TRABAJO FIN DE MÁSTER

Máster Universitario en Formación del Profesorado de Educación Secundaria Obligatoria, Bachillerato, Formación Profesional y Enseñanza de Idiomas y Máster Universitario en Matemáticas

Diseño de redes de ferrocarril: cobertura por pares vs. cobertura por nodos

Antonio Jesús Varo Borrego

Tutor académico: Federico Perea Rojas-Marcos

Curso 2023-2024

Sevilla, 2 de Febrero de 2024

Resumen

En los últimos años, se ha observado un creciente interés y esfuerzo en la construcción, mejora y expansión de redes de transporte rápido, impulsado por la creciente demanda de viajes, congestión del tráfico, la necesidad de reducir la contaminación y el consumo de energía. Este fenómeno responde a la insostenibilidad de los combustibles fósiles en el transporte personal.

Sin embargo, el diseño de estas redes presenta desafíos considerables. Las infraestructuras son difíciles de modificar, y la participación de diversos agentes, como políticos, ciudadanos e ingenieros provocan una dificultad añadida para resolver el problema. Los diseños urbanísticos de las ciudades tienen un impacto sobre la planificación de la red al igual que la red sobre el desarrollo urbanístico de la ciudad, esto hace que sea importante un buen estudio.

Las inversiones en estas redes son muy altas, por lo que hay que definir muy bien los objetivos que se quieren alcanzar basados en las necesidades de la población y todo lo que rodea la construcción de una red de transporte rápido. Es por ello que se buscan modelos matemáticos cada vez más cercanos a la realidad, aunque esto conlleva una superior complejidad computacional que en muchos casos no es posible solventar. Los objetivos mas habituales dentro de la literatura en este campo son el maximizar la población cubierta por las estaciones y el de maximizar la cobertura de la demanda origen-destino, entre otros. Para llegar a estos objetivos se plantean diversos modelos algorítmicos, entre ellos los modelos de programación lineal entera que serán la base de los modelos planteados en este trabajo.

En este Trabajo de Fin de Máster propondremos dos modelos, el modelo 1 buscará la maximización de la cobertura de la demanda origen-destino cubierta por la red. El modelo 2, tendrá como función objetivo el maximizar la población cubierta por las estaciones. También se propondrá una modificación del modelo 2 con el fin de obtener distintas redes en poco tiempo. Posteriormente, tomaremos datos experimentales con el objetivo de comprobar el comportamiento de cada uno de los modelos, tanto en términos de la bondad de la solución, como en términos de la complejidad computacional. Se compararán las redes obtenidas en cada modelo, observando cuanto se acercan a la red óptima del otro modelo en términos de la función objetivo. El principal objetivo es que el modelo dé la mejor solución posible, en términos de la cobertura de la demanda origen-destino, en un tiempo razonable.

Abstract

In recent years, there has been growing interest and effort in the construction, improvement, and expansion of rapid transit networks, driven by increasing travel demand, traffic congestion, the need to reduce pollution, and energy consumption. This phenomenon is a response to the unsustainable nature of fossil fuels in personal transportation.

However, the design of these networks bring up significant challenges. These infrastructures are difficult to modify, and the involvement of various stakeholders, including politicians, citizens, and engineers, adds complexity to the problem-solving. City urban designs impact network planning, just as the network influences city development, highlighting the importance of thorough studies.

This networks need substantial financial investment, so it is essential to have clear objectives based on population needs, and surrounding factors must be defined in the construction of rapid transit networks. Hence, there is a pursuit of mathematical models that closely represent reality, despite the increased computational complexity that is sometimes impossible to solve. Common objectives in the literature include maximizing station-covered population and maximizing coverage of origin-destination demand.

This study will conduct an analysis of solutions, focusing on maximizing the population covered by stations versus maximizing the coverage of origin-destination demand by the network. Two models will be proposed, each targeting one of these objectives.

Using experimental data, the performance of each model will be evaluated in terms of computational complexity and solution quality. Network outcomes from each model will be compared to assess how closely they approach the optimal network of the other model in terms of the objective function. The primary goal is for the model to provide the best possible solution in a reasonable amount of time.

Índice general

1. Introducción **3**

2. Diseño de Redes **7**

2.1. Construcción de la red 8

2.1.1. Objetivos 9

2.1.2. Localización de redes de transporte rápido: modelos 13

3. Modelos para el diseño de Redes **16**

3.1. Modelo 1 16

3.1.1. Datos y notación 17

3.1.2. Variables 17

3.1.3. Función objetivo y restricciones 18

3.2. Modelo 2 20

3.2.1. Datos y notación 20

3.2.2. Variables 21

3.2.3. Función objetivo y restricciones 21

3.2.4. Modificación del modelo 2 21

3.2.5. Post Modelo 22

4. Entorno tecnológico **24**

4.1. AMPL 24

4.1.1. Solver 27

4.2. Python 28

5. Resultados experimentales **29**

5.1. Análisis por nodos 37

6. Conclusiones **40**

6.1. Dificultades encontradas 41

7. Anexos **43**

7.1. Códigos empleados 43

7.1.1. Generación de datos en Python 43

7.2. Modelos AMPL 46

7.2.1. Modelo 1 46

7.2.2. Modelo 2 modificado 47

Capítulo 1

Introducción

En los últimos años, se ha dedicado mucho esfuerzo a la construcción, mejora o expansión de las redes de transporte rápido. Este fenómeno está motivado por el aumento en la demanda de viajes, la congestión del tráfico en las ciudades, la creciente longitud de los trayectos, la necesidad de reducir el consumo de energía y la contaminación [Gendreau et al. (1995)], además de la insostenibilidad del consumo de combustibles fósiles de los transportes personales.

Este diseño de redes se encuentra muchas dificultades o problemas a tratar, debido a que estas infraestructuras no son de fácil modificación y que hay muchos agentes involucrados. Dentro de estos agentes tenemos el factor humano, donde encontramos los políticos, ciudadanos, ingenieros..., generando un conflicto de intereses entre ellos. Otro agente que encontramos son los diseños urbanísticos de la ciudad y el impacto que tendrá esta red sobre él. A la hora de construir una nueva red de transporte rápido es importante prestar especial atención a la inversión necesaria en el proceso de construcción debido a su elevado coste y que, como hemos dicho antes, estas infraestructuras no pueden modificarse fácilmente en un corto plazo. Esto hace que en ciudades

donde ya tienen una red, su principal inversión radica en la construcción de nuevas líneas que mejoren la red ya existente.

Finalmente, también encontramos el agente matemático que para abordar el problema con la mayor exactitud necesitará de gran número de restricciones, lo que complica mucho el modelo. Esto se verá con mayor detenimiento en los capítulos posteriores. Esta dificultad provoca que, en muchos casos, no sea realista la aplicación directa de los modelos matemáticos. Algunos autores [Laporte et al. (2011b)] han observado con el análisis de las redes existentes y planificadas que los organizadores de los proyectos de redes de transporte rápidos utilizan poco o nada la investigación operativa.

En el Capítulo 2 veremos como se ha tratado este problema con los distintos modelos encontrados en la literatura.

Para ello será necesario introducir unas nociones previas:

- Teoría de grafos.

  La principal herramienta que se utiliza a la hora del diseño de redes es la teoría de grafos. Un grafo $G(N, A)$ es un conjunto finito de objetos llamados nodos (N) y aristas (A) que unen pares de estos nodos, esta unión representa una relación binaria entre los elementos. Esta unión puede ser dirigida o no dirigida, según si la relación es recíproca o no. A estas aristas se les puede asignar un valor numérico o pesos, que según el contexto describirán distintos valores como costes, distancias, capacidad...

- Optimización combinatoria.

  Para la resolución de estos problemas de redes es necesario tener conocimientos de optimización combinatoria. Ésta se centra en encontrar la mejor solución dentro del conjunto de soluciones factibles. Depen-

diendo de la bondad de la solución podemos clasificar los algoritmos de resolución de problemas de optimización combinatoria en tres grupos:

1. Algoritmos exactos: Estos se basan en encontrar la solución óptima, es decir, la solución factible que obtiene mejor valor objetivo. Estos algoritmos son complicados desde el punto de vista de la complejidad computacional, especialmente si se quiere resolver un problema NP-duro[1]. Es por ello que cuando el tamaño del problema crece no son capaces de dar una solución óptima en un tiempo razonable.

2. Algoritmos aproximados: Estos algoritmos proporcionan una solución factible del problema, no necesariamente óptima, pero nos garantizan un mínimo teórico de bondad en la solución. Además, generalmente estos algoritmos son más rápidos que los exactos.

3. Algoritmos heurísticos: Al igual que los aproximados, estos algoritmos nos proporcionan una solución factible del problema, pero en este caso no se puede garantizar la bondad de la solución teóricamente, y se recurre a los resultados experimentales. Suelen ser los más rápidos de los tres tipos, por lo que en muchos casos puede ser interesante diseñar una heurística, ya sea porque el problema tenga un tamaño muy grande, haya poco tiempo disponible y en muchos casos es mejor una solución aproximada de un problema exacto que una solución exacta de un problema aproximado.

[1]NP-duro: El problema es al menos tan complejo como cualquier problema NP, por lo que no hay un algoritmo conocido que pueda encontrar la solución en un tiempo polinómico para todos los casos.

Los algoritmos propuestos en este TFM serán exactos, aunque el segundo de los modelos buscará una buena solución del problema reduciendo el tiempo de computación. En concreto, serán modelos de programación lineal entera mixta. Estos problemas son NP-duros, por lo que es importante que estos modelos estén bien diseñados para reducir al máximo el tiempo de cómputo. Alguno de los métodos para la resolución de modelos de programación lineal entera son los planos de cortes, "ramificación y acotación", y en el que se basa el solver que utilizaremos "ramificación y cortes" que combina tanto planos de corte como "ramificación y acotación".

En el Capítulo 3 introduciremos los dos modelos que vamos a comparar, por un lado, el que maximiza la cobertura por pares y por otro el que maximiza la cobertura por nodos.

En los Capítulos 4 y 5, introduciremos el entorno tecnológico usado, así como los solvers usados para obtener los resultados experimentales.

Finalmente, en el capítulo 6 realizaremos un repaso de los resultados obtenidos, obteniendo las conclusiones y resultados principales.

Capítulo 2

Diseño de Redes

Las redes de transporte rápido han demostrado mejorar mucho la movilidad dentro de las ciudades, además de reducir el tráfico y número de vehículos contaminantes. Debido al crecimiento de la población al que se enfrentan las ciudades, se está optando por invertir en la construcción de nuevas redes de transporte en las ciudadades o mejorar las ya existentes. Estas redes de transporte rápido, a menudo, se opta por que sean subterráneas, pues así serán independientes al tráfico de otros medios de transporte y causan un menor impacto urbanístico.

En la construcción de una red de transporte rápido se ven involucrados factores humanos como pueden ser ingenieros, políticos, organizaciones ciudadanas... o factores de conservación del patrimonio, conservación del urbanismo, crecimiento a futuro de la ciudad, características del terreno.

