y recientes avances híbridos; finalmente, se comentan implicancias prácticas y direcciones futuras de investigación. La

motivación principal es entender cómo se ha abordado este problema en la literatura y qué herramientas pueden ser más efectivas para enfrentarlo en escenarios reales.

2 Definición del Problema

El *Bus Evacuation Problem* (BEP) es un problema de optimización que surge en el contexto de la gestión de evacuaciones ante catástrofes naturales o emergencias urbanas. Su objetivo es organizar de forma eficiente el traslado de personas desde puntos de encuentro hasta refugios seguros utilizando una flota limitada de autobuses. Este tipo de planificación es esencial cuando muchas personas dependen del transporte público y se requiere evacuar rápidamente zonas de riesgo [1].

El problema considera varios elementos clave:

- **Depósitos**: puntos desde donde parten los autobuses.
- **Puntos de encuentro**: ubicaciones donde se reúnen los evacuados.
- **Refugios**: lugares seguros con capacidad limitada.
- **Autobuses**: vehículos con capacidad finita para trasladar personas.

Las decisiones a tomar incluyen qué rutas debe seguir cada bus, cuántas personas debe transportar por viaje y en qué orden realizar los recorridos. Las principales restricciones consisten en:

- Todos los evacuados deben ser trasladados a refugios.
- No se deben exceder las capacidades de los buses ni de los refugios.
- El tiempo total de evacuación debe ser mínimo, definido como el tiempo que toma el bus que más demora en completar su tarea.

El BEP es una variante especializada del *Vehicle Routing Problem* (VRP), ampliamente estudiado en la literatura de investigación operativa [13, 9]. Sin embargo, introduce aspectos humanitarios como la urgencia y la equidad en la evacuación, que no son considerados en el VRP tradicional [10].

Variantes del BEP

A lo largo de los años, el BEP ha dado origen a diversas variantes que abordan desafíos adicionales o se adaptan a condiciones más realistas:

- **Robust Bus Evacuation Problem (RBEP)**: Esta variante considera la presencia de incertidumbre en los datos, como tiempos de viaje variables o rutas que pueden quedar bloqueadas. Modelos robustos buscan soluciones que mantengan buen rendimiento bajo diferentes escenarios posibles [6]. Es especialmente útil en situaciones de desastres naturales donde la infraestructura puede estar dañada.

- **Casualty Evacuation Problem (CEP)**: Se enfoca en la evacuación prioritaria de personas con necesidades médicas o de movilidad reducida. Introduce objetivos múltiples, como minimizar el tiempo para evacuar a los heridos más graves, al tiempo que se maximiza la cobertura total [4]. Esta variante incorpora factores éticos y criterios de priorización.
- **Network Flow Planning (NFP)**: Propuesto por Feng et al. [5], este enfoque descompone el problema en dos etapas: primero se formula un modelo de flujo para distribuir eficientemente la carga de evacuación, y luego se asignan rutas detalladas para los buses.
- **Integrated BEP (IBEP)**: La literatura menciona enfoques integrados que abordan simultáneamente decisiones relacionadas con asignación, ruteo y programación temporal, superando la clásica descomposición por etapas del BEP tradicional [1].

Estas variantes reflejan los avances metodológicos y la necesidad de adaptar el BEP a escenarios realistas. Su estudio ha permitido el desarrollo de soluciones más robustas, eficientes y sensibles al contexto humanitario. Además, algunas de estas aproximaciones emplean heurísticas, algoritmos evolutivos o enfoques híbridos, como ocurre en otros problemas complejos de satisfacción de restricciones [11].

3 Estado del arte

El *Bus Evacuation Problem* (BEP) es una especialización del clásico *Vehicle Routing Problem* (VRP) como se mencionó antes, surgida a comienzos de los años 2000, cuando la necesidad de evacuar grandes grupos de personas ante catástrofes naturales evidenció la falta de soluciones logísticas específicas. Su formalización más influyente se encuentra en Bish [1], donde se propone evacuar personas desde zonas de riesgo hacia refugios utilizando buses, minimizando el tiempo total requerido.

Este problema ha recibido atención creciente tras eventos como el huracán Katrina (2005), motivando una línea de investigación centrada en optimización bajo restricciones humanitarias, operacionales y temporales. Desde entonces, se han desarrollado múltiples enfoques para resolver el BEP, los que pueden clasificarse en tres grandes familias: métodos exactos, algoritmos de aproximación y heurísticas o metaheurísticas.

Los primeros avances se basaron en modelos de programación entera mixta (MIP). El modelo base más conocido es el propuesto por Goerigk et al. [6], que formula el BEP como un problema de *makespan* clásico[1]. Sobre esta formulación se aplican algoritmos exactos como *branch and bound*, que exploran exhaustivamente el espacio de decisiones para encontrar soluciones óptimas. No

[1] *Makespan*, en problemas de planificación, corresponde al tiempo total que tarda el último recurso (por ejemplo, bus) en completar su tarea asignada. Se utiliza como criterio de minimización en problemas de programación y logística.

obstante, estos métodos escalan mal a problemas grandes, debido a su complejidad exponencial, es decir, a la naturaleza NP-hard del problema [7]. Una mejora importante fue introducida por Dikas y Minis [4], quienes combinaron generación de columnas (exacta) con búsqueda en vecindarios variables (VNS y LNS), logrando resolver instancias más grandes sin perder calidad. Este enfoque híbrido se adapta bien a variantes prácticas del problema, como evacuaciones con prioridad médica o con bloqueos en la red.

Debido a la alta complejidad computacional del BEP, también se han propuesto algoritmos de aproximación que sacrifican optimalidad a cambio de eficiencia. Pedrosa y Schouery [10] propusieron esquemas con garantías teóricas: una cota de 10.2 en el caso general, y 4.2 cuando los refugios tienen capacidad ilimitada. Estos resultados son relevantes para la teoría de la computación, pero su aplicabilidad práctica es limitada por su baja eficiencia en instancias reales.

El uso de heurísticas e inteligencia artificial se ha convertido en la tendencia dominante, especialmente para entornos urbanos a gran escala. Se han implementado algoritmos genéticos, búsqueda local, *simulated annealing*, *tabu search*, optimización por enjambre de partículas (PSO), y recientemente matheurísticas que combinan técnicas heurísticas con modelos exactos [3, 4, 5]. Uno de los enfoques más relevantes es el propuesto por Feng et al. [5], quien desarrolló el algoritmo *Network Flow Planning* (NFP). Este método divide el problema en dos etapas: primero se optimiza un modelo de flujo de red con programación lineal, y luego se asignan rutas de forma balanceada entre los buses. Ha sido aplicado exitosamente en situaciones reales como el aluvión de Xingguo (China), mostrando buena escalabilidad. Los movimientos más eficaces aplicados en heurísticas incluyen 2-opt, intercambio de nodos, realineamiento de rutas y redistribución de carga, especialmente en el marco de VNS y LNS [2, 12].

