



Metaheurísticas y Optimización Sobre Redes

Obligatorio 2017

Germán Faller & Octavio Pérez Kempner

Docente: Dr. Ing. Claudio Risso

Instituto de Computación

Facultad de Ingeniería

Índice

1. Introducción	2
2. Parte I - Primera aproximación	3
2.1. Minimización por costos	3
2.2. Minimización por delay	4
2.3. Caminos de costo mínimo	4
2.4. Comparación de resultados	5
3. Parte II - Reutilización de tramos	5
3.1. Formulación MIP	5
3.2. Resultados	6
4. Parte III - Algunas variantes	7
5. Parte IV - Abordaje por Metaheurísticas	8
5.1. Introducción	8
5.2. Inicialización	8
5.3. Selección	9
5.4. Cruzamiento y mutaciones	9
6. Referencias consultadas	10
7. Anexo I: Pruebas realizadas	11
7.1. Cálculo de cotas y factibilidad de la población inicial	11
8. Anexo II: Instrutivo de las implementaciones realizadas	12
8.1. Introducción	12
8.2. Parte I	12
8.3. Parte II	12
8.4. Parte III	12

1. Introducción

El obligatorio a resolver consiste en abordar el problema de diseñar una red de Trenes Ligeros (Light Rail Transit o LRT) para Montevideo.

Considerando como característico de nuestra capital el tener una población que tiende a vivir lejos de su centro de estudio y/o trabajo, se supondrá que en la ciudad existen tres puntos de concentración: Tres Cruces, Palacio Municipal y la Plaza Independencia que serán los destinos de los pasajeros en la mañana y su punto de partida en la tarde. Se asumirá que se contará con la infraestructura adecuada para que al llegar a cualquiera de estos puntos la movilidad urbana entre ellos será rápida.

Por otra parte identificamos cuatro agregadores zonales, las terminales: Cerro, Colón, Carrasco y Pocitos. Se tendrá como objetivo de la red definir los caminos desde cualquier agregador zonal a cualquier punto de concentración.

A continuación se muestran las terminales, los puntos de concentración y los tramos posibles a seleccionar para construir la red LRT.



En las siguientes secciones se presentarán los distintos desafíos del obligatorio y conjuntamente con las soluciones propuestas.

2. Parte I - Primera aproximación

2.1. Minimización por costos

En una primera aproximación el objetivo fue obtener el trazado de costo mínimo para conectar las terminales con los puntos de concentración planteando el problema como uno de programación lineal (LP) y considerando las siguientes restricciones:

- Que existan dos líneas (caminos) para conectar tanto Cerro, Colón y Carrasco con los nodos del centro y tres líneas para conectar Pocitos con los nodos del centro.
- Que ninguno de los tramos del plano fuera utilizado por más de una línea (las estaciones sí pueden ser parte de más de una línea).

Formulación LP

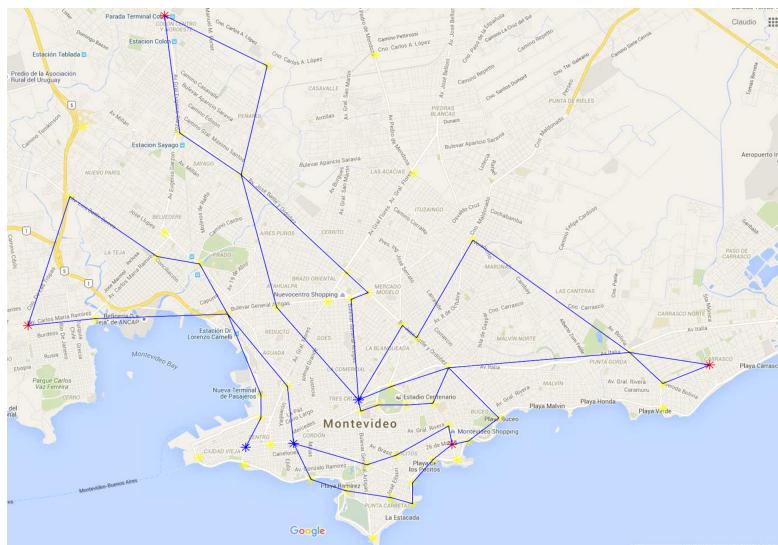
$$\min \sum_{(i,j) \in E} c_{ij} \cdot x_{ij}$$

s.t.