Además, a la hora de planificar una red es importante conocer la movilidad actual de la ciudad, las necesidades de los ciudadanos, las redes ya existentes tanto del mismo medio de transporte como de otros medios. Todo esto hace que la planificación sea una tarea muy compleja. Es por ello que la primera etapa de la planificación consiste en un análisis minucioso de la

ciudad o zona donde se pretende construir o modificar la red, evaluando la densidad de población dentro de la ciudad, los patrones de desplazamientos habituales, identificar posibles conexiones con otros medios de transporte, etc. A menudo, se evalúan estas posibles redes desde varios objetivos distintos, analizando cada uno de ellos con detalle. Esto hace que la planificación de las redes acabe siendo un problema estratégico que conlleva gran cantidad de tiempo.

Centrándonos en el plano matemático, tener en cuenta tantos factores, así como la poca claridad en el objetivo y restricciones de cada uno, hace que sea muy complicado obtener modelos que se reflejen exactamente la realidad. Además, un modelo que represente muy bien la realidad tendría una complejidad computacional muy alta, por lo que no sería posible obtener la solución en un tiempo razonable.

Todo esto hace que parezca imposible obtener un modelo exacto para la planificación de estas redes, pero veremos como podemos aproximarnos bastante para obtener buenos modelos que ayuden a la hora de planificar nuevas redes, analizar distintas opciones de rutas, así como simular el impacto que tendría dentro de la movilidad de la urbe o el impacto que tendría en una futura ampliación.

2.1. Construcción de la red

En esta sección veremos los pasos a seguir en la construcción de una red de tránsito rápido. Para ello revisaremos distintos modelos propuestos en la literatura, observando los distintos objetivos que se proponen, distintas medidas para calcular la eficiencia de las redes y como paulatinamente se van añadiendo factores para dar mayor realismo al problema.

2.1.1. Objetivos

El objetivo de los sistemas de transportes colectivos es mejorar la movilidad de la población. Como estos sistemas tienen gran capacidad, reducen significativamente el tráfico, la contaminación, el consumo de energía, dando una movilidad sostenible. Además, estos sistemas tienden a ofrecer tiempos de desplazamientos más cortos.

Estos sistemas también influyen en la estructura urbana al articular el desarrollo de áreas residenciales, comerciales y empresariales. Sin embargo, la implementación de sistemas de tránsito rápido implica inversiones significativas en construcción y mantenimiento, principalmente relacionadas con la construcción de túneles, sistemas de comunicación, tasas de pago y adquisición de material.

Hay tres grupos principales involucrados en la planificación de una red: la sociedad en general, representada por agencias de transporte y delegados gubernamentales; los potenciales usuarios; y la empresa que proporciona el servicio, que puede ser tanto de carácter público como privado. Estos actores desempeñan un papel clave en la toma de decisiones y la implementación de sistemas de tránsito rápido.

Es habitual a la hora de la planificación tomar en consideración el potencial de población cubierta por la red. Como dice [Vuchic (2005)] podemos entender que una parada cubre a una población ubicada aproximadamente en un radio de 400 m o un tránsito peatonal de 5 minutos en zonas de alta densidad de población, mientras que para zonas de baja densidad este límite puede llegar hasta 1 km.

Otra medida utilizada reside en el potencial de la red para cubrir las demandas Origen-Destino de la población. Estas demandas se deben estudiar a la par que las alternativas de desplazamientos disponibles. Los potenciales

usuarios no solo buscarán la conectividad que proporciona este medio de transporte, sino también el tiempo que se tarda en realizar dicho trayecto.

Un objetivo secundario es el minimizar el número de transbordos entre líneas, ya que esto reduce el tiempo de desplazamiento y mejora el confort del usuario.

Finalmente, encontramos un tercer objetivo, que pretende optimizar la gestión de costes tanto en la construcción, el mantenimiento y los ingresos potenciales.

Estas medidas son habitualmente usadas como función objetivo y restricciones en modelos de programación matemática.

Otra posible medida destinada a aumentar la eficiencia de una red, la propone [Laporte et al. (1997)] valorando la red a través de población/red (dividiendo el tiempo mínimo de viaje entre el tiempo total de la red) y de población/plano (que nos mide lo bien que cubre la red el plano). Estos autores se apoyan en estas dos medidas para analizar la utilidad de los diseños predefinidos, de rueda y estrella (Figura 2.1),"U and cross" y de circunferencia (Figura 2.2) y de cuadrícula y triangular (Figura 2.3), sobre una ciudad circular, obteniendo los mejores resultados en los diseños de circunferencia, de rueda y triangulares. Los diseños radiales de estrella y de cuadrículas no obtienen los mejores resultados, pero no tan malos los del diseño como el diseño "U and cross".

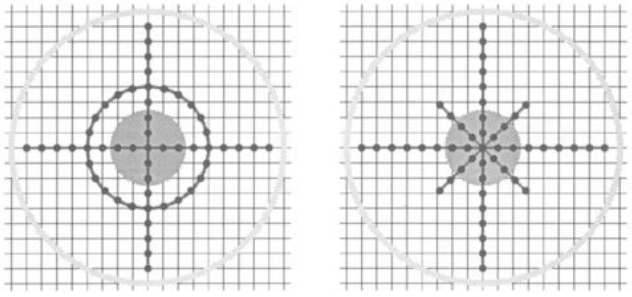


Figura 2.1: Ejemplos de diseños de rueda y de estrella.

Fuente: [Laporte et al. (1997)]

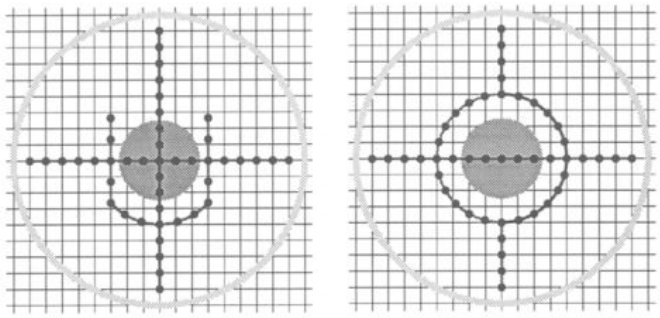


Figura 2.2: Ejemplos de diseños “U and cross” y de circunferencia.

Fuente: [Laporte et al. (1997)]

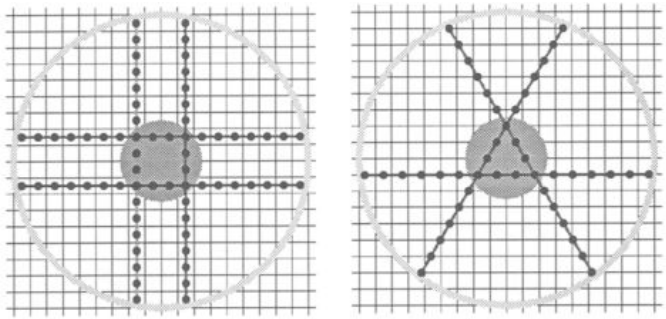


Figura 2.3: Ejemplos de diseños de cuadrícula y triangular.

Fuente: [Laporte et al. (1997)]

Muchos de los estudios actuales analizan las redes en término de robustez ante la presencia de fallos ya sean fortuitos o provocados. Debido a esto se ha visto aumentado el interés del conocimiento de las propiedades estructurales de los sistemas, tomando la teoría de grafos como herramienta para estudiar estas propiedades desde un punto de vista más abstracto.

Alguno de estos indicadores son mencionados por [Laporte et al. (2015)], el coeficiente de agrupamiento C y la longitud del camino característico L. Para ello, dado el grafo, toma d_{ij} como la distancia topológica entre dos vértices (el menor número de aristas para ir de un vértice al otro).

$$C = \frac{3 \times \text{número de triángulos}}{\text{número de tripletas conectadas}}.$$

Se entiende como tripletas a tres vértices donde al menos uno de ellos está conectado a los otros dos, y triángulo a tres vértices donde cada uno está unidos directamente a los otros dos.

$$L = \frac{1}{|V|(|V|-1)} \sum_{i \neq j} d_{ij}.$$

Esta expresión de la distancia media del camino más corto (longitud del camino característico), se puede adaptar a una red métrica. Para ello reemplazaremos esta expresión por una de la eficiencia global y local, ver [Latora y Marchiori (2001)].

$$E_g(G) = \frac{2}{|V|(|V|-1)} \sum_{i<j} \frac{1}{d_{ij}} \quad \text{y} \quad E_l(G) = \frac{1}{|V|} \sum_{v_i \in V} E_g(G_i),$$

donde G_i es el subgrafo de los vecinos de v_i.

2.1.2. Localización de redes de transporte rápido: modelos

Los proyectos de construcción de las redes de transporte rápido se pueden clasificar en tres grupos:

- La planificación de una primera línea.
- La planificación de varias líneas simultáneas desde cero.
- Ampliación de una red ya existente.

Localización de una línea

El problema de localización de una línea en un sistema de tránsito rápido se basa en buscar los caminos y las estaciones idóneas. Desde el punto de vista de la teoría de grafos, el problema consiste en seleccionar el camino entre dos nodos eligiendo nodos intermedios que se tomarán como estaciones, optimizando el objetivo sujeto a ciertas restricciones. Esto lo observamos cuando nuestra red se encuentra sobre la superficie, pues las conexiones están sujetas al urbanismo propio de cada ciudad, mientras que en las construcciones de redes subterráneas podemos tomar un enfoque de red continua.

El primer intento real de resolver este problema lo podemos encontrar en [Dufourd et al. (1996)] teniendo en cuenta una distancia máxima y mínima entre estaciones y un número máximo de ellas, con el objetivo de maximizar la población cubierta por las estaciones. Posteriormente [Bruno et al. (1998)] propone este mismo problema desde un criterio más realista, maximizando la cobertura de viajes que proporciona la línea. Con el fin de buscar más realismo en los modelos, muchos autores consideran la presencia de determinados medios de transporte alternativos, como podemos ver en [Laporte et al. (2005)]

donde presenta un modelo cuyo objetivo es maximizar la cobertura de la demanda origen-destino, con el transporte privado como principal competidor.

Localización de una red

Consideraremos ahora el problema de localizar una red de transporte rápido desde cero o extender una red ya existente.

El primer intento de modelizar y resolver el problema de la red de transporte rápido la encontramos en [Laporte et al. (2007)], donde da una aproximación computacionalmente tratable dividiendo el problema en tres fases:

1. Seleccionar estaciones claves debido a la gran afluencia de población (aeropuertos, hospitales, campus universitarios...) o áreas muy pobladas alejadas del centro de la ciudad.

2. Conectar las estaciones claves con el núcleo principal de la red.

3. Ubicar estaciones intermedias en las rutas de la segunda fase.

En este mismo artículo, se presenta un modelo de programación lineal entera cuyo objetivo es maximizar la cobertura de viajes con la presencia de un medio de transporte alternativo.