Una forma útil de analizar los avances en el estudio del BEP es comparar el **tamaño del espacio de búsqueda** asociado a cada formulación o enfoque. Este aspecto influye directamente en la **viabilidad computacional** de los algoritmos, especialmente en contextos donde el tiempo de respuesta es crítico. En la Figura 1 se presenta una comparación cualitativa de diferentes enfoques, desde modelos exactos como BEP-I y Branch and Bound, hasta propuestas heurísticas y de flujo agregado como NFP o BEP-II. La escala logarítmica ilustra la cantidad relativa de combinaciones posibles que deben evaluarse en cada caso.

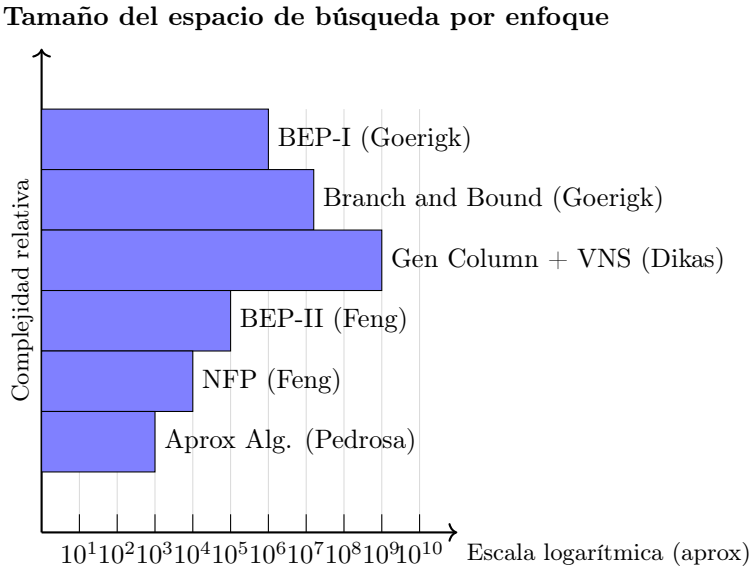


Figure 1: Comparación cualitativa del tamaño del espacio de búsqueda de distintos algoritmos para resolver el BEP.

En términos de aplicabilidad práctica, el modelo NFP (Feng) se posiciona como el más eficiente y escalable, especialmente en entornos reales de gran tamaño. Aunque no garantiza optimalidad, su rendimiento cercano al óptimo y su implementación en situaciones reales le dan ventaja frente a métodos más costosos. Esta perspectiva comparativa no solo permite evaluar la eficiencia de los modelos existentes, sino que también anticipa el rumbo que ha tomado la investigación en años recientes.

A partir de esta comparación, se observa que las representaciones más eficaces del BEP han sido aquellas basadas en modelos de red, especialmente los modelos de flujo agregado como BEP-II o NFP. Estas representaciones simplifican el espacio de búsqueda, evitando modelar el recorrido individual de cada bus. También han ganado protagonismo los enfoques híbridos, que permiten aplicar técnicas exactas a partes del problema, combinadas con metaheurísticas que guían la exploración.

Actualmente, la tendencia más fuerte en la resolución del BEP consiste en adoptar enfoques híbridos con representación de red agregada, priorizando algoritmos escalables y adaptativos. Las líneas de desarrollo más activas incluyen:

- Modelos robustos que incorporan incertidumbre en demanda, tiempos y rutas.
- Algoritmos híbridos que combinan técnicas exactas (como generación de columnas) con heurísticas.

- Integración de inteligencia artificial y planificación adaptativa en tiempo real [8].
- Incorporación de criterios de equidad y priorización humanitaria.

Finalmente, el BEP comparte similitudes estructurales con otros problemas complejos de satisfacción de restricciones, como se discute en trabajos clásicos de algoritmos híbridos para CSP [11].

4 Modelo Matemático

El *Bus Evacuation Problem* (BEP) se modela como un problema de programación entera mixta (MIP) cuyo objetivo es evacuar a todas las personas desde puntos de riesgo hacia refugios utilizando una flota de buses, minimizando el tiempo total de operación. A continuación, se presentan dos modelos representativos: un modelo clásico propuesto por Goerigk et al. [6], conocido como **BEP-I**, y una versión agregada más eficiente computacionalmente desarrollada por Feng et al. [5], denominada **BEP-II**.

Modelo BEP-I (Programación entera basada en rutas)

Este modelo se enfoca en el seguimiento detallado de cada bus y cada viaje. Está diseñado para encontrar una solución óptima minimizando el **makespan**, es decir, el tiempo que más tarda un bus en completar su secuencia de viajes.

Conjuntos:

- $\mathcal{B}$: conjunto de buses disponibles, con $|\mathcal{B}| = B$.
- $\mathcal{D}$: conjunto de depósitos donde inician los buses.
- $\mathcal{P}$: conjunto de puntos de encuentro (zonas a evacuar).
- $\mathcal{S}$: conjunto de refugios (zonas seguras).
- $\mathcal{R}$: conjunto de iteraciones de viaje posibles para cada bus.
- $\mathcal{I}$: conjunto de arcos (i, j) posibles entre nodos.

Parámetros:

- t_{ij}: tiempo de viaje entre los nodos i y j.
- Q_p: número de personas a evacuar desde el punto p.
- U_s: capacidad máxima del refugio s.
- B_d: cantidad de buses asignados al depósito d.

Variables de decisión y dominios:

- $x_{b,r,(i,j)} \in \{0,1\}$: vale 1 si el bus b realiza su r-ésimo viaje desde i hasta j, y 0 en caso contrario, para todo $b \in \mathcal{B}$, $r \in \mathcal{R}$, $(i,j) \in \mathcal{I}$.
- $T_e \in \mathbb{R}_{\geq 0}$: tiempo total de evacuación (makespan).

Función objetivo:

$$\min T_e = \max_{b \in \mathcal{B}} \sum_{r \in \mathcal{R}} \sum_{(i,j) \in \mathcal{I}} t_{ij} \cdot x_{b,r,(i,j)}$$

Se busca minimizar el *makespan*, es decir, el tiempo total que tarda el último bus en completar su servicio.