$$\sum_{j \in S(i)} x_{ij} - \sum_{j \in E(i)} x_{ji} = \begin{cases} 0 & \text{si } i \text{ es un nodo interno del camino} \\ f_i & (\text{flujo saliente de } i), \text{ si } i \text{ es una terminal} \end{cases}$$

$$0 \leq x_{ij} \leq 1$$

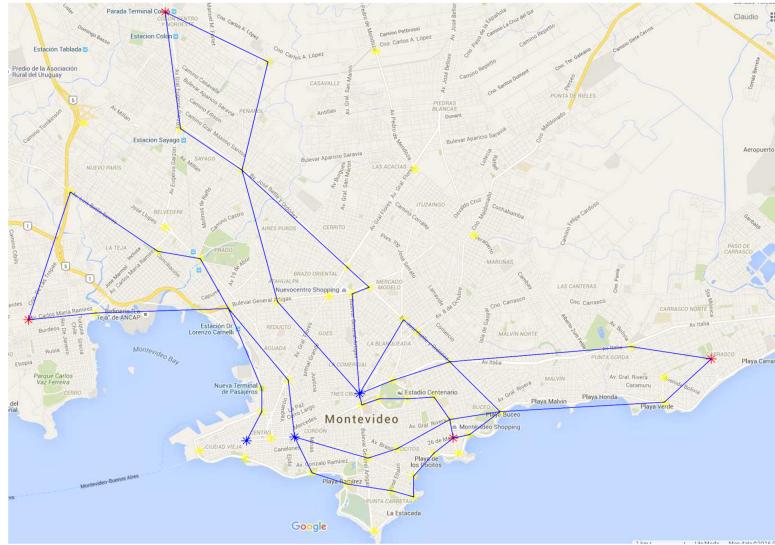
Considerando entonces el costo total mínimo, se obtuvo que este era de 18076, a continuación se muestra la solución obtenida:



2.2. Minimización por delay

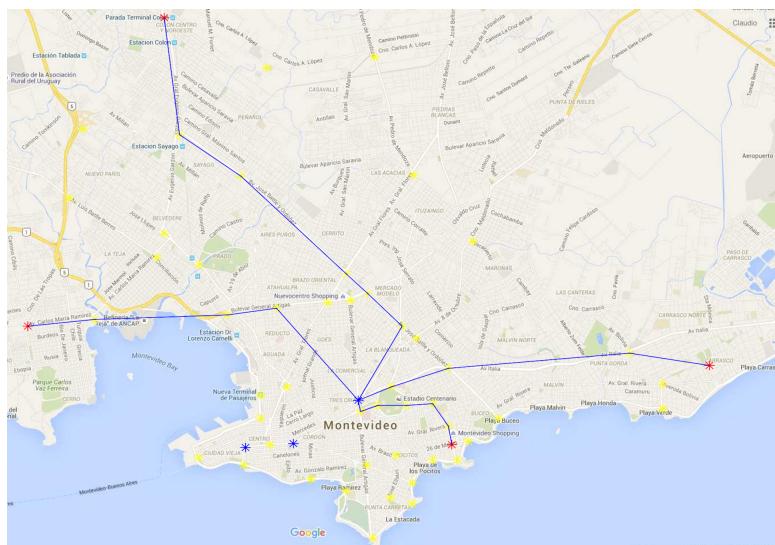
Una variante del problema anterior a resolver es la minimización por delay en lugar de costos que resulta análoga a la anterior (considerando la columna de delay en lugar a la de costos).

En este caso el óptimo hallado se corresponde a un delay total de 8512, a continuación se muestra la solución obtenida:



2.3. Caminos de costo mínimo

Como última variante en esta primera parte se calcularon los caminos de costo mínimo, el total fue de 7257, la solución obtenida se muestra a continuación:



2.4. Comparación de resultados

La siguiente tabla consolida las distintas propuestas considerando para cada una el mínimo total y sus respectivos valores en costo/delay.