Posteriormente, en 2009, con el objetivo de obtener un modelo que interprete mejor el comportamiento del usuario, [Marin y Garcia Ródenas (2009)] introduce una función "logit" con el fin de distribuir los viajeros entre aquellos que usan el transporte rápido y aquellos que usan el transporte privado. Para seguir manteniendo la linealidad del problema realizan una interpolación lineal a trozos de la función logit.

Algunas de las investigaciones más recientes, tratan de valorar la robustez de la red desde el punto de vista de la teoría de juegos [Laporte et al. (2010)] o dando rutas alternativas en caso de interrupción en la vía [Laporte et al. (2011a)].

Localización de estaciones

El problema de localizar las estaciones, es distinto si nos encontramos en el caso de construir una línea desde cero, o si estamos en el caso de ampliación de una red existente. En el primer caso, muchas de las localizaciones atraerán gran volumen de pasajeros y serán candidatas a estaciones. Las demás estaciones deberán ser colocadas con la ayuda de herramientas analíticas. Por otro lado, si ya tenemos una red dada, aparece el problema de localizar eficientemente las nuevas estaciones.

El objetivo prioritario es atraer tantos viajeros como sea posible, analizando la población que vive en un círculo centrado en cada estación. Aunque las distancias andando no son euclídeas, esta medida de la atractividad de la estación es buena. Como se propone en [Laporte et al. (2005)] se puede tomar una probabilidad de atracción de la estación en función de la cercanía y esta no tiene por que ser con la distancia euclídea, también puede ser con la distancia Manhattan.

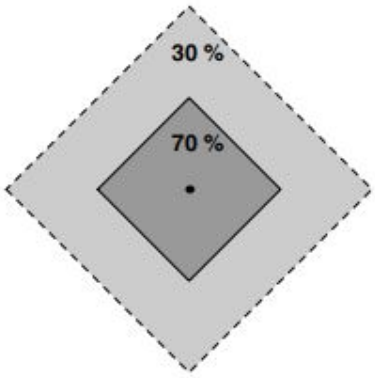


Figura 2.4: Estación con zonas de captación.

Fuente: [Laporte et al. (2005)]

Capítulo 3

Modelos para el diseño de Redes

En este capítulo veremos dos modelos matemáticos para el diseño de redes de transporte rápido. Estos modelos de localización son muy complejos computacionalmente en situaciones realistas, por lo que se evaluarán con bases de datos tratables.

Para estos modelos consideraremos, conocidos los tiempos de viaje entre las potenciales localizaciones de las estaciones, que todas las localizaciones pueden ser unidas, las demandas de viajes de cada una de las posibles estaciones a otra y el tiempo de transporte por un medio de transporte alternativo.

3.1. Modelo 1

Este modelo se basa en el descrito en [Laporte et al. (2011a)], y en él se buscará maximizar la utilidad de la red, en el sentido que sea usada por el mayor número de pasajeros.

3.1.1. Datos y notación

En primer lugar, tenemos una red subyacente $G = (N, A)$ no dirigida asociada al problema, con $N = \{1, ..., n\}$ el conjunto de posibles localizaciones de estación y $A \subseteq \{(i, j) \in N \times N : i \neq j\}$ el conjunto de posibles conexiones entre las estaciones. Definimos el conjunto $W \subseteq \{(k, l) \in N \times N\}$ como el conjunto de pares origen destino.

Cada nodo i tiene asociado un coste de construcción de la estación C_i. Cada arco (i, j) tiene asociado un tiempo de viaje t_{ij} y un coste de construcción de la conexión de C_{ij}. Cada par $(k, l) \in W$ tiene asociado el tiempo de viaje de k a l mediante un medio de transporte alternativo u_{kl} y una demanda de viaje de k a l de g_{kl}.

También tendremos un coste máximo de la red $C_{\text{máx}}$.

3.1.2. Variables

Todas las variables usadas en este modelo son binarias.

Tendremos unas variables que nos permiten decidir qué estaciones y conexiones se construyen. Éstas son $y_i = 1$ si se construye la estación i y $x_{ij} = 1$ si se construye la conexión (i, j).

Por otro lado, tenemos una variable $f_{ij}^{kl} = 1$ que toma valor 1 si la demanda $(k, l) \in W$ usa el arco de la red (i, j). Esta variable nos permite controlar que el flujo de pasajeros sea equilibrado en cada nodo.

Finalmente, una variable $z_{kl} = 1$ si la demanda $(k, l) \in W$ es captada por nuestra red. Consideraremos que un par (k, l) es captado por la red si el tiempo que emplea utilizando dicha red es inferior al tiempo que necesitaría utilizando el medio alternativo de transporte u_{kl}.

3.1.3. Función objetivo y restricciones

El objetivo de nuestro problema será maximizar la demanda captada, es decir,

$$\text{máx} \sum_{(k,l)\in W} g_{kl} z_{kl}. \tag{3.1}$$

Las restricciones que nos encontramos son:

Restricción de presupuesto, ya que no disponemos de dinero ilimitado. Ésta será que la suma de los costes multiplicados por la variable si se construye estación y conexión debe ser menor que el presupuesto $C_{\text{máx}}$:

$$\sum_{(i,j)\in A} C_{ij} x_{ij} + \sum_{i\in N} C_i y_i \leq C_{\text{máx}} \tag{3.2}$$

Restricciones de diseño de la red:

$$x_{ij} = x_{ji} \quad \forall (i,j) \in A \tag{3.3}$$

$$x_{ij} \leq y_i \quad \forall (i,j) \in A \tag{3.4}$$

$$y_i \leq \sum_{j\in N:(i,j)\in A} x_{ij} + \sum_{j\in N:(j,i)\in A} x_{ji} \quad \forall i \in N \tag{3.5}$$

$$\sum_{i\in N} y_i \leq \sum_{(i,j)\in A} \frac{x_{ij}}{2} + 1 \tag{3.6}$$

La restricción (3.3) nos indica que si se construye una conexión también se ha construido la de sentido inverso, dándole el carácter de no dirigido al grafo. La restricción (3.4) nos refleja que si se construye una conexión, se deben construir las estaciones inicio y fin de esa conexión. La restricción (3.5) nos asegura que si hemos construido una estación, al menos debe tener una conexión con otra estación. Finalmente, tenemos la restricción (3.6) basada en la restricción de estructura de árbol que encontramos en algunos modelos de diseños de redes con líneas como en [Laporte et al. (2011a)] con el fin de obtener al menos una línea principal, pero realizamos una relajación permitiendo subciclos dentro de la red.

Restricciones de flujo: estas restricciones están dirigidas a que si para ir de k a l se pasa por i entonces tiene que entrar y salir de i, y si i es terminal entonces solo entra o solo sale.

$$\sum_{j\in N:(i,j)\in A} f_{ij}^{kl} - \sum_{j\in N:(j,i)\in A} f_{ji}^{kl} = \begin{cases} z_{kl} & si \quad i = k \\ -z_{kl} & si \quad i = l \\ 0 & \text{caso contrario} \end{cases} \qquad \forall (k,l) \in W, \forall i \in N \tag{3.7}$$

Restricciones de localización: éstas se usarán para relacionar las variables x e y con z y f. Estas restricciones son:

$$2z_{kl} \leq y_k + y_l \quad \forall (k,l) \in W \tag{3.8}$$

$$f_{ij}^{kl} + f_{ji}^{kl} \leq x_{ij} \quad \forall (i,j) \in A, \forall (k,l) \in W \tag{3.9}$$

$$\sum_{i\in N:(i,l)\in A} f_{il}^{kl} \leq y_l \quad \forall (k,l) \in W \tag{3.10}$$

$$\sum_{j\in N:(k,j)\in A} f_{kj}^{kl} \leq y_k \quad \forall (k,l) \in W \tag{3.11}$$

La restriccion (3.8) nos dice que, si se capta la demanda de k a l entonces hay que construir las estaciones k y l. La restricción (3.9) nos indica que, si se usa la conexión (i,j) o la (j,i), nótese que no se pueden usar las dos, para ir de k a l entonces hay que construir la conexión. Finalmente, las restricciones (3.10) y (3.11) nos relacionan la construcción de estación con la presencia de flujo en la red, imponiendo que si se construye la estación en el origen o destino del par puede salir o entrar flujo, y por el contrario si no se construye, no puede salir ni entrar flujo.

Restricción de elección: esta se basará en comparar el tiempo de tránsito mediante la red creada respecto al medio de transporte alternativo

$$\sum_{(i,j)\in A} t_{ij} f_{ij}^{kl} \leq u_{kl} \quad \forall (k,l) \in W \tag{3.12}$$

En ella se suman los tiempos de transporte de todas las conexiones implicadas en la ruta y se compara con el tiempo del medio de transporte alternativo.

3.2. Modelo 2

Este modelo es una simplificación del modelo 1, en el que para aligerar el coste computacional se eliminan todas las variables y restricciones que involucran el controlar el tránsito de la demanda

3.2.1. Datos y notación

Al igual que en el modelo 1, tenemos una red subyacente $G = (N, A)$ no dirigida asociada al problema, con $N = \{1, ..., n\}$ el conjunto de posibles localizaciones de estación y $A \subseteq \{(i,j) \in N \times N : i \neq j\}$ el conjunto de posibles conexiones entre las estaciones

Cada nodo i tiene asociado un coste de construcción de la estación C_i, y una población cubierta por la estación de h_i. Cada arco (i,j) tiene asociado un tiempo de viaje t_{ij}, un coste de construcción de la conexión de C_{ij} y el coste máximo de la red C_{max}.

3.2.2. Variables

Todas las variables usadas en este modelo son binarias y nos permitirán decidir qué estaciones y conexiones se construyen. Estas son $y_i = 1$ si se construye la estación i y $x_{ij} = 1$ si se construye la conexión (i, j). Nótese que al contrario que en el modelo 1, no son necesarios más conjuntos de variables.

3.2.3. Función objetivo y restricciones

El objetivo de nuestro problema será maximizar la cantidad de población cubierta por alguna estación, es decir:

$$\text{máx} \sum_{i \in N} h_i y_i$$

Las restricciones que nos encontramos son:

Restricción de presupuesto, para que la red sea viable desde el punto de vista financiero, necesitaremos una restricción de coste, ésta será la suma de los costes multiplicados por las variables si se construye estación o conexión

$$\sum_{(i,j) \in A} C_{ij} x_{ij} + \sum_{i \in N} C_i y_i \leq C_{max} \tag{3.13}$$

Restricciones de diseño de la red: Para este diseño haremos uso de las mismas restrcciones (3.3) a (3.6) del modelo 1, que le daban la estructura a la red en el modelo anterior. Este modelo, al contrario que el modelo 1, no necesita más grupos de restricciones.