Restricciones:

- **Partida desde depósitos:** Cada bus puede salir solo desde su depósito:

$$\sum_{b \in \mathcal{B}} \sum_{p \in \mathcal{P}} x_{b,1,(d,p)} \leq B_d, \quad \forall d \in \mathcal{D}$$

- **Conservación de flujo:** Si un bus entrega pasajeros en un refugio, debe retornar a un punto de encuentro:

$$\sum_{s \in \mathcal{S}} x_{b,r,(p,s)} = \sum_{s' \in \mathcal{S}} x_{b,r+1,(s',p)}, \quad \forall b \in \mathcal{B},\ \forall p \in \mathcal{P},\ \forall r$$

- **Máximo un viaje por iteración:**

$$\sum_{(p,s) \in \mathcal{I}} x_{b,r,(p,s)} \leq 1, \quad \forall b, \forall r$$

- **Evacuar toda la demanda:**

$$\sum_{s \in \mathcal{S}} \sum_{b \in \mathcal{B}} \sum_{r \in \mathcal{R}} x_{b,r,(p,s)} \geq Q_p, \quad \forall p \in \mathcal{P}$$

- **No exceder la capacidad de refugios:**

$$\sum_{p \in \mathcal{P}} \sum_{b \in \mathcal{B}} \sum_{r \in \mathcal{R}} x_{b,r,(p,s)} \leq U_s, \quad \forall s \in \mathcal{S}$$

Espacio de búsqueda: Cada variable $x_{b,r,(i,j)}$ es binaria. El número total de variables es $B \cdot R \cdot A$ donde $A = |\mathcal{I}|$. Por tanto, la cardinalidad del espacio de búsqueda es:

$$\Omega = 2^{B \cdot R \cdot A}$$

lo que evidencia su naturaleza exponencial y la dificultad para escalar a instancias grandes.

Modelo BEP-II (modelo de flujo agregado)

Este modelo, propuesto por Feng et al. [5], representa una formulación más eficiente computacionalmente que BEP-I. En lugar de modelar el recorrido de cada bus de manera individual, agrega los flujos de buses sobre los arcos de una red, lo cual reduce considerablemente la dimensionalidad del problema. Su objetivo es minimizar el tiempo total de desplazamiento de todos los buses, lo cual resulta en tiempos de cómputo mucho menores, sacrificando parcialmente la precisión del makespan.

Conjuntos:

- $\mathcal{D}$: conjunto de depósitos desde los cuales parten los buses.
- $\mathcal{P}$: conjunto de puntos de encuentro donde se ubican las personas a evacuar.
- $\mathcal{S}$: conjunto de refugios disponibles para la evacuación.
- $\mathcal{I}$: conjunto de arcos posibles (i, j) entre nodos del conjunto $\mathcal{D} \cup \mathcal{P} \cup \mathcal{S}$.

Parámetros:

- t_{ij}: tiempo de viaje entre los nodos i y j.
- Q_p: número total de personas a evacuar desde el punto de encuentro $p \in \mathcal{P}$.
- U_s: capacidad máxima del refugio $s \in \mathcal{S}$.
- K_{ij}: máximo número de buses que puede transitar por el arco (i, j), definido por restricciones físicas o de demanda.

Variables de decisión y dominios:

- $X_{ij} \in \mathbb{Z}_+$: cantidad de buses que recorren el arco (i, j), con $0 \leq X_{ij} \leq K_{ij}$ para todo $(i, j) \in \mathcal{I}$.
- $T_{\text{tot}} \in \mathbb{R}_{\geq 0}$: tiempo total acumulado de desplazamiento.

Función objetivo:

$$\min T_{\text{tot}} = \sum_{(i,j)\in\mathcal{I}} t_{ij} \cdot X_{ij}$$

Esta función busca minimizar el tiempo total de desplazamiento acumulado de todos los buses, considerando el tiempo asociado a cada arco recorrido.

Restricciones:

- **Evacuar toda la demanda de cada punto de encuentro:**

$$\sum_{s \in \mathcal{S}} X_{ps} \geq Q_p, \quad \forall p \in \mathcal{P}$$

  Cada punto de encuentro debe evacuar a todas las personas que se encuentran allí.

- **No exceder la capacidad de los refugios:**

$$\sum_{p \in \mathcal{P}} X_{ps} \leq U_s, \quad \forall s \in \mathcal{S}$$

  Cada refugio debe recibir a lo más su capacidad máxima de personas.

- **Conservación de flujo en los puntos de encuentro:**

$$\sum_{i \in \mathcal{D} \cup \mathcal{S}} X_{ip} = \sum_{s \in \mathcal{S}} X_{ps}, \quad \forall p \in \mathcal{P}$$

  Todo flujo que llega a un punto de encuentro debe salir hacia algún refugio, manteniendo la consistencia del sistema.

Espacio de búsqueda: Cada variable X_{ij} es un entero positivo acotado por K_{ij}. Si existen $A = |\mathcal{I}|$ arcos posibles, entonces:

$$\Omega = \prod_{(i,j) \in \mathcal{I}} (K_{ij} + 1)$$

Este espacio es mucho más pequeño que en el modelo BEP-I, ya que no se consideran buses individuales ni iteraciones de viaje. La cardinalidad del dominio de cada variable depende de su límite superior K_{ij}, definido por la capacidad de buses, personas a evacuar y capacidad de los refugios. Este enfoque reduce la complejidad computacional y permite resolver instancias de gran tamaño de manera eficiente.

5 Representación

La implementación del Bus Evacuation Problem (BEP) se basa en una representación computacional sencilla pero eficiente, alineada estrechamente con su formulación matemática. La solución candidata se expresa explícitamente como un **vector de estructuras** `Bus`, en el que cada elemento almacena la información completa sobre los viajes realizados por ese bus durante la evacuación.

5.1 Estructura de datos utilizada

Los distintos componentes del escenario y las restricciones del problema se modelaron mediante las siguientes estructuras:

- **Matriz** `d_ini` **(**$Y \times P$**)**: almacena las distancias desde cada estación y hacia cada punto de encuentro p.
- **Matriz** `d_pr` **(**$P \times R$**)**: representa las distancias entre puntos de encuentro p y refugios r.
- **Vector** `q_p` **(**P**)**: contiene la cantidad de personas que deben ser evacuadas desde cada punto de encuentro.
- **Vector** `u_r` **(**R**)**: guarda la capacidad disponible en cada refugio.
- **Estructura** `Viaje`: representa un viaje individual como un par ordenado (p, r), indicando que el bus recoge personas en el punto p y las traslada al refugio r.
- **Vector de** `Bus`: **estructura principal que representa la solución**. Cada bus almacena:
  - Una lista ordenada de viajes (p, r) realizados.
  - El tiempo total acumulado de operación.
  - La estación de origen.
  - El último refugio visitado, lo que permite calcular los desplazamientos entre viajes consecutivos.