Cuadro 1: Resultados

Objetivo	Costo Total	Delay Total
Total por costos	118076	8700
Total por delay	182572	8512
Único camino por costos	7257	3513
Único camino por delay	7850	3229

Cabe notar que en cualquier caso la complejidad del problema es polinomial dado que cuando tenemos múltiples fuentes y sumideros estos problemas se pueden resolver con el algoritmo de Ford-Fulkerson mientras que en el caso de tener caminos únicos alcanza con el algoritmo de Dijkstra que es también de orden polinomial.

3. Parte II - Reutilización de tramos

Bajo las mismas condiciones que la parte anterior se busca ahora la reutilización de tramos entre líneas de distinto origen. Esta nueva versión es una relajación del problema anterior dado que soluciones del problema anterior además son soluciones en esta nueva versión. Naturalmente, dado que el objetivo continúa siendo la minimización de costos, al poder reutilizar tramos entre líneas de distinto origen, el mínimo encontrado conecta los agregadores y algunos de ellos al centro, por lo que el delay se ve sensiblemente afectado respecto a la solución de la versión anterior.

3.1. Formulación MIP

$$\min \sum_{(i,j) \in E} c_{ij} \cdot u_{ij}$$

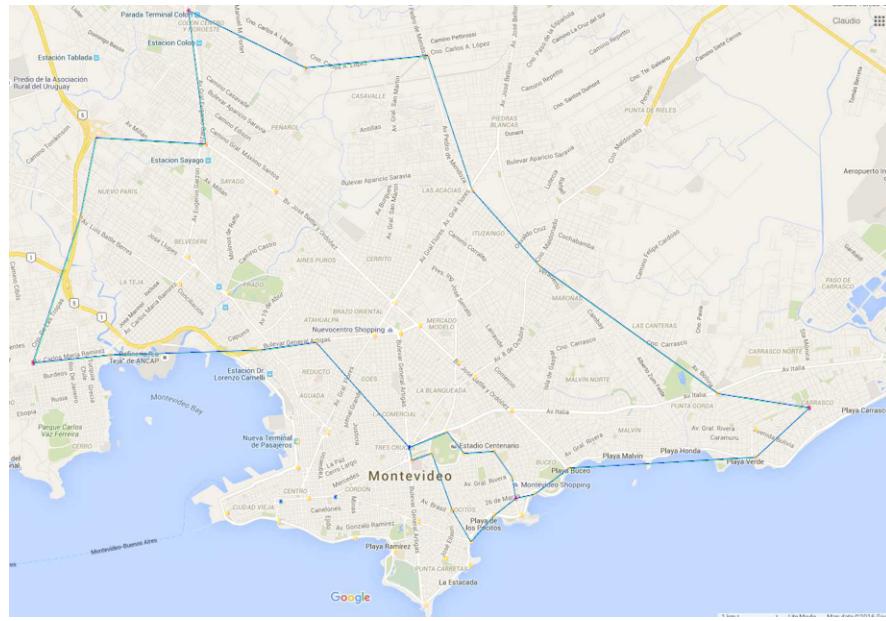
s.t.

$$\sum_{j \in S(i)} x_{ij}^k - \sum_{j \in E(i)} x_{ji}^k = \begin{cases} 0 & \forall i \neq k \\ f_i & \forall i = k \end{cases} \quad (\text{equilibrio de flujo})$$

$$0 \leq x_{ij}^k \leq u_{ij} \leq 1, \forall k \in \{4, 5, 6, 7\}$$

3.2. Resultados

El costo de construir estos caminos fue de 9109 y la imagen correspondiente a la solución óptima obtenida se encuentra a continuación:



Cuadro 2: Delay por tramos

Concentrador	Total P1	Promedio P1	Total P2	Promedio P2
Carrasco	2131	1066	4804	2402
Cerro	2045	1023	4804	2402
Pocitos	2063	688	5453	1818
Colon	2192	1096	4804	2402

En esta tabla puede apreciarse el aumento del delay al reutilizar al máximo los tramos. Este problema motiva la búsqueda de variantes en la siguiente sección para lograr costos razonables sin comprometer los delays de la red más allá de algún umbral establecido.