3.2.4. Modificación del modelo 2

El modelo 2 busca que las estaciones cubran la mayor cantidad posible de población, sin embargo, esta simplificación del problema se puede aprovechar para obtener resultados aproximados a la solución óptima del modelo 1 en un

corto tiempo modificando los coeficientes de la función objetivo para que no se tengan en cuenta solo la población cercana a las estaciones, sino también los pasajeros potenciales. La motivación de este cambio radica en la existencia de puntos dentro de una ciudad que no tiene gran población, pero sí tiene gran interés de visita, como pueden ser recintos de espectáculos, aeropuertos o lugares turísticos alejados del núcleo urbano. Para ello propondremos la siguiente modificación:

Tenemos asociados a cada nodo i un d_i que denota la demanda de viajes a la estación i. Tomaremos un parámetro $\alpha \in [0, 1]$ que nos denotará el peso de la demanda/población a la hora de elaborar nuestra red, es decir, los nuevos coeficientes de la función objetivo serán una combinacion convexa de la población cercanas a las estaciones y sus viajeros potenciales de la forma $\alpha h_i + (1 - \alpha)d_i$. Con $\alpha = 0$ el modelo que busca maximizar la cobertura de los lugares más demandados, y para $\alpha = 1$ tenemos el modelo 2 que busca maximizar la cobertura de las estaciones en los lugares más habitados.

La modificación que hacemos a este modelo la encontramos en la función objetivo:

$$\text{máx} \sum_{i \in N} (\alpha h_i + (1 - \alpha)d_i)y_i \tag{3.14}$$

Las restricciones de este modelo seran las mismas que las del modelo 2.

3.2.5. Post Modelo

La ejecución de este modelo se realiza en un tiempo bastante rápido, por lo que es posible realizar varias redes haciendo variar el α y comprobando la captura de demanda de la red. De todas las redes obtenidas podemos tomar la que mejor valor en la cobertura por pares nos dé (Modelo 1). Gracias a que fijamos las variables x e y, la complejidad de la ejecución del modelo 1 con las

variables fijada es mucho menor, ya que se reduce notablemente el número de posibles soluciones factibles. Esto nos permite obtener las soluciones en un tiempo razonable.

Además, las redes que se obtienen son idénticas para muchos valores de α, por lo que es razonable tanto en tiempo como en utilidad probar con gran número de α, desechar las α para las que se obtiene la misma red. Obteniendo, como hemos visto antes, la cobertura por pares de cada una de las redes fijando las variables x y y que nos definen la red.

Capítulo 4

Entorno tecnológico

Para el cálculo numérico de los problemas lineales existen numerosos softwares y solvers, tanto libres como comerciales. Cada uno de ellos tiene sus ventajas y sus inconvenientes. Es por ello que decidir bien el software a elegir facilitará mucho la ejecución del problema. Los entornos más comunes dentro de la resolución de estos modelos de programación lineal entera son AMPL y Python, ambos permiten escoger gran cantidad de solvers de resolución. El entorno escogido para este problema es el AMPL, aunque para algunos procesos como la generación de datos se ha utilizado Python.

4.1. AMPL

AMPL (A Mathematical Programming Language) es un lenguaje de modelado algebraico que permite la formulación y resolución de modelos de programación matemática de gran tamaño.

Una ventaja destacada de AMPL radica en su sintaxis, la cual se asemeja significativamente a la notación matemática utilizada para describir problemas de optimización. Esto posibilita una formulación de problemas en

el ámbito de la optimización que es tanto concisa como fácil de entender. La similitud con la notación matemática facilita la expresión clara y legible de los problemas, permitiendo una definición eficiente y comprensible en el contexto de la optimización.

Otra gran ventaja de este entorno de resolución es la disponibilidad de un servidor `https://neos-server.org/neos/index.html` donde resolver el modelo, teniendo a disposición solvers comerciales como gurobi o CPLEX de forma gratuita.

Otra de las ventajas de este entorno es la claridad a la hora de incluir los datos, conjuntos, variables y restricciones. Ya que en este entorno se divide el problema en tres archivos, separando, por un lado, el modelo (xx.mod), los datos (xx.dat) y otro archivo de ejecución (xx.run) en el que se elegirá el modelo que se va a resolver, el conjunto de datos del problema y el solver que se va a usar. También permite decidir los elementos que deseas obtener, así como poder operar entre ellos o fijarlos para la ejecución de otro modelo.

Esta separación permite una flexibilidad a la hora de ejecutar el mismo modelo con distintos datos, mantener los datos y cambiar el modelo, ejecutar varios modelos en el mismo archivo.run, etc.

Primero de todo tenemos el modelo donde se introducen los conjuntos mediante set, después se introducen los datos que va a necesitar el modelo y finalmente se introducen las variables indicando que tipo de variables son, la función objetivo y las restricciones de nuestro modelo. Como vemos en el ejemplo de la Figura 4.1

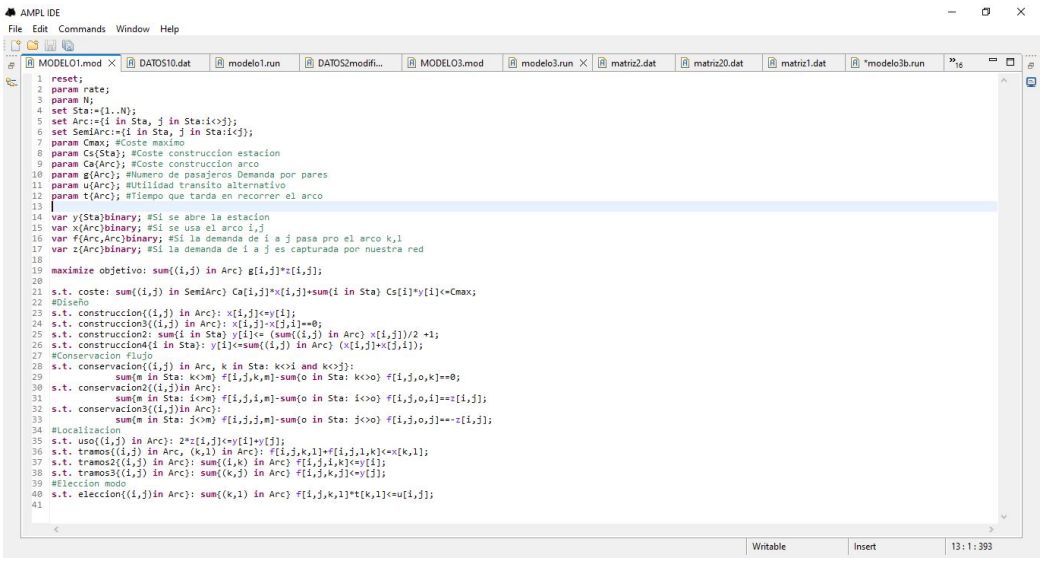


Figura 4.1: Ejemplo de archivo.mod.

En la Figura 4.2 vemos un ejemplo de archivo .dat donde se introduce el conjunto de datos teniendo en cuenta el formato requerido por el modelo, es decir matriz, lista...

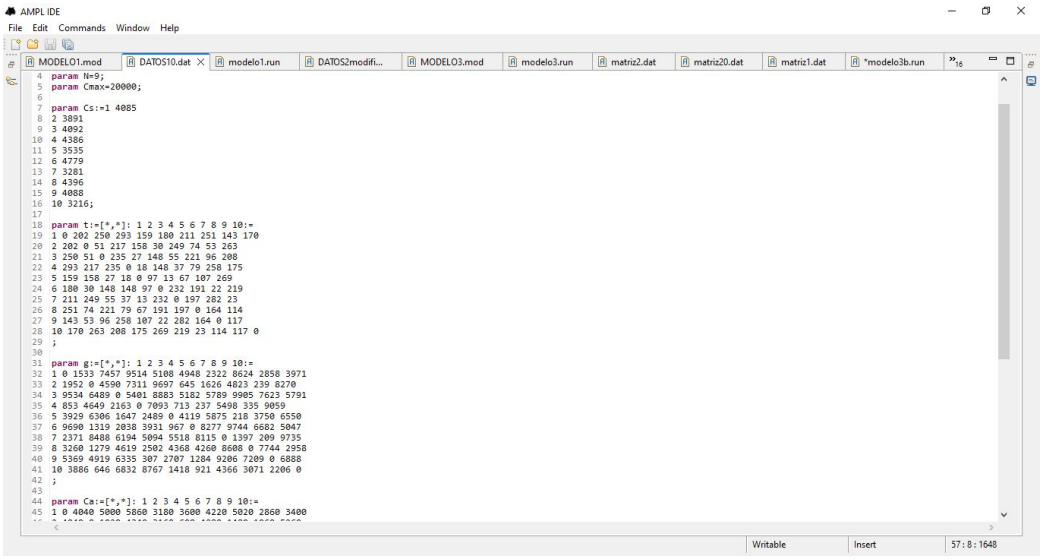


Figura 4.2: Ejemplo de archivo.dat.

Finalmente, se ejecuta el archivo .run como el ejemplo de la Figura 4.3 introduciendo el solver elegido, el modelo y conjunto de datos a resolver y los resultados que se desean obtener. Estos resultados se verán reflejados en la consola o extraídos en algún archivo externo.

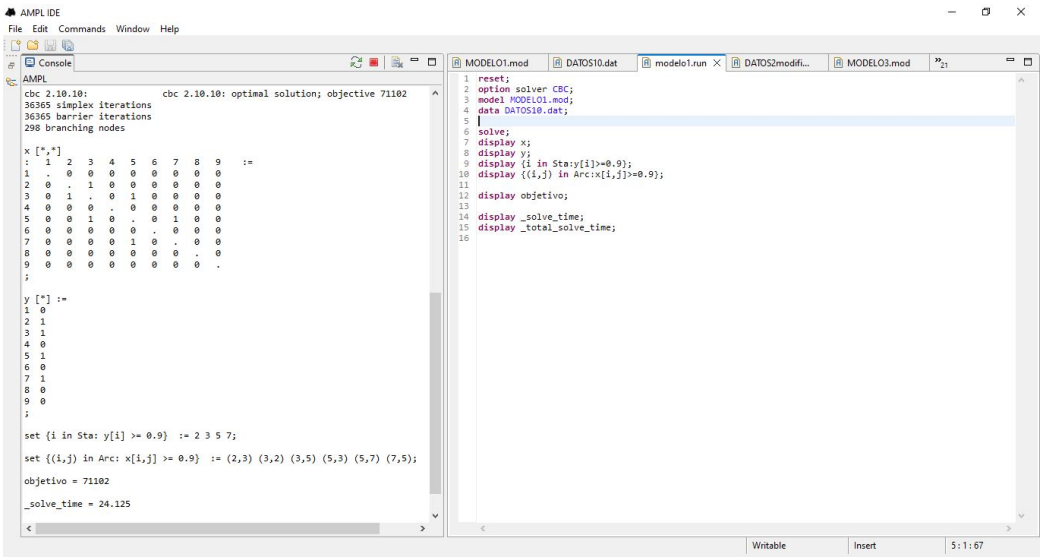


Figura 4.3: Ejemplo de resolución con AMPL y archivo.run.

4.1.1. Solver

Para la resolución de los problemas es necesario elegir el solver a utilizar. Existe una gran variedad de solvers donde cada uno trata el modelo de programación lineal entera de una manera distinta. Por ello elegir el solver adecuado permite obtener soluciones más rápidas. El solver escogido es CBC, ya que es el que mejores resultados ha dado entre los solver libres probados (HiGHS, GCG, scip, CBC).