5.2 Ejemplo de representación de la solución

Cada solución se compone de una secuencia de viajes para cada bus, representada como una lista de pares ordenados. Por ejemplo:

$$\begin{array}{ll} \texttt{Bus 1:} & [(p_1, r_2),\ (p_2, r_1)] \\ \texttt{Bus 2:} & [(p_3, r_2)] \\ \texttt{Bus 3:} & [(p_2, r_2),\ (p_1, r_1)] \end{array}$$

Cada viaje (p, r) afecta directamente dos recursos:

- Disminuye el número de personas pendientes por evacuar en el punto p.
- Reduce la capacidad disponible en el refugio r.

5.3 Ventajas de la representación

Esta estructura de solución resulta particularmente adecuada por las siguientes razones:

- **Simplicidad y eficiencia**: Permite acceso rápido al historial de cada bus y al cálculo de su tiempo total.
- **Modificabilidad**: Facilita operaciones como agregar, quitar o intercambiar viajes, esenciales para aplicar heurísticas como Hill Climbing.
- **Correspondencia directa con el modelo**: Cada viaje puede asociarse con una variable binaria $x_{b,r,(i,j)}$ del modelo BEP-I, lo que permite mantener la coherencia entre el algoritmo y la formulación teórica.

Junto con las matrices de distancias y los vectores que controlan la demanda y la capacidad, esta representación permite modelar tanto el estado dinámico de la evacuación como las restricciones logísticas del problema.

6 Descripción del algoritmo

El algoritmo implementado para resolver el Bus Evacuation Problem (BEP) combina una heurística **Greedy** para construir una solución inicial factible y un proceso de **Hill Climbing con mejor mejora** para refinar esa solución. La implementación es completamente iterativa (no recursiva) y fue desarrollada en lenguaje C++.

6.1 Estructura general del algoritmo

A continuación se presenta un pseudocódigo que resume la estructura general del sistema:

```
1. Leer y parsear la instancia
2. Inicializar estructuras de datos (buses, personas, capacidades)
3. Solución inicial ← Algoritmo Greedy
4. Mejorar solución ← Hill Climbing (mejor mejora)
5. Imprimir solución y métricas
```

Cada uno de estos pasos se describe a continuación:

6.2 Construcción inicial mediante heurística Greedy

Esta etapa construye una solución factible siguiendo una lógica de prioridad. El pseudocódigo de la estrategia es:

```
Mientras existan personas por evacuar:
    1. Seleccionar el punto de encuentro con más personas (p_max)
    2. Seleccionar el refugio más cercano a p_max con capacidad
```

```
3. Calcular qué bus puede realizar el viaje con menor tiempo adicional
4. Asignar viaje (p_max, refugio) a ese bus
5. Actualizar capacidades y tiempo acumulado
```

Comentarios de implementación:

- Se utilizan operadores de búsqueda lineal sobre los vectores para identificar el punto con más personas y el mejor refugio, lo que permite máxima transparencia y facilidad de depuración.
- La métrica para seleccionar el bus es el *menor tiempo adicional*, considerando si el bus está en su estación o si ya hizo viajes.
- Las distancias entre viajes sucesivos se estiman usando la matriz `d_pr` de forma simétrica (p $\rightarrow$ r y r $\rightarrow$ p) para simplificar la implementación.

Durante la construcción Greedy, cada decisión de asignación del **movimiento** garantiza el cumplimiento de todas las restricciones del problema de forma explícita:

- **Cobertura total:** el ciclo continúa hasta que la suma de personas por evacuar (`q_p`) en todos los puntos de encuentro es cero. Esto asegura que todas las personas sean evacuadas.
- **Capacidad del bus y del refugio:** antes de realizar cada viaje, se calcula la cantidad de personas que el bus puede evacuar usando la siguiente expresión:

  $$q = \min(C,\ q_p,\ u_r)$$

  donde C es la capacidad del bus, q_p las personas disponibles en el punto p y u_r la capacidad restante en el refugio r. Esta expresión garantiza simultáneamente el cumplimiento de:

  - la **restricción de capacidad del bus**, evitando transportar más personas de las que permite su capacidad máxima;
  - y la **restricción de capacidad del refugio**, ya que nunca se envían más personas de las que el refugio puede recibir.

De esta forma, el algoritmo Greedy no solo construye una solución inicial eficiente, sino que garantiza que cada paso cumpla todas las restricciones del problema sin necesidad de validaciones posteriores.

6.3 Mejoramiento local: Hill Climbing con mejor mejora

Una vez construida la solución inicial, se ejecuta una búsqueda local para reducir el makespan. Se identifican los buses más lento y más rápido y se prueba mover viajes desde el primero al segundo:

```
Repetir mientras se encuentre mejora:
    1. Identificar bus más lento (mayor tiempo) y más rápido
    2. Para cada viaje del bus lento:
        a. Mover el viaje al bus rápido
        b. Recalcular tiempo de ambos buses
        c. Evaluar makespan resultante
        d. Si mejora, guardar como mejor vecino
    3. Aplicar el mejor movimiento encontrado
```

Comentarios de implementación:

- Se implementa de forma iterativa, sin recursión.
- Se calcula el tiempo acumulado de cada bus desde cero tras cada movimiento, para evitar errores de acumulación.
- Solo se mueven viajes completos; no se parte ni divide un viaje.
- Se guarda la mejor mejora posible en cada iteración (estrategia de mejor mejora) en lugar de la primera encontrada.
- No se usa un umbral de terminación por iteraciones: se detiene cuando no hay más mejoras posibles.