4. Parte III - Algunas variantes

El KnapSack Problem (KSP) puede verse como un problema de optimización en el que se busca maximizar la ganancia de artículos incluídos en la mochila considerando que los pesos respectivos de los objetos no superen un umbral total. La siguiente es una formulación LP de este problema:

$$\begin{aligned} \max \quad & \sum_{i=0}^n v_i x_i \\ \text{s.t.} \quad & \sum_{i=0}^n p_i x_i \leq b \\ & 0 \geq x_i \geq 1, \quad \forall i \in N \end{aligned} \tag{1}$$

Por otra parte, el problema del camino de costo mínimo (CCM) consiste en encontrar un CCM dados dos vértices s y t en un grafo G . En este problema el costo está dado por el peso de la arista pero existen varias variantes a este problema que incluyen restricciones adicionales dando lugar a una familia de problemas que recibe el nombre de Constrained Shortest Path Problems (CSPP).

Si suponemos que las aristas tienen dos atributos diferentes (peso y costo) y que queremos encontrar el CCM restricto a que los pesos de las aristas del camino no superen un umbral dado tenemos una variante del CCM que resulta ser la instancia más sencilla del problema a resolver en esta parte dado que como mínimo siempre tendremos al menos una fuente y un sumidero.

Por lo tanto, si probamos que esta variante es NP-C el problema de esta parte también será NP-C dado que en todas sus instancias precisaremos resolver al menos una vez un problema que es NP-C. Cabe notar que al probar la NP-Complejidad del problema queda demostrado que este es NP-Hard por definición de NP-Complejidad.

Realizaremos la prueba a partir de una reducción del KSP a la variante del CCM descrita anteriormente.

DEMOSTRACIÓN 1 *Dada una instancia del KSP con artículos en $\{1, \dots, n\}$, cada uno de valor v_i y peso p_i , para cada $j = 1, \dots, n$, construiremos un grafo de n nodos de la siguiente forma:*



En este grafo de aristas paralelas asociaremos a las superiores las tuplas $(M - v_j, p_j)$ ¹ y $(M, 0)$ a las inferiores considerando que en la variante del CCM el costo se corresponde con el primer componente de una tupla y la cantidad máxima de flujo de una arista con la segunda componente.

Luego, podemos ver entonces que si existe un camino de costo mínimo en el grafo que cumpla con las restricciones de flujo encontramos una solución para el KSP y viceversa con lo cual hemos encontrado una reducción y por tanto que la variante de esta parte es un problema NP-C.

¹ $M = \max\{v_j : j = 1, \dots, n - 1\}$

5. Parte IV - Abordaje por Metaheurísticas

5.1. Introducción

La metaheurística escogida para atacar el problema de esta parte ha sido Computación Evolutiva. Dentro de este espectro de técnicas se ha optado en particular por implementar un algoritmo genético con operadores de cruzamiento y mutación.

Como principales motivos de la elección se destacan los siguientes:

- La representación de individuos resulta bastante directa al considerar como genes de un individuo a los caminos que conforman una solución del problema.
- La función de fitness para evaluar a los individuos resulta sencilla como expresión del costo.
- La generación de una población inicial para este tipo de problemas puede hacerse polinómicamente y con un muestreo considerable resolviendo independientemente cada subproblema del grafo y combinando soluciones.
- Los operadores de cruzamiento y mutación también tienen una traducción casi directa en términos de caminos y genes de los individuos.
- La exploración del espacio de búsqueda también resulta natural a partir de la variedad de la población inicial y las posibilidades que surgen de la aplicación de los operadores.

En definitiva, al momento de optar por esta metaheurística se valoró la sencillez de la traducción de este problema en particular de grafos a uno de algoritmos genéticos, que la función de fitness no insumiera un overhead considerable (evaluar a un individuo) y que los parámetros de ajuste tendieran a ser pocos a partir de aprovechar una variedad inicial dada por la aleatorización de los caminos iniciales para cada individuo.