El solver CBC (COIN-OR Branch and Cut) es un solver libre de programación lineal entera que trabaja con el solver CLP (COIN-OR Linear Programming) y la librería CGL (COIN-OR cut Generator Library). Como

su nombre indica, este solver se basa en el método Branch and Cut de resolución de los problemas lineales enteros. Para ello se hará uso del solver CLP para resolver el problema lineal y de la librería CGL para obtener los cortes del algoritmo Branch and Cut.

4.2. Python

Python es un lenguaje de programación de alto nivel y de propósito general, es decir, su utilidad no se limita a un solo tipo campo de aplicación, lo que le hace ser muy versátil a la hora de trabajar con él.

Entre sus ventajas destaca la legibilidad del código frente a otros lenguajes del mismo carácter y la licencia de código abierto que tiene para gran variedad de plataformas. Aunque es posible trabajar directamente desde la consola, nosotros haremos uso del editor Spyder a través del navegador Anaconda, que da acceso a gran número de editores. Spyder es un editor de código abierto disponible para gran cantidad de plataformas que permite el lenguaje Python, mostrándote los resultados en la misma pantalla del editor.

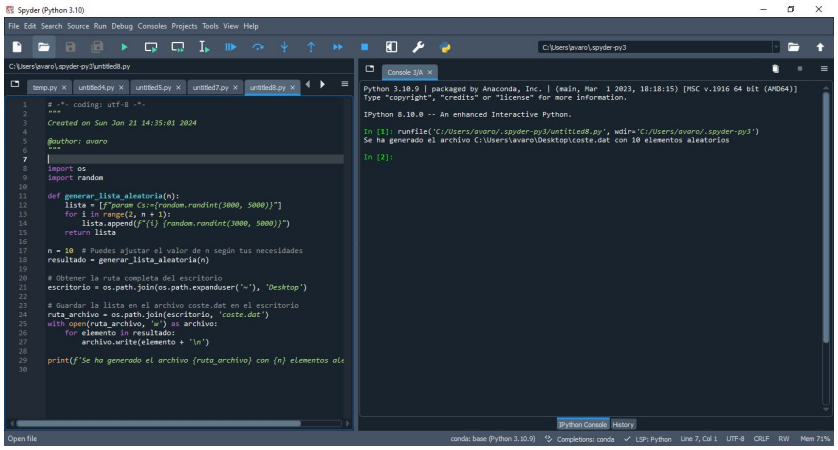


Figura 4.4: Ejemplo de ejecución de un código en Python.

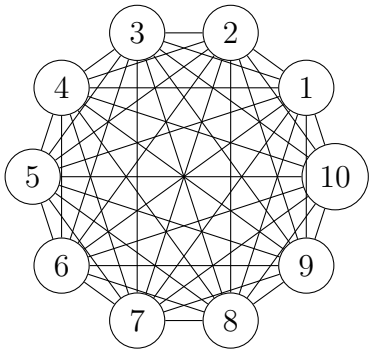
Capítulo 5

Resultados experimentales

En este capítulo se presentarán los resultados obtenidos a la hora de implementar los modelos 1 y el 2 modificado, posteriormente compararemos los resultados tanto numéricos como en tiempo de ejecución. Para ello obtendremos datos generados aleatoriamente en Python y compararemos la eficacia de los dos modelos.

Los primeros datos que utilizaremos para evaluar los modelos son los siguientes:

Nuestra red subyacente (red de posibles estaciones y posibles enlaces) es el grafo completo de 10 nodos,



con la matriz de distancias asociada (la generación de estos datos se ha realizado tomando valores aleatorios entre 1 y 1000 y haciendo la matriz simétrica con el código Python del anexo).

$$t = \begin{pmatrix} 0 & 202 & 250 & 293 & 159 & 180 & 211 & 251 & 143 & 170 \\ 202 & 0 & 51 & 217 & 158 & 30 & 249 & 74 & 53 & 263 \\ 250 & 51 & 0 & 235 & 27 & 148 & 55 & 221 & 96 & 208 \\ 293 & 217 & 235 & 0 & 18 & 148 & 37 & 79 & 258 & 175 \\ 159 & 158 & 27 & 18 & 0 & 97 & 13 & 67 & 107 & 269 \\ 180 & 30 & 148 & 148 & 97 & 0 & 232 & 191 & 22 & 219 \\ 211 & 249 & 55 & 37 & 13 & 232 & 0 & 197 & 282 & 23 \\ 251 & 74 & 221 & 79 & 67 & 191 & 197 & 0 & 164 & 114 \\ 143 & 53 & 96 & 258 & 107 & 22 & 282 & 164 & 0 & 117 \\ 170 & 263 & 208 & 175 & 269 & 219 & 23 & 114 & 117 & 0 \end{pmatrix}$$

Supondremos que el tiempo entre las conexiones por el medio alternativo es el doble que el tiempo de conexión directa por nuestra red y el coste de construcción de la conexión es 20 veces el tiempo entre las conexiones.

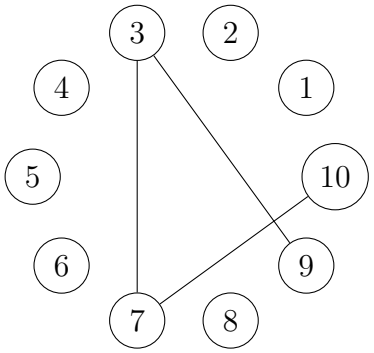
Tenemos una matriz de demandas Origen-Destino (la generación de estos datos se ha realizado tomando valores aleatorios entre 1 y 10000 con el código Python del anexo).

$$g = \begin{pmatrix} 0 & 1533 & 7457 & 9514 & 5108 & 4948 & 2322 & 8624 & 2858 & 3971 \\ 1952 & 0 & 4590 & 7311 & 9697 & 645 & 1626 & 4823 & 239 & 8270 \\ 9534 & 6489 & 0 & 5401 & 8883 & 5182 & 5789 & 9905 & 7623 & 5791 \\ 853 & 4649 & 2163 & 0 & 7093 & 713 & 237 & 5498 & 335 & 9059 \\ 3929 & 6306 & 1647 & 2489 & 0 & 4119 & 5875 & 218 & 3750 & 6550 \\ 9690 & 1319 & 2038 & 3931 & 967 & 0 & 8277 & 9744 & 6682 & 5047 \\ 2371 & 8488 & 6194 & 5094 & 5518 & 8115 & 0 & 1397 & 209 & 9735 \\ 3260 & 1279 & 4619 & 2502 & 4368 & 4260 & 8608 & 0 & 7744 & 2958 \\ 5369 & 4919 & 6335 & 307 & 2707 & 1284 & 9206 & 7209 & 0 & 6888 \\ 3886 & 646 & 6832 & 8767 & 1418 & 921 & 4366 & 3071 & 2206 & 0 \end{pmatrix}$$

Para la obtención de las h_i y d_i sumaremos por filas y por columnas la matriz de demandas obteniendo el número de habitantes y número de demandas de ir al sitio i.

Supondremos un presupuesto de 20000 u.m. y un coste de apertura de las estaciones de $(4085, 3891, 4092, 4386, 3535, 4779, 3281, 4396, 4088, 3216)$ respectivamente, generados tomando aleatoriamente valores entre 3000 y 5000 con el código Python del anexo.

Aplicando el modelo 1 obtenemos un resultado de la demanda captada por la red de 71174, además obtenemos la red óptima en un tiempo de 100 segundos.



Por otro lado, con el modelo 2 mejorado evaluando para cada $\alpha = \frac{k}{100}$ con $k = 0 \dots 100$ obtenemos 101 redes, pero gran cantidad de ellas se repiten, quedando 4 redes distintas. El tiempo de ejecutar las 101 redes es de 19 segundos y el de eliminar las repetidas 0.01 segundos

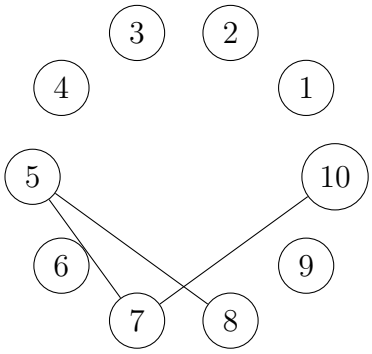


Figura 5.1: $0 \leq \alpha \leq 0.11$

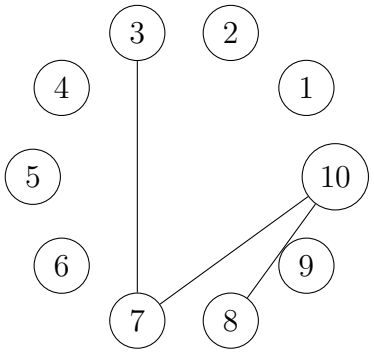


Figura 5.2: $0.12 \leq \alpha \leq 0.58$

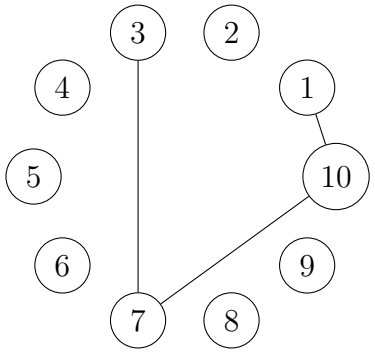


Figura 5.3: $0.59 \leq \alpha \leq 0.73$

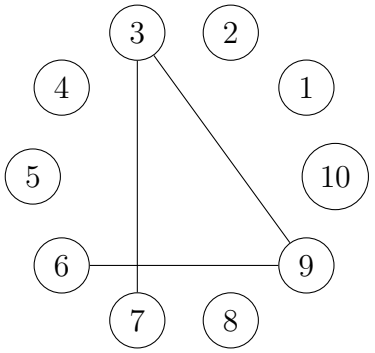


Figura 5.4: $0.74 \leq \alpha \leq 1$

Fijando las variables x e y en cada una de las redes obtenidas y evaluando el modelo 1 obtenemos unos resultados de captación de demanda de 54082 para la primera red (Figura 5.1), 69265 para la segunda (Figura 5.2), 68248 para la tercera (Figura 5.3) y 66934 para la cuarta (Figura 5.4). La ejecución del modelo 1 con las variables fijadas ha tenido una duración de 0.25 para cada una de las redes.

En la siguiente tabla se indican los resultados obtenidos, además añadiremos una columna con la bondad de estos resultados con la medida RPD[1].

Red	Demanda captada	RPD	Tiempo
Modelo 1	71174		100s
$0 \leq \alpha \leq 0.11$	54082	24.01 %	20s (cómputo total)
$0.12 \leq \alpha \leq 0.58$	69265	2.68 %	
$0.59 \leq \alpha \leq 0.73$	68248	4.11 %	
$0.74 \leq \alpha \leq 1$	66934	5.95 %	

Cuadro 5.1: Resultados obtenidos para 10 estaciones potenciales.