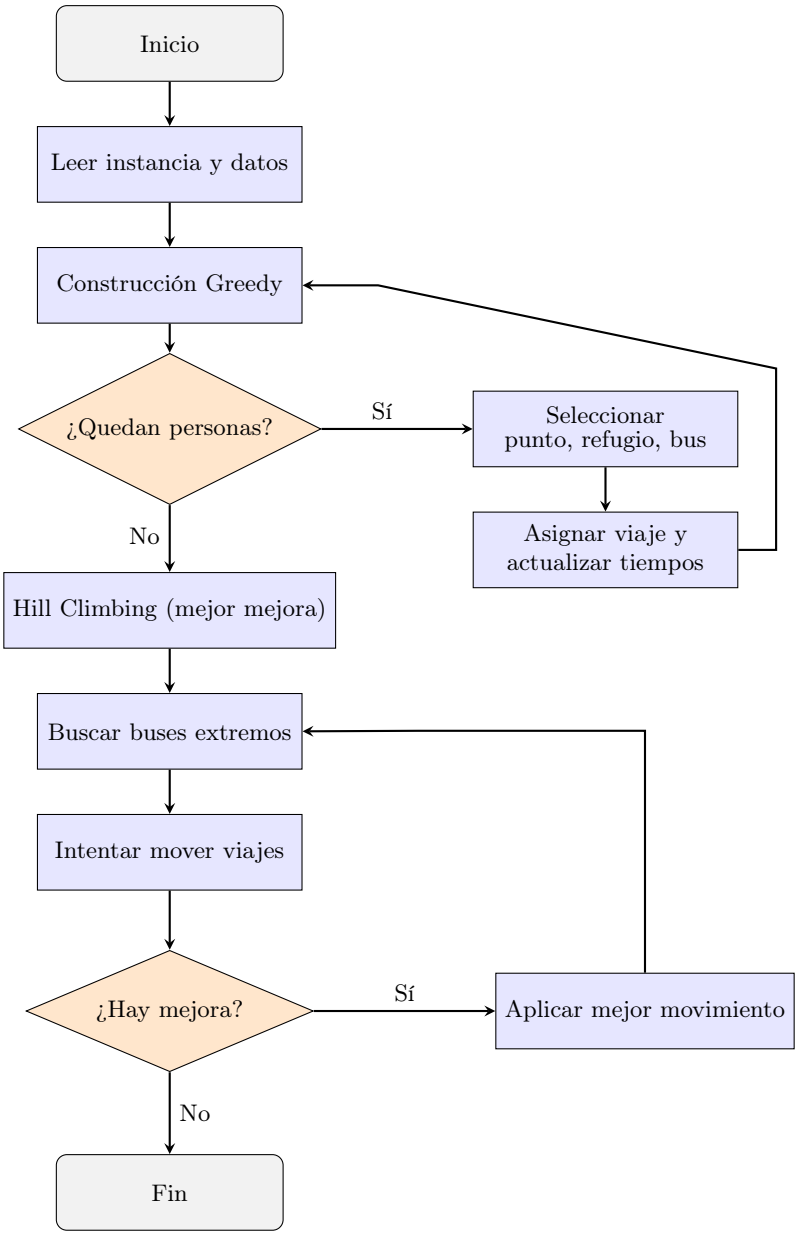
El algoritmo Hill Climbing parte desde una solución factible (Greedy) y explora vecindarios mediante **movimientos** que también **preservan las restricciones** del problema:

- **Cobertura total:** al mover un viaje de un bus a otro, el viaje sigue siendo parte de la solución. No se elimina ni altera el conjunto de personas evacuadas, por lo tanto, la cobertura se conserva intacta.
- **Capacidad del bus:** como los viajes ya fueron previamente construidos bajo esta restricción, mover un viaje completo a otro bus no cambia su cantidad de personas transportadas. No se permite la fusión ni fragmentación de viajes, por lo que no hay riesgo de sobrecarga.
- **Capacidad del refugio:** los viajes no se modifican en su origen ni en su destino. Al moverlos entre buses, la carga total hacia cada refugio no cambia, lo cual preserva automáticamente el cumplimiento de esta restricción.

Por tanto, el espacio de soluciones vecinas que explora el algoritmo está restringido únicamente a soluciones válidas. Esto permite concentrar el criterio de aceptación exclusivamente en la mejora del makespan sin necesidad de revalidar factibilidad en cada iteración.

6.4 Diagrama de Flujo del Algoritmo

Para visualizar de mejor forma el flujo del algoritmo se presenta un diagrama de flujo que detalla el recorrido lógico del sistema, integrando tanto la fase de construcción como la de mejora de la solución:



Este diagrama ilustra con claridad la transición entre fases: desde la lectura de datos, la construcción inicial basada en una heurística Greedy (que evacúa personas mientras sea posible), hasta la fase de refinamiento mediante Hill Climbing, en la que se intentan reasignaciones entre buses para minimizar el makespan. El flujo se detiene cuando no se encuentran más mejoras posibles.

En general, el algoritmo implementado busca balancear calidad de solución y eficiencia. La heurística Greedy garantiza factibilidad, mientras que Hill Climbing permite mejorar el makespan sin comprometer restricciones.

7 Experimentos

Para evaluar el rendimiento del algoritmo propuesto, se diseñó un conjunto de experimentos controlados en los que se analizaron distintas variantes del enfoque heurístico combinado (Greedy + Hill Climbing), así como también la sensibilidad del algoritmo frente a mejoras estructurales. Cada combinación se evaluó bajo el mismo conjunto de instancias y condiciones experimentales.

Cabe destacar que todos los experimentos se ejecutaron en un entorno controlado bajo **Ubuntu 24.04.1 LTS** utilizando **WSL (Windows Subsystem for Linux)**. El hardware correspondía a un procesador **Intel(R) Core(TM) i7-8565U @ 1.80GHz** con **8 núcleos lógicos** y **7,7 GiB de memoria RAM**. El código fue desarrollado en `C++17` y compilado con `g++` empleando la bandera de optimización `-O2`.

No se utilizó paralelismo ni ejecución multihilo, con el objetivo de mantener condiciones uniformes en todos los experimentos.

7.1 Instancias utilizadas

Para la evaluación del algoritmo se utilizaron las instancias oficiales entregadas para el desarrollo del proyecto, las cuales siguen una estructura estandarizada descrita en el documento de instrucciones. Estas instancias simulan escenarios de evacuación realistas con complejidades crecientes. Cada instancia sigue la nomenclatura:

$$\texttt{InstanceBEP-\#\_}|Y|-|P|-|S|-|B|.txt$$

Cada archivo define:

- El número de estaciones $|Y|$, puntos de encuentro $|P|$, refugios $|S|$ y buses $|B|$.
- La cantidad de personas por punto y la capacidad de cada refugio.
- Las distancias entre estaciones, puntos de encuentro y refugios.

Estas instancias presentan una diversidad de tamaños: desde configuraciones pequeñas (como la Instancia 1, con 4 puntos y 2 refugios) hasta estructuras más complejas con 40 personas y 20 refugios. Esto permite analizar la escalabilidad de cada estrategia. Todas las instancias fueron entregadas como parte del proyecto académico y reproducen condiciones realistas pero controladas.