5.2. Inicialización

A partir del problema planteado en la parte I, se construyen soluciones para dicho problema desde una fuente hacia las terminales. Estos caminos surgen de perturbar los costos de las aristas y el mismo proceso se aplica para cada una de las fuentes. Luego, para cada conjunto de líneas factible (en términos de su delay) se combinan y se agrupan soluciones de cada parte para generar la red (esto es, un individuo). Experimentalmente se notó que existen al menos 2 caminos de costo mínimo distintos y factibles para cada fuente a los nodos del centro por lo que se obtiene una población inicial de al menos 16 individuos. Los detalles de estas pruebas pueden verse en el Anexo I.

5.3. Selección

Basada en ranking, todos los individuos sea rankean y la probabilidad de que sean seleccionados es inversamente proporcional a su posición en el ranking. Se seleccionan 2 a reproducirse y el hijo o los hijos se agregan a la población actual mientras que el miembro más débil es descartado. Así con cada iteración.

5.4. Cruzamiento y mutaciones

- (Cruzamiento) Buscar nodos en común entre individuos distintos y cruzar en ese punto (es decir que el resto del camino de uno pasa a ser el del otro y viceversa). En principio esto permitiría generar diversidad pero debe estudiarse la factibilidad de los individuos resultantes.
- (Mutación) Dado un individuo se buscan caminos de fuentes distintas que comparten un nodo y considerando el mínimo de los tramos faltantes a partir del nodo en común se toma ese mínimo y se modifican el resto de los tramos para continuar por el seleccionado. De esta forma se garantiza que los caminos resultantes van a continuar con delay mínimo y mejorando los costos al reutilizar el menor de los tramos a partir del nodo que tenían en común. La dificultad de este problema es buscar el camino de delay mínimo desde el nodo en común al sumidero que como vimos en la parte I es polinomial.
- (Mutación) Dado un individuo, tomar una línea y borrarla entera para regenerarla con la restricción de que tenga un delay menor o igual. La factibilidad de este enfoque deberá ser contrastado experimentalmente para estimar que tan posible es que siempre se obtenga la línea que había sido borrada y no una nueva (a pesar de que se incluye la restricción de no usar las aristas pertenecientes a las otras líneas).

6. Referencias consultadas

Referencias

- [1] Alves Pessoa, et. al. Robust constrained shortest path problems under budgeted uncertainty.
- [2] Ziegelmann, M. Constrained Shortest Path and Related Problems. Universitat der Saarlandes.
http://scidok.sulb.uni-saarland.de/volltexte/2004/251/pdf/MarkZiegelmann_ProfDrKurtMehlhorn.pdf
- [3] Lecture Notes: Solving linear and integer programs using the GNU linear programming kit. Duke University.
<https://www.cs.duke.edu/courses/spring08/cps296.2/solvers.pdf>
- [4] Lecture Notes: Integer Programming, Relaxations and Bounds. Technische Universität Kaiserslautern.
http://www.mathematik.uni-kl.de/fileadmin/AGs/opt Lehre/WS1314/IntegerProgramming_WS1314/ip-chapter6.pdf
- [5] Lecture Notes: The Knapsack Problem. University of Texas at Dallas.
<https://www.utdallas.edu/~scniu/OPRE-6201/documents/DP3-Knapsack.pdf>