[1]Relative Percentage Deviation(RPD)= $\frac{\text{Demanda captada 1} - \text{Demanda captada 2}}{\text{Demanda captada 1}} \times 100\,\%$

La red con mejor cobertura de demanda se acerca mucho al valor óptimo (69265 viajeros captados, a un 2.68 % de la red óptima encontrada por el modelo 1) y en un tiempo 5 veces inferior, el cómputo de todas las redes del segundo modelo, ya que cada una de ella se obtiene en menos de 1 segundo. Sin embargo, el potencial de este modelo 2 modificado reside en problemas con una cantidad de estaciones potenciales mayor, ya que el tiempo de ejecución del modelo modificado no se incrementa tanto como los del modelo 1.

Para ver esto, tomaremos unos datos de 26 estaciones potenciales con sus respectivas demandas, tiempos y costes generados aleatoriamente con Python como en los datos anteriores y fijamos el coste máximo en 50000 u.m. Con estos datos obtenemos, con el modelo 1, una red con una demanda captada de 845606 en un tiempo de 17400 segundos= 4 horas 50 minutos, resuelto con el solver comercial Gurobi, ya que con el solver libre CBC se supera el límite de tiempo, fijado en 8 horas.

La red óptima se muestra en la Figura 5.5

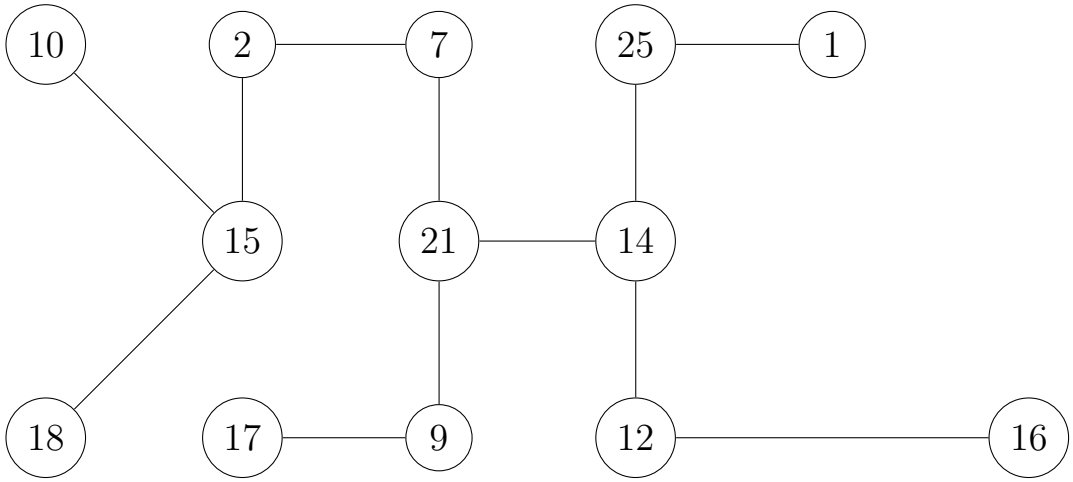


Figura 5.5: Red óptima del Modelo 1 para 26 estaciones potenciales.

Por otro lado, ejecutando el modelo 2 modificado obtenemos y eliminamos las redes repetidas en un tiempo de 45 segundos. Quedando solo 5 redes distintas.

Obteniendo la demanda captada por cada una de las redes, obtenemos unos resultados de 252824 para $0 \leq \alpha \leq 0.1$, de 818674 para $0.11 \leq \alpha \leq 0.34$, de 830351 para $0.35 \leq \alpha \leq 0.55$, de 816096 para $0.56 \leq \alpha \leq 0.79$ y de 819976 para $0.8 \leq \alpha \leq 1$ en un tiempo de computación del modelo 1 con las variables x, y fijadas de 8 segundos para cada red. Lo que nos da un tiempo total de computación del modelo de 45 segundos + 40 segundos= 1 minuto 25 segundos.

En la siguiente tabla podemos ver un resumen de los resultados obtenidos, además de la medida de bondad RPD.

Red	Demanda captada	RPD	Tiempo
Modelo 1	845606		4h 50min
$0 \leq \alpha \leq 0.10$	252824	70.10 %	
$0.11 \leq \alpha \leq 0.34$	818674	3.18 %	1 min 25 seg
$0.35 \leq \alpha \leq 0.55$	830351	1.80 %	(cómputo total)
$0.56 \leq \alpha \leq 0.79$	816096	3.48 %	
$0.8 \leq \alpha \leq 1$	819776	3.05 %	

Cuadro 5.2: Resultados obtenidos para 26 estaciones potenciales.

Tomamos la mejor red del modelo 2 modificado, es decir $0.35 \leq \alpha \leq 0.55$ y nos queda la red siguiente.

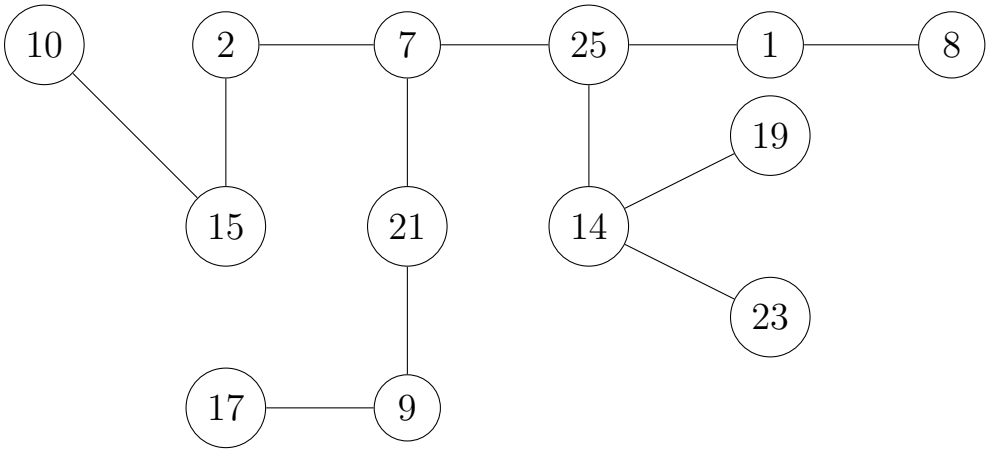


Figura 5.6: Mejor red del Modelo 2 modificado para 26 estaciones potenciales.

Comparando las dos redes resultantes, como observamos en la Figura 5.7, vemos que el modelo 1 tiene 4 conexiones no compartidas (azul), el modelo 2 modificado tiene otras 4 conexiones no compartidas (rojo) y las restantes 8 conexiones son compartidas por las dos redes (negro), por lo que vemos que el resultado es bastante similar. Además, el total de viajes captados con esta red está a tan solo un 1.80 % de la red óptima.

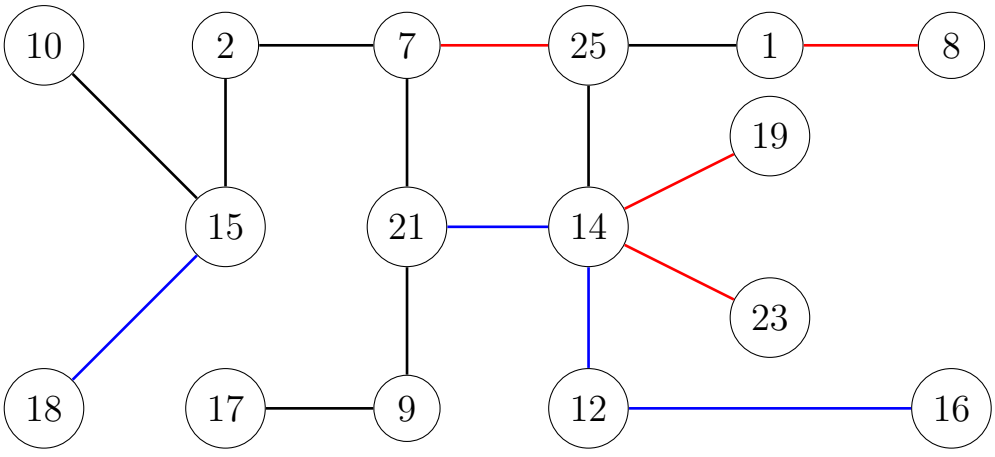


Figura 5.7: Comparación de las redes de los dos modelos.

Nos podríamos preguntar que ocurriría si en vez de calcular el modelo cada $\alpha = \frac{k}{100}$ lo calculamos en intervalos más pequeños, por ejemplo $\alpha = \frac{k}{1000}$, probamos con los datos anteriores y de las 1001 redes obtenidas, nos quedamos con las mismas 5, pero ahora con los intervalos del α más ajustados ([0,0.106], [0.107,0.348], [0.349,0.550], [0.551,0.793], [0.794,1]), sin embargo, este aumento de la cantidad de redes generadas supone un coste computacional muy alto (450 s), por lo que perdemos la principal ventaja de este modelo, su rapidez sin mejorar la mejor red encontrada con los 101 α distintos.

Por otro lado, podemos ver como se comportan ambos modelos al modificar el presupuesto de construcción de la red, tanto de bondad de la solución del modelo 2 modificado, como su utilidad en conforme del tiempo.

En la siguiente tabla veremos el resultado óptimo del modelo 1, el mejor resultado del modelo 2 modificado y la bondad en términos de RPD del resultado del modelo 2 modificado respecto a la del modelo 1, entre paréntesis tendremos los tiempos totales de ejecución de cada uno de los modelos.

Presupuesto ($C_{\text{máx}}$)	Modelo 1	Modelo 2 modificado	RPD
40000	540088 (20720s)	532773 (100+15s)	1.35 %
45000	686886 (18276s)	677022 (80+28s)	1.43 %
50000	845606 (17400)	830351 (45+40s)	1.80 %
55000	1019250 (17442s)	961762 (40+40s)	5.64 %
60000	1182560 (16329s)	805453 (30+48s)	31.88 %
65000	1409630 (8543s)	1051520 (27+75s)	25.46 %
100000	2860240 (2894s)	1916200 (17+90s)	33.00 %
10^8	3319780 (63s)		

Cuadro 5.3: Resultados obtenidos para 26 estaciones potenciales según presupuesto (tiempos con gurobi el modelo 1 y con CBC el modelo 2).

Observamos que en el modelo 1 va mejorando el tiempo de computación del modelo conforme el presupuesto es más alto (menos restrictivo), sin embargo, en el modelo 2 modificado observamos que el tiempo de generación de los modelos mejora al tener una restricción de presupuesto más relajada. Por otro lado, la segunda parte del modelo 2 modificado, el cálculo de la utilidad de la red, conforme el presupuesto aumenta, lo hace también el tiempo de computación del modelo 1 con las variables fijadas. Esto se debe a que la red con mayor presupuesto es más grande, por lo que el número de soluciones factibles es mayor.