7.2 Configuraciones probadas

Se evaluaron cinco variantes del algoritmo heurístico:

- **Greedy original + HC (alguna mejora)**: se aplica el primer movimiento que mejora el makespan, sin buscar el mejor posible.
- **Greedy original + HC (mejor mejora)**: se evalúan todos los movimientos posibles y se aplica aquel que produzca la mayor mejora en el makespan.
- **Greedy original + HC (intercambio + mejor mejora)**: se permite intercambiar viajes entre buses en lugar de moverlos unidireccionalmente, aplicando siempre la mejor mejora.
- **Greedy original + HC (iterativo + mejor mejora)**: se repite la búsqueda de la mejor mejora mientras sigan existiendo movimientos que reduzcan el makespan.
- **Greedy mejorado + HC (mejor mejora)**: versión mejorada del algoritmo Greedy que prioriza buses menos cargados y refugios más cercanos, seguida por Hill Climbing con mejor mejora.

Todas las variantes comparten los siguientes parámetros:

- Se selecciona como heurística base el menor tiempo adicional por viaje.
- No se permite dividir un viaje: cada asignación se ejecuta como bloque indivisible.
- El criterio de término en Hill Climbing es no encontrar mejoras adicionales en una pasada completa.

Cada variante del algoritmo fue ejecutada sobre las mismas 9 instancias. Dado que el algoritmo no utiliza aleatoriedad, no fue necesario realizar múltiples ejecuciones por semilla. La comparación se centró en analizar:

- El **makespan final** obtenido tras aplicar Hill Climbing.
- El **tiempo promedio** de ejecución por instancia.
- La **diferencia relativa** entre soluciones greedy y soluciones mejoradas.

Este enfoque permite observar el efecto de cada estrategia sobre el rendimiento del algoritmo sin introducir sesgos estocásticos, favoreciendo una comparación directa entre variantes.

Aunque el algoritmo propuesto no compite directamente con los modelos exactos desarrollados por Goerigk [6] o Dikas [4], su evaluación se fundamenta en instancias inspiradas en dichas formulaciones, lo que permite establecer un marco comparativo razonable. Esta diferencia de enfoque se explica por las características intrínsecas del *Bus Evacuation Problem* (BEP), un problema de naturaleza combinatoria cuya resolución óptima es computacionalmente intratable en casos de tamaño moderado a grande.

Los enfoques exactos buscan garantizar la optimalidad mediante técnicas como programación entera mixta (MIP), *branch and bound* o generación de columnas, y aunque han logrado importantes avances en la calidad de las soluciones, su tiempo de ejecución crece exponencialmente con el tamaño del problema. En particular, trabajos como el de Goerigk plantean formulaciones detalladas del BEP-I con tiempos de resolución que escalan mal ante variaciones en la cantidad de buses, puntos de evacuación o personas. Dikas, por su parte, propone mejoras mediante métodos híbridos que combinan exactitud y metaheurísticas, pero que aún requieren recursos computacionales considerables, especialmente en instancias con múltiples restricciones prácticas (como ventanas temporales, transferencias o prioridades médicas).

Por esta razón, el algoritmo heurístico propuesto no busca reemplazar estos modelos, sino ofrecer una alternativa eficiente y flexible que permita obtener soluciones de buena calidad en tiempos reducidos. Esto es especialmente relevante en contextos operacionales donde el tiempo de cómputo es crítico, como sistemas de evacuación en tiempo real, simulaciones iterativas de planificación o herramientas de soporte a la toma de decisiones. En estos escenarios, sacrificar una pequeña proporción de optimalidad a cambio de una respuesta rápida puede ser no solo aceptable, sino deseable.

8 Resultados

Esta sección presenta y analiza los resultados obtenidos al evaluar las distintas variantes del algoritmo propuesto para resolver el extitBus Evacuation Problem (BEP). Se comparan los valores de makespan final obtenidos tras aplicar Hill Climbing a cada heurística, considerando las 9 instancias descritas previamente. Además, se incluye un análisis comparativo entre configuraciones y se selecciona la más eficiente.

8.1 Comportamiento según tamaño de las instancias

Las 9 instancias utilizadas no están ordenadas estrictamente por tamaño, pero representan una progresión en términos de complejidad: mayor número de puntos de encuentro, refugios y personas por evacuar. Esta característica permite

observar cómo se comportan las variantes del algoritmo frente a un aumento general de la carga computacional.

Observaciones:

- Las instancias más pequeñas (1 y 2) muestran tiempos de ejecución bajos en todas las variantes (entre 1,2 y 1,8 ms), con diferencias mínimas entre estrategias.
- A medida que las instancias aumentan (como la 6, 8 y 9), se observa un incremento general en los tiempos de ejecución, especialmente en la variante **Greedy mejorado + HC + mejor mejora**, que alcanza hasta 4,09 ms (instancia 8).
- La variante **Greedy + HC con intercambio + mejor mejora** presenta un comportamiento más estable, con tiempos entre 1,2 ms y 3,1 ms incluso en instancias grandes, mostrando buena escalabilidad.
- En cuanto a la calidad de solución, el makespan final no crece proporcionalmente al tamaño de la instancia, gracias al efecto de la mejora local. Sin embargo, instancias como la 4 (la más exigente) evidencian diferencias más marcadas entre variantes.

Este análisis sugiere que, si bien el tiempo de ejecución crece con la complejidad de la instancia, el crecimiento es controlado. Además, las estrategias más simples (como alguna mejora) tienden a ser más rápidas, pero menos efectivas. Por el contrario, estrategias más exhaustivas como **mejor mejora** logran mantener la calidad en instancias grandes, sin penalizar significativamente el tiempo.