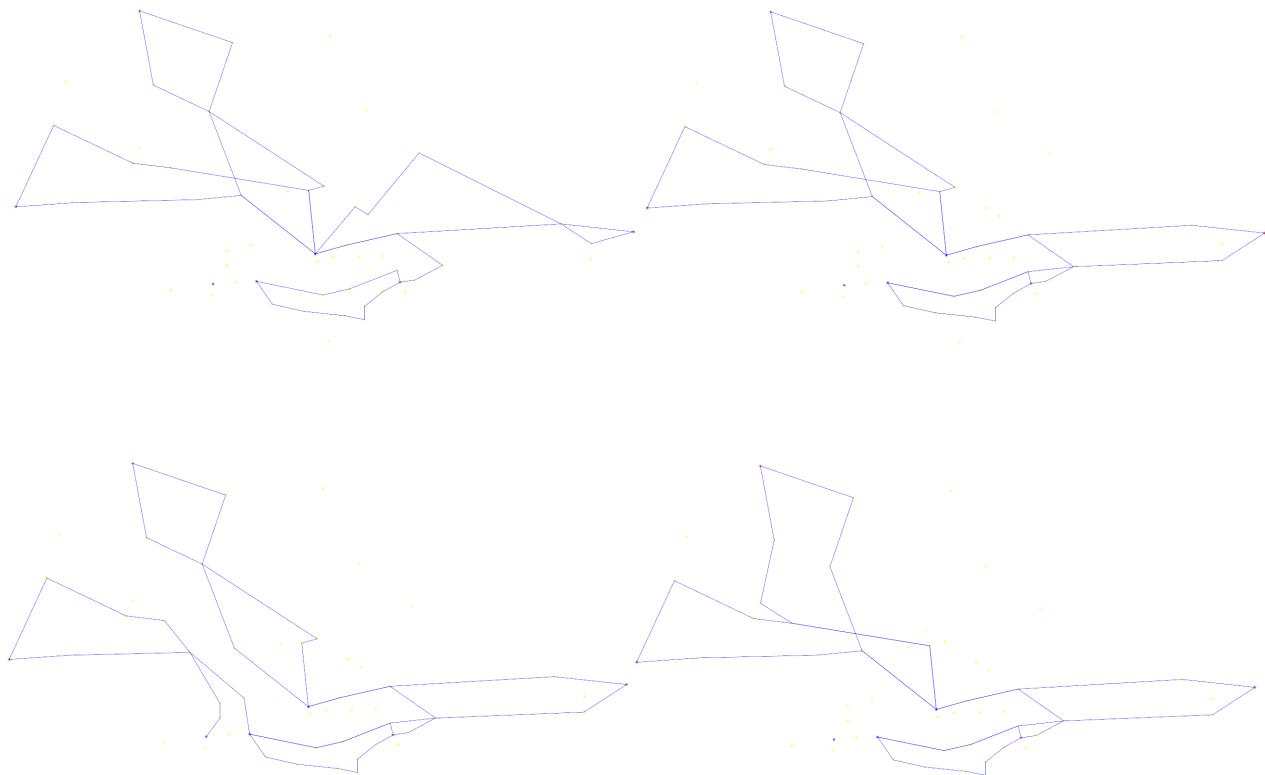
7. Anexo I: Pruebas realizadas

7.1. Cálculo de cotas y factibilidad de la población inicial

A efectos de establecer las cotas de los delays para la parte IV, se resolvió el problema de maximizar el delay para un caminimo a partir de las soluciones de la parte I. Obteniendo como cotas superiores los valores 1263, 1187, 841 y 1263 para Carrasco, Cerro, Pocitos y Colón respectivamente.

Por otra parte se realizaron pruebas para determinar la factibilidad de la población inicial propuesta. En este sentido se encontró que perturbando el costo de las aristas, un 50 % de las soluciones encontradas eran de utilidad para construir un individuo.

A continuación se muestran individuos generados a partir del método propuesto:



8. Anexo II: Instrutivo de las implementaciones realizadas

8.1. Introducción

Para las partes I, II y III se programaron scripts en Octave que se encuentran en los directorios Parte1, Parte2 y Parte3 del entregable que acompaña este informe.

8.2. Parte I

El script implementado para esta parte puede recibir un parametro entero (1, 2 o 3) que le indicará que modo ejecutar y su fichero correspondiente es el archivo parte1.m

- 1 - minimza costos.
- 2 - minimza delays.
- 3 - crea un CCM para cada fuente.

El mismo usa la librería glpk que viene en Octave y fue necesario adaptar el archivo original primero a una estructura auxiliar (para facilitar el manejo de datos) y luego a una matriz de ecuaciones para el problema de minimización.

8.3. Parte II

En este caso, el script implementado no recibe parámetros y su ejecutable es el fichero parte2.m

Fue implementado a partir del problema anterior y adaptado para la realidad de este otro agregando más restricciones (ecuaciones).

8.4. Parte III

El fichero parte3.m implementa un conjunto de funciones utilizado para obtener estadísticas, probar inicializaciones y detectar fallos de éstas. Se verificó la factibilidad de la inicialización, la perturbación de valores y se graficaron los individuos factibles. Se hicieron distintas pruebas en el código, esta última versión refleja lo más sustantivo.