Para el presupuesto "ilimitado"($C_{max} = 10^8$), el modelo 2 modificado genera un único modelo compuesto por todas las x e y igual a 1, por lo que la ejecución del modelo 1 con las variables fijadas, el valor que se obtiene será el mismo que en el modelo 1, ya que las variables f y z no tienen restricción con la x e y igual a 1.

Fijándonos en la bondad de la aproximación, observamos que mientras menor sea el presupuesto, mejor aproximación obtenemos.

5.1. Análisis por nodos

Se ha observado que el modelo de cobertura por nodos obtiene muy buenos resultados a la hora de conseguir una buena cobertura por pares. Es obvio que el objetivo principal a la hora de diseñar la red de transporte es maximizar el número de viajes cubiertos, pero cabría la pregunta: ¿Cómo de bueno será el modelo que maximiza por pares a la hora de la cobertura por nodos? Aunque a la hora de diseñar no tendrá interés, sería interesante que además de cubrir los máximos viajes posibles, también cubriera a la mayor cantidad de habitantes posibles ya que los viajes habituales de éstos pueden cambiar

con el tiempo. Por lo que haremos un análisis de las redes resultantes según la cobertura por nodos.

Si tratamos ahora de comparar el modelo 1 con el modelo 2 (modelo 2 modificado con $\alpha = 1$) desde el punto de vista del volumen de población cubierta por las estaciones, utilizando los mismos datos que en las estimaciones anteriores obtenemos los siguientes resultados.

Resultados para la base de datos de 10 estaciones potenciales		
Red	Población captada por nodos	Tiempo de computación
Modelo 1	188055	100 s
Modelo 2	203637	0.25 s

Cuadro 5.4: Resultados de cobertura por nodos con una base de datos de 10 estaciones potenciales (RDP del modelo 1 respecto al modelo 2 = 7.65 %).

Veamos ahora qué ocurre si la cantidad de estaciones potenciales es mucho mayor.

Resultados para la base de datos de 26 estaciones potenciales		
Red	Población captada por nodos	Tiempo de computación
Modelo 1	1642810	4h 50min
Modelo 2	1724430	0.45 s

Cuadro 5.5: Resultados de cobertura por nodos con una base de datos de 26 estaciones potenciales (RDP del modelo 1 respecto al modelo 2 = 4.73 %).

Como hemos visto antes en el caso de la cobertura por pares, veamos qué sensibilidad al presupuesto tienen estos modelos respecto a la cobertura por nodos. Además, veremos la bondad de utilizar el modelo 1 respecto al

modelo 2. Los tiempos de computación del modelo 1 coinciden con los del otro análisis de la sensibilidad al presupuesto. Los del modelo 2, al no tener que generar muchas redes ni calcular el modelo 1 con variables fijadas, se reducen considerablemente a tiempos menores al segundo de computación.

Sensibilidad al presupuesto			
Presupuesto	Modelo 1	Modelo 2	RDP del modelo 1 respecto al 2
40000	1165186	1381050	15.63 %
45000	1546504	1551740	0.33 %
50000	1642810	1724430	4.73 %
55000	1804369	1870350	3.52 %
60000	1936405	2024100	4.33 %
65000	2151319	2199140	2.17 %
100000	3078613	3112860	1.10 %
10^8	3319780	3319780	

Cuadro 5.6: Resultados según el presupuesto

Como vemos con ambos modelos obtenemos una cobertura de población similar, lo que nos hace pensar que un buen modelo para la cobertura por pares, debe ser un buen modelo para la cobertura por nodos. Sin embargo, el primer modelo tiene un coste computacional muy alto para llegar a la solución óptima, por lo que no sería una opción viable para el cálculo de la cobertura por nodos como ocurre en el caso del modelo 2 modificado para el cálculo de cobertura por pares.

Capítulo 6

Conclusiones

El diseño de sistemas de tránsito rápido es un proceso complejo y costoso que involucra a diversos actores. La formulación de modelos matemáticos y algoritmos para abordar estos problemas es un desafío debido a la falta de definición clara de los objetivos y restricciones en los problemas reales, en contraste con los modelos de investigación operativa, donde las restricciones y objetivos deben ser lo más precisos posibles.

Para abordar estos problemas, se recurre a distintos modelos que se intentan asemejar lo más posible a la realidad, cada uno de ellos se aproxima más en algún aspecto en concreto. El objetivo de este Trabajo de Fin de Máster es proponer y evaluar dos modelos de programación lineal entera, tratando de optimizar la utilidad de la red, maximizando los viajeros beneficiados, desde dos perspectivas distintas, por un lado, mejorando directamente los viajeros potenciales y, por otro lado, ubicando las estaciones para obtener la mayor población que reside cerca de las estaciones y la demanda de los puntos de interés dentro de la ciudad cercana a la estación.

El primer modelo, al buscar directamente la función objetivo, obtiene mejores resultados que el segundo modelo. Pese a esto, el segundo modelo es

una simplificación computacional del primero, que como hemos visto, obtiene buenos resultados en cuanto el objetivo real (demanda captada por pares) en menor tiempo para problemas más grandes.

Por otro lado vemos que desde la perspectiva de maximizar la cobertura por nodos, el modelo 2 aporta la solución óptima, mientras que los resultados del modelo 1 son buenos. Esto nos hace prever que, si buscamos la optimización por pares como se busca en el modelo 1, obtendremos paralelamente una buena solución del problema de maximizar la población cubierta por las estaciones.

Puesto que estos dos modelos matemáticos son aproximaciones a la realidad, nos dan una información valiosa para la hora de tomar decisiones políticas de transporte y urbanismo. Pero estos deben ser una ayuda a la hora de planificar, por lo que obtener buenos resultados, no necesariamente óptimos, en poco tiempo, en algunos casos es mejor que obtener el resultado óptimo en un tiempo de computación excesivamente alto.

6.1. Dificultades encontradas

La literatura que existe sobre cómo obtener la red que busca maximizar la cobertura por nodos es escasa. Esto ha supuesto una dificultad en la elaboración de este trabajo. La mayor parte de la bibliografía sobre modelos de cobertura por nodos, se centra en la localización de estaciones como un paso previo a obtener las estaciones potenciales y posteriormente las posibles conexiones entre ellas.

Otro contratiempo encontrado es la complejidad computacional de los modelos, en su fase experimental, ya que algunas de las ejecuciones requerían de mucho tiempo hasta obtener los resultados, con el añadido que la licencia

del solver comercial Gurobi estaba limitada, por lo que cada una de las ejecuciones había que adaptarlas al server NEOS.

Capítulo 7

Anexos

7.1. Códigos empleados

7.1.1. Generación de datos en Python

- Generación de las t,u y Ca

```
# -*- coding: utf-8 -*-
"""

@author: avaro
"""

import os
import numpy as np

def generar_matriz_enteros(tamaño):
    matriz = np.random.randint(1, 301, size=(tamaño, tamaño))
    matriz = np.triu(matriz, k=1)
    matriz = matriz + matriz.T
    return matriz

def guardar_matriz_en_dat(matriz, ruta_archivo):
    with open(ruta_archivo, 'w') as file:
        # Agregar números del 1 al tamaño de la matriz en la primera línea
        file.write("param t:=[*,*]:")
        for i in range(1, len(matriz) + 1):
            file.write(f' {i}')
        file.write(":=\n")

        # Agregar la matriz con sus valores
        for i, fila in enumerate(matriz):
            file.write(f'{str(i+1)}')
```

```
            for valor in fila:
                file.write(f' {valor}')
            file.write('\n')
        file.write(";\n")

def multiplicar_matriz_por_dos(matriz):
    return matriz * 2

def multiplicar_matriz_por_veinte(matriz):
    return matriz * 20

if __name__ == "__main__":
    tamaño_matriz = 10
    nombre_archivo = "matriz.dat"
    nombre_archivo2 = "matriz2.dat"
    nombre_archivo20 = "matriz20.dat"

    # Generar matriz de enteros entre 1 y 1000
    matriz_resultante = generar_matriz_enteros(tamaño_matriz)

    # Establecer la diagonal a cero
    np.fill_diagonal(matriz_resultante, 0)

    # Obtener la ruta del escritorio
    escritorio = os.path.join(os.path.expanduser('~'), 'Desktop')

    # Unir la ruta del escritorio con el nombre del archivo
    ruta_completa = os.path.join(escritorio, nombre_archivo)
    ruta_completa2 = os.path.join(escritorio, nombre_archivo2)
    ruta_completa20 = os.path.join(escritorio, nombre_archivo20)

    # Guardar en formato .dat para AMPL en el escritorio
    guardar_matriz_en_dat(matriz_resultante, ruta_completa)

    # Multiplicar la matriz por dos y guardar en un segundo archivo
    matriz_multiplicada_por_dos = multiplicar_matriz_por_dos(matriz_resultante)
    guardar_matriz_en_dat(matriz_multiplicada_por_dos, ruta_completa2)

    # Multiplicar la matriz por veinte y guardar en un tercer archivo
    matriz_multiplicada_por_veinte = multiplicar_matriz_por_veinte(matriz_resultante)
    guardar_matriz_en_dat(matriz_multiplicada_por_veinte, ruta_completa20)
```

- Generación de g

```
import os
import numpy as np

def generar_matriz_enteros(tamaño):
    matriz = np.random.randint(1, 10000, size=(tamaño, tamaño))
    #matriz = np.triu(matriz, k=1)
    matriz = matriz
    return matriz

def guardar_matriz_en_dat(matriz, ruta_archivo):
```

```
    with open(ruta_archivo, 'w') as file:
        # Agregar números del 1 al tamaño de la matriz en la primera línea
        file.write("param t:=[*,*]:")
        for i in range(1, len(matriz) + 1):
            file.write(f' {i}')
        file.write(":=\n")

        # Agregar la matriz con sus valores
        for i, fila in enumerate(matriz):
            file.write(f'{str(i+1)}')
            for valor in fila:
                file.write(f' {valor}')
            file.write('\n')
        file.write(";\n")

if __name__ == "__main__":
    tamaño_matriz = 10
    nombre_archivo = "matriz1.dat"

    # Generar matriz de enteros entre 1 y 1000
    matriz_resultante = generar_matriz_enteros(tamaño_matriz)

    # Establecer la diagonal a cero
    np.fill_diagonal(matriz_resultante, 0)

    # Obtener la ruta del escritorio
    escritorio = os.path.join(os.path.expanduser('~'), 'Desktop')

    # Unir la ruta del escritorio con el nombre del archivo
    ruta_completa = os.path.join(escritorio, nombre_archivo)

    # Guardar en formato .dat para AMPL en el escritorio
    guardar_matriz_en_dat(matriz_resultante, ruta_completa)
```

- Generación de las Cs

```
import os
import random

def generar_lista_aleatoria(n):
    lista = [f"param Cs:={random.randint(3000, 5000)}"]
    for i in range(2, n + 1):
        lista.append(f"{i} {random.randint(3000, 5000)}")
    return lista

n = 10  # Puedes ajustar el valor de n según tus necesidades
resultado = generar_lista_aleatoria(n)

# Obtener la ruta completa del escritorio
escritorio = os.path.join(os.path.expanduser('~'), 'Desktop')

# Guardar la lista en el archivo coste.dat en el escritorio
ruta_archivo = os.path.join(escritorio, 'coste.dat')
```

```
with open(ruta_archivo, 'w') as archivo:
    for elemento in resultado:
        archivo.write(elemento + '\n')
```