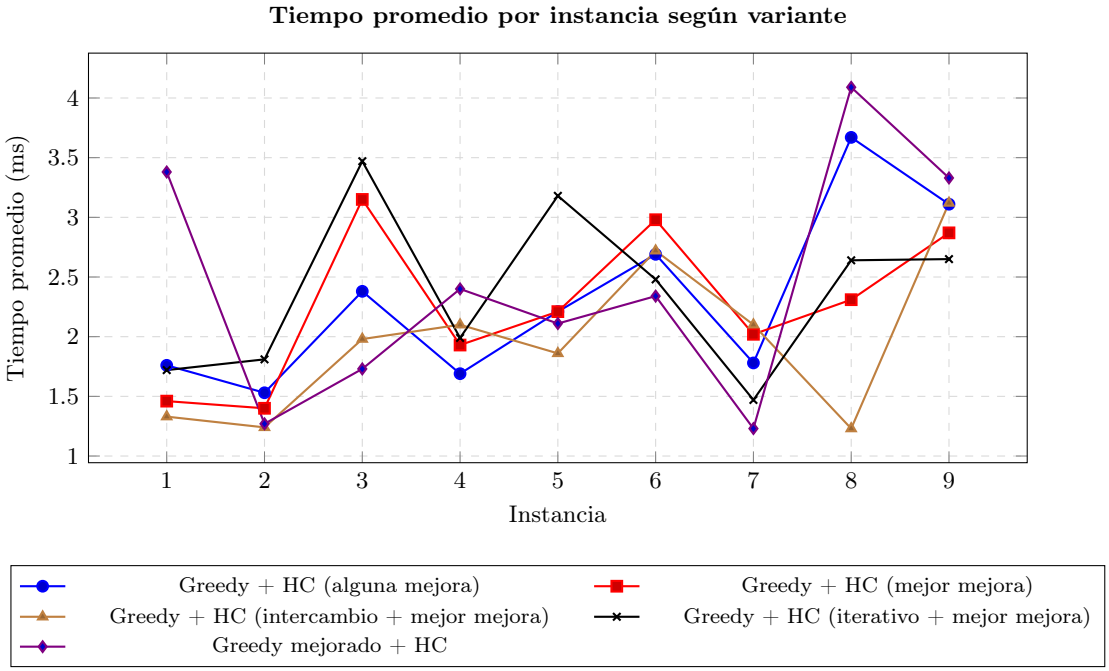


Figure 2: Comparación del tiempo promedio por instancia entre las distintas variantes del algoritmo.

8.2 Comparación entre variantes

En la Figura 3 se muestra la evolución del makespan final (posterior a Hill Climbing) para cada variante del algoritmo, aplicada sobre las 9 instancias del BEP.

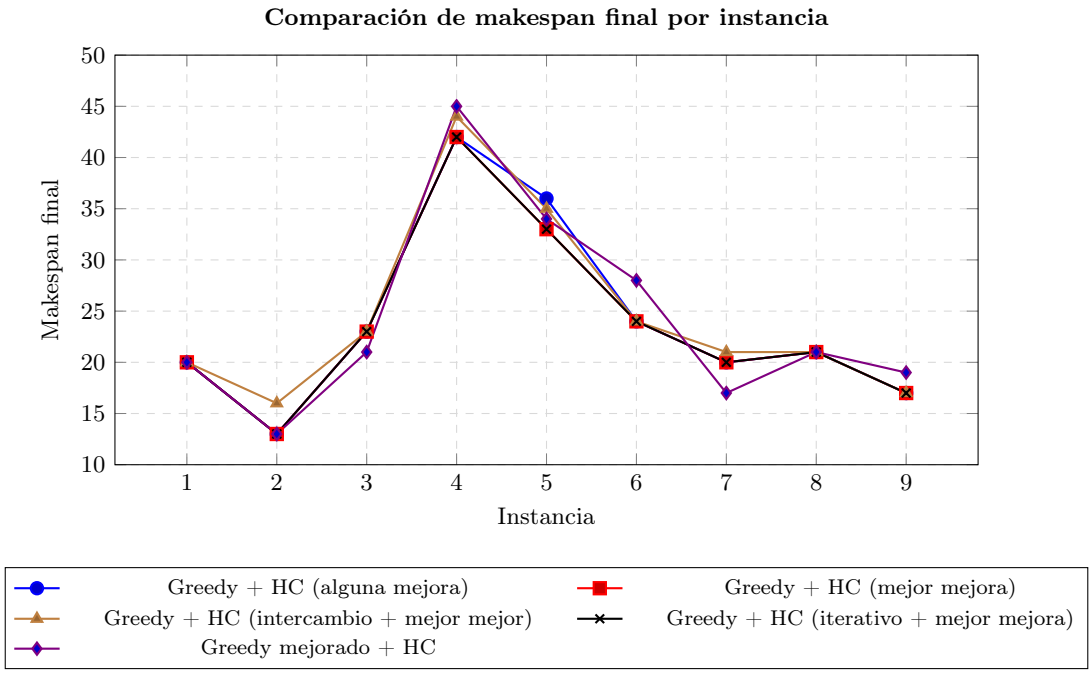


Figure 3: Makespan final obtenido por cada variante del algoritmo sobre las 9 instancias evaluadas.

8.3 Resumen de rendimiento por variante

Variante del algoritmo	**Makespan total final (HC)**	**Tiempo promedio total (ms)**
Greedy + HC (alguna mejora)	216	20,82
Greedy + HC (mejor mejora)	213	20,33
Greedy + HC (intercambio)	221	17,68
Greedy + HC (iterativo)	213	21,41
Greedy mejorado + HC	218	21,88

Table 1: Resumen del makespan total final y tiempo total promedio por variante del algoritmo evaluadas sobre 9 instancias.

8.4 Análisis y selección del mejor enfoque

Se observa que todas las variantes basadas en Hill Climbing logran mejorar los makespanes iniciales generados por las soluciones Greedy. La estrategia de **mejor mejora** aplicada al Greedy original consigue el menor makespan total (213), igualando el rendimiento del enfoque iterativo, pero con un **menor tiempo total de ejecución** (20,33 ms vs. 21,41 ms).

Aunque el Greedy mejorado obtiene buenos resultados en ciertas instancias, no logra superar de forma consistente al Greedy original con mejor mejora. Por tanto, se concluye que el enfoque **Greedy original + Hill Climbing con mejor mejora** es el más eficaz en términos de calidad de solución y simplicidad de implementación.

Además, esta configuración muestra un equilibrio favorable entre tiempo de ejecución y rendimiento, lo que la convierte en una excelente candidata para su uso en contextos reales de evacuación donde el tiempo de respuesta es crucial.

Los resultados muestran que:

- Las variantes que aplican **mejor mejora** tienden a converger a mejores soluciones sin sacrificar eficiencia.
- El intercambio de viajes introduce diversidad, pero no mejora de forma consistente la calidad global, aunque sí reduce el tiempo de ejecución.
- La mejora iterativa no aporta diferencia significativa respecto a una sola pasada de mejor mejora, pero sí incrementa ligeramente el tiempo.
- Una buena solución inicial (como la del Greedy mejorado) puede ayudar, pero no es determinante si el proceso de mejora posterior es suficientemente sólido.

En síntesis, se valida empíricamente que un enfoque sencillo y bien calibrado puede competir con propuestas más complejas, siempre que se utilicen operadores de mejora efectivos.

9 Conclusiones

El estudio del *Bus Evacuation Problem (BEP)* ha permitido comprender la complejidad de planificar evacuaciones masivas bajo restricciones logísticas, sociales y computacionales. A través del análisis del estado del arte, se identificaron enfoques exactos, heurísticos e híbridos que abordan el problema desde distintas perspectivas. Si bien los modelos exactos, como los propuestos por Goerigk [6] o Dikas [4], garantizan optimalidad, su escalabilidad es limitada. En contraste, los modelos agregados como BEP-II y NFP, o las heurísticas avanzadas, ofrecen una mayor eficiencia computacional a costa de precisión.