7.2. Modelos AMPL

7.2.1. Modelo 1

- Modelo1.mod

```
reset;
param rate;
param N;
set Sta:={1..N};
set Arc:={i in Sta, j in Sta:i<>j};
set SemiArc:={i in Sta, j in Sta:i<j};
param C_{max}; #Coste maximo
param Cs{Sta}; #Coste construccion estacion
param Ca{Arc}; #Coste construccion arco
param g{Arc}; #Numero de pasajeros Demanda por pares
param u{Arc}; #Utilidad transito alternativo
param t{Arc}; #Tiempo que tarda en recorrer el arco
param h{i in Sta}=sum{(i,j) in Arc} g[i,j];

var y{Sta}binary; #Si se abre la estacion
var x{Arc}binary; #Si se usa el arco i,j
var f{Arc,Arc}binary; #Si la demanda de i a j pasa por el arco k,l
var z{Arc}binary; #Si la demanda de i a j es capturada por nuestra red

maximize objetivo: sum{(i,j) in Arc} g[i,j]*z[i,j];

s.t. coste: sum{(i,j) in SemiArc} Ca[i,j]*x[i,j]+sum{i in Sta} Cs[i]*y[i]<=C_{max};
#Diseño
s.t. construccion{(i,j) in Arc}: x[i,j]<=y[i];
s.t. construccion3{(i,j) in Arc}: x[i,j]-x[j,i]==0;
s.t. construccion2: sum{i in Sta} y[i]<= (sum{(i,j) in Arc} x[i,j])/2 +1;
s.t. construccion4{i in Sta}: y[i]<=sum{(i,j) in Arc} (x[i,j]+x[j,i]);
#Conservacion flujo
s.t. conservacion{(i,j) in Arc, k in Sta: k<>i and k<>j}:
sum{m in Sta: k<>m} f[i,j,k,m]-sum{o in Sta: k<>o} f[i,j,o,k]==0;
s.t. conservacion2{(i,j)in Arc}:
sum{m in Sta: i<>m} f[i,j,i,m]-sum{o in Sta: i<>o} f[i,j,o,i]==z[i,j];
s.t. conservacion3{(i,j)in Arc}:
sum{m in Sta: j<>m} f[i,j,j,m]-sum{o in Sta: j<>o} f[i,j,o,j]==-z[i,j];
#Localizacion
s.t. uso{(i,j) in Arc}: 2*z[i,j]<=y[i]+y[j];
s.t. tramos{(i,j) in Arc, (k,l) in Arc}: f[i,j,k,l]+f[i,j,l,k]<=x[k,l];
s.t. tramos2{(i,j) in Arc}: sum{(i,k) in Arc} f[i,j,i,k]<=y[i];
s.t. tramos3{(i,j) in Arc}: sum{(k,j) in Arc} f[i,j,k,j]<=y[j];
```

```
#Eleccion modo
s.t. eleccion{(i,j)in Arc}: sum{(k,l) in Arc} f[i,j,k,l]*t[k,l]<=u[i,j];
```

- modelo1.run

```
reset;
option solver CBC;
model MODELO1.mod;
data DATOS26.dat;

solve;
display C_{max};
display x;
display y;
display {i in Sta:y[i]>=0.9};
display {(i,j) in Arc:x[i,j]>=0.9};

display objetivo;
display sum{i in Sta} h[i]*y[i];
display _solve_time;
display _total_solve_time;
```

7.2.2. Modelo 2 modificado

- Modelo3.mod (Generación de las redes)

```
reset;
param N;

param rate;
set Sta:={1..N};
set Arc:={i in Sta, j in Sta:i<>j};
set SemiArc:={i in Sta, j in Sta:i<j};
param C_{max}; #Coste maximo
param Cs{Sta}; #Coste construccion estacion
param Ca{Arc}; #Coste construccion arco
param g{Arc}; #Numero de pasajeros Demanda por pares
param u{Arc}; #Utilidad transito alternativo
param t{Arc}; #Tiempo que tarda en recorrer el arco

param h{i in Sta}=sum{(i,j) in Arc} g[i,j];
param d{j in Sta}=sum{(i,j) in Arc} g[i,j];

var y{Sta}binary; #Si se abre la estacion
var x{Arc}binary; #Si se usa el arco i,j


maximize objetivo: sum{i in Sta} (rate*h[i]+(1-rate)*d[i])*y[i];

s.t. coste: sum{(i,j) in SemiArc} Ca[i,j]*x[i,j]+sum{i in Sta} Cs[i]*y[i]<=C_{max};
#Diseño
```

```
s.t. construccion{(i,j) in Arc}: x[i,j]<=y[i];
s.t. construccion3{(i,j) in SemiArc}: x[i,j]-x[j,i]<=0;
s.t. construccion3b{(i,j) in SemiArc}: x[i,j]-x[j,i]>=0;

s.t. construccion6{j in Sta}: y[j]<=sum {(i,j) in Arc} (x[i,j]+x[j,i]);
s.t. construccion2: sum{i in Sta} y[i]<= (sum{(i,j) in Arc} x[i,j])/2 +1;
```

- Modelo3.run (Obtención de las redes)

```
reset;
option solver CBC;
model MODELO3.mod;
data DATOS26.dat;
let rate:=0;
for {1..101}{
solve;
display rate>>resultado26.txt;
display rate;
display objetivo;
display {i in Sta:y[i]>=0.8}>>resultado26.txt;
display {(i,j) in Arc:x[i,j]>=0.8}>>resultado26.txt;
let rate:=rate+0.01;}

display _solve_time;
display _total_solve_time;
```

- Modelo1modi.mod (Modelo1 con las variables fijadas)

```
var f{Arc,Arc}binary; #Si la demanda de i a j pasa pro el arco k,l
var z{Arc}binary; #Si la demanda de i a j es capturada por nuestra red

maximize objetivo2: sum{(i,j) in Arc} g[i,j]*z[i,j];


#Conservacion flujo
s.t. conservacionm{(i,j) in Arc, k in Sta: k<>i and k<>j}:
sum{m in Sta: k<>m} f[i,j,k,m]-sum{o in Sta: k<>o} f[i,j,o,k]==0;
s.t. conservacion2m{(i,j)in Arc}: sum{m in Sta: i<>m} f[i,j,i,m]-sum{o in Sta: i<>o} f[i,j,o,i]==z[i,j];
s.t. conservacion3m{(i,j)in Arc}: sum{m in Sta: j<>m} f[i,j,j,m]-sum{o in Sta: j<>o} f[i,j,o,j]==-z[i,j];
#Localizacion
s.t. usom{(i,j) in Arc}: 2*z[i,j]<=y[i]+y[j];
s.t. tramosm{(i,j) in Arc, (k,l) in Arc}: f[i,j,k,l]+f[i,j,l,k]<=x[k,l];
s.t. tramos2m{(i,j) in Arc}: sum{(i,k) in Arc} f[i,j,i,k]<=y[i];
s.t. tramos3m{(i,j) in Arc}: sum{(k,j) in Arc} f[i,j,k,j]<=y[j];
#Eleccion modo
s.t. eleccionm{(i,j)in Arc}: sum{(k,l) in Arc} f[i,j,k,l]*t[k,l]<=u[i,j];
```

- Modelo3b.run (Ejecución del modelo1 con las variables fijadas)

```
reset;
option solver CBC;
```

```
model MODELO3.mod;

data DATOS26.dat;
let rate:=0;


solve;
display rate;

display objetivo;
display {i in Sta:y[i]>=0.8};
display {(i,j) in Arc:x[i,j]>=0.8 and i<j};
fix y;
fix x;
model MODELO1modi.mod;
objective objetivo2;
solve;
display rate;
display objetivo2;

unfix x;
unfix y;

display _solve_time;
display _total_solve_time;
```

Bibliografía

[Bruno et al. (1998)] Bruno, G., Ghiani, G., & Improta, G. (1998). A multi-modal approach to the location of a rapid transit line. European Journal of Operational Research, 104(2), 321-332.

[Dufourd et al. (1996)] Dufourd, H., Gendreau, M., & Laporte, G. (1996). Locating a transit line using tabu search. Location Science, 4(1-2), 1-19.

[Gendreau et al. (1995)] Gendreau, M., Laporte, G., & Mesa, J. A. (1995). Locating rapid transit lines. Journal of Advanced Transportation, 29(2), 145-162.

[Latora y Marchiori (2001)] Latora, V., & Marchiori, M. (2001). Efficient behavior of small-world networks. Physical review letters, 87(19), 198701.

[Laporte et al. (1997)] Laporte, G., Mesa, J. A., & Ortega, F. A. (1997). Assessing the efficiency of rapid transit configurations. Top, 5(1), 95-104.

[Laporte et al. (2002)] Laporte, G., Mesa, J. A., & Ortega, F. A. (2002). Locating stations on rapid transit lines. Computers & Operations Research, 29(6), 741-759.

[Laporte et al. (2005)] Laporte, G., Mesa, J. A., Ortega, F. A., & Sevillano, I. (2005). Maximizing trip coverage in the location of a single rapid transit alignment. Annals of Operations Research, 136, 49-63.

[Laporte et al. (2007)] Laporte, G., Marín, Á., Mesa, J. A., & Ortega, F. A. (2007). An integrated methodology for the rapid transit network design problem. In Algorithmic Methods for Railway Optimization: International Dagstuhl Workshop, Dagstuhl Castle, Germany, June 20-25, 2004, 4th International Workshop, ATMOS 2004, Bergen, Norway, September 16-17, 2004, Revised Selected Papers (pp. 187-199). Springer Berlin Heidelberg.

[Laporte et al. (2010)] Laporte, G., Mesa, J. A., & Perea, F. (2010). A game theoretic framework for the robust railway transit network design problem. Transportation Research Part B: Methodological, 44(4), 447-459.

[Laporte et al. (2011a)] Laporte, G., Marin, A., Mesa, J. A., & Perea, F. (2011). Designing robust rapid transit networks with alternative routes. Journal of advanced transportation, 45(1), 54-65.

[Laporte et al. (2011b)] Laporte, G., Mesa, J. A., Ortega, F. A., & Perea, F. (2011). Planning rapid transit networks. Socio-Economic Planning Sciences, 45(3), 95-104.

[Laporte et al. (2015)] Laporte, G., & Mesa, J. A. (2015). The design of rapid transit networks. Location science, 581-594.

[Marin y Garcia Ródenas (2009)] Marín, Á., & García-Ródenas, R. (2009). Location of infrastructure in urban railway networks. Computers & Operations Research, 36(5), 1461-1477.

[Vuchic (2005)] Vuchic V.R. (2005) Urban transit operations, planning and economics. Hoboken, New Jersey, Wiley