En este contexto, el algoritmo propuesto —basado en una heurística **Greedy** seguida de un proceso de **Hill Climbing con mejor mejora**— busca un equilibrio entre simplicidad, calidad de solución y tiempo de ejecución. Su implementación fue cuidadosamente diseñada para mantener validez en cada paso, permitiendo que las mejoras locales se realicen sin violar restricciones clave del BEP.

Los resultados experimentales evidencian que todas las variantes probadas mejoran el makespan de la solución inicial. En particular, la configuración **Greedy original + Hill Climbing con mejor mejora** demostró ser la más

efectiva, logrando el menor makespan total (213) con un tiempo promedio competitivo (20,33 ms), lo cual la convierte en una solución eficaz y de bajo costo computacional.

El análisis comparativo por instancia permitió observar tendencias claras:

- Las variantes más exhaustivas tienden a ofrecer soluciones de mayor calidad, aunque con un ligero incremento de tiempo.
- Las variantes con intercambio de movimientos no siempre logran mejores makespan, pero sí destacan por su estabilidad en tiempos de ejecución.
- La variante iterativa no aportó mejoras significativas respecto a una sola pasada de mejor mejora.
- El algoritmo Greedy mejorado mostró beneficios iniciales, pero no fue suficiente para superar consistentemente al enfoque base con buena mejora posterior.

Si bien el algoritmo no compite directamente con modelos exactos en calidad teórica, su rendimiento práctico es más que suficiente para aplicaciones en tiempo real o sistemas de soporte a la decisión. Su principal debilidad radica en la **falta de diversidad de operadores de movimiento**, lo que limita la exploración del espacio de soluciones. Además, la heurística Greedy podría beneficiarse de un mayor grado de aleatoriedad o ponderación multicriterio para evitar soluciones sesgadas.

A partir de estos hallazgos, se sugieren las siguientes líneas de mejora y proyección:

- Incorporar operadores de movimiento más complejos como inserción, *swap* múltiple o reordenamiento local.
- Integrar técnicas como *Simulated Annealing* o *Variable Neighborhood Search (VNS)* para escapar de óptimos locales.
- Diseñar una heurística Greedy no determinista, que explore diversas soluciones iniciales.
- Aplicar los algoritmos a instancias reales o simuladas de gran escala, usando datos geoespaciales reales.
- Implementar versiones paralelas del algoritmo para reducir aún más los tiempos de cómputo.

En conclusión, el BEP sigue siendo un problema activo de investigación con alto impacto práctico. El enfoque propuesto confirma que soluciones simples, bien diseñadas y eficientes pueden ofrecer resultados competitivos frente a métodos más complejos, siempre que se ajusten a los contextos operacionales donde la velocidad y la factibilidad son más relevantes que la optimalidad estricta.

10 Bibliografía

References

[1] Douglas R. Bish. Planning for a bus-based evacuation. *OR Spectrum*, 33(3):629–654, 2011. Usado en Estado del Arte, introducción al problema (pp. 630–632).

[2] Jean-François Cordeau, Michel Gendreau, Gilbert Laporte, Jean-Yves Potvin, and François Semet. A guide to vehicle routing heuristics. *Journal of the Operational Research Society*, 53(5):512–522, 2002. Usado en Estado del Arte, antecedentes sobre métodos heurísticos generales.

[3] Kaouthar Deghdak, Vincent T'kindt, and Jean-Louis Bouquard. Heuristics for scheduling evacuation operations in case of natural disaster. In *Proceedings of the 3rd International Conference on Operations Research and Enterprise Systems (ICORES)*, pages 294–300. SCITEPRESS, 2014. Usado en Estado del Arte, sección sobre heurísticas y matheurísticas.

[4] George Dikas and Ioannis Minis. Solving the bus evacuation problem and its variants. *Computers & Operations Research*, 70:75–86, 2016. Usado en Estado del Arte, sección sobre heurísticas e integración (pp. 78–82).

[5] Yuanyuan Feng, Yi Cao, Shuanghua Yang, Lili Yang, and Tangjian Wei. A two-step sub-optimal algorithm for bus evacuation planning. *Operational Research*, 23(36), 2023. Usado en Modelo Matemático (pp. 5–7) y Estado del Arte (pp. 3–6).

[6] Marc Goerigk, Bob Grün, and Philipp Heßler. Branch and bound algorithms for the bus evacuation problem. *Computers & Operations Research*, 40(12):3010–3020, 2013. Usado en Modelo Matemático (pp. 3013–3016) y Estado del Arte (pp. 3011–3012).

[7] A. H. G. Rinnooy Kan and J. Karel Lenstra. Complexity of vehicle routing and scheduling problems. *Networks*, 11(2):221–227, 1981. Usado en Definición del Problema, para explicar la complejidad NP-hard del VRP.

[8] Jingqing Li, Abdeslam Boularias, and Steven L. Waslander. Reinforcement learning for fleet management in mobility-on-demand systems. *IEEE Transactions on Intelligent Transportation Systems*, 20(12):4449–4460, 2019. Usado en Estado del Arte, como ejemplo de planificación adaptativa con IA.

[9] Aida Calviño Martínez. Cooperación en los problemas del viajante (tsp) y de rutas de vehículos (vrp): una panorámica. Universidad Complutense de Madrid, 2011. Usado en Estado del Arte, antecedentes históricos y colaborativos del VRP.

[10] Luis C. C. Pedrosa and Rafael C. S. Schouery. Approximation algorithms for the bus evacuation problem. *Journal of Combinatorial Optimization*,

36(1):131–141, 2018. Usado en Estado del Arte, sección sobre algoritmos de aproximación (pp. 134–136).

[11] Patrick Prosser. Hybrid algorithms for the constraint satisfaction problem. *Computational Intelligence*, 9(3):268–299, 1993. Usado en Estado del Arte, referencia a problemas de satisfacción de restricciones.

[12] Stefan Ropke and David Pisinger. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4):455–472, 2006. Usado en Estado del Arte, referencia para técnicas LNS aplicadas al BEP.

[13] Paolo Toth and Daniele Vigo. An overview of vehicle routing problems. In P. Toth and D. Vigo, editors, *The Vehicle Routing Problem*, pages 1–26. Society for Industrial and Applied Mathematics (SIAM), 2002. Usado en Estado del Arte, para contextualizar la evolución de VRP y variantes como el BEP.