# Mathematical Aspects of Public Transportation Networks

Problem Set 10

So 2018

Dimitrios Bogiokas - BMS
FU ID: 5048996

## Exercise 1

(a) Let $l_e$ be the trivial path containing only the edge $e$, for every $e \in E$. Then the vector $x \in \{0,1\}^{\mathcal{L}}$, with:

$$x_l = \begin{cases} 1 & , \quad \exists e \in E: \; l = l_e \\ 0 & , \quad \text{else} \end{cases}$$

is a solution to the $k$-line pool problem, if $k \geq |E|$. Indeed:

$$\sum_{l \in \mathcal{L}} x_l = |E| \leq k$$

Moreover, choose the vector $f \in (\mathbb{N}_0)^{\mathcal{L}}$ to be:

$$f_l = \begin{cases} f_e^{\min} & , \quad \exists e \in E: \; l = l_e \\ 0 & , \quad \text{else} \end{cases}$$

obviously satisfying $f_l \leq F x_l$ for all $l \in \mathcal{L}$. Then, for every $e \in E$, we have:

$$\sum_{l \in \mathcal{L}_e} f_l = f_{l_e} = f_e^{\min} \in [f_e^{\min}, f_e^{\max}]$$

(b) Let $G = (V, E)$ be a directed graph. Define then the undirected graph $\tilde{G} = (\tilde{V}, \tilde{E})$, with vertices:

$$\tilde{V} := \bigcup_{v \in V} \{v^i, v^o\}$$

and edges:

$$\tilde{E} := \bigcup_{v \in V} \left\{\{v^i, v^o\}\right\} \cup \bigcup_{(u,v) \in E} \left\{\{u^o, v^i\}\right\}$$

Also let $k = 1$ and for every $e \in \tilde{E}$, let $f_e^{\max} = 1$ and:

$$f_e^{\min} = \begin{cases} 1 & , \quad e = \{v^i, v^o\} \\ 0 & , \quad e = \{u^o, v^i\} \end{cases}$$

Obviously we have then also $F = 1$. Then, the 1-line pool generation problem on $\tilde{G}$, with this frequency function has a solution if and only if $G$ has a directed Hamilton path. Indeed:

($\Leftarrow$) Let $P = (v_1, v_2, \dots, v_n)$ be a directed Hamilton path on $G$. Define then:

$$\tilde{P} := (v_1^i, v_1^o, v_2^i, v_2^o, \dots, v_n^i, v_n^o)$$

to be a walk in $\tilde{G}$. This is indeed a walk, since the edges $\{v_j^i, v_j^o\}$ are indeed elements of $\tilde{E}$, for every $j \in [n]$ and also the edges $\{v_j^o, v_{j+1}^i\}$ are also elements in $\tilde{E}$, for every $j \in [n-1]$, since $(v_j, v_{j+1}) \in E$. Moreover, $\tilde{P}$ is vertex-distinct and thus a path, since $P$ was vertex-distinct. Let now $x_{\tilde{P}} = f_{\tilde{P}} = 1$ and $x_l = f_l = 0$, for all other paths $l$. Then, obviously $f_l = Fx_l$, for every $l \in \tilde{\mathcal{L}}$, path in $\tilde{G}$, and $\sum_{l \in \tilde{\mathcal{L}}} x_l = 1 = k$. Moreover, let $e \in \tilde{E}$. Then, we have:

- If $e = \{v^i, v^o\}$ for some $v \in V$, we have that $P$ goes through $v$ (since it is a Hamilton path) and thus $v = v_j$ for some index $j$. This means that $e$ is an edge of $\tilde{P}$, which gives:

$$\sum_{l \in \tilde{\mathcal{L}}_e} f_l = f_{\tilde{P}} = 1 = f_e^{\min} = f_e^{\max}$$

- If $e = \{u^o, v^i\}$ for some $(u, v) \in E$, we have that:

$$f_e^{\min} = 0 \leq \sum_{l \in \tilde{\mathcal{L}}_e} f_l \leq f_{\tilde{P}} = 1 = f_e^{\max}$$

  where the second inequality depends on $e$ being an edge of $\tilde{P}$ or not.

($\Rightarrow$) For the inverse, consider $x \in \{0,1\}^{\tilde{\mathcal{L}}}$ be a solution to this 1-line pool generation problem on $\tilde{G}$. Since

$$\sum_{l \in \tilde{\mathcal{L}}} x_l \leq 1$$

we have that either $x_l = 0$ for every path $l \in \tilde{\mathcal{L}}$, or there exists a path $L \in \tilde{\mathcal{L}}$ with $x_L = 1$ and $x_l = 0$ otherwise. In the first case, we also get that $f_l \leq Fx_l = 0$. Let $v \in V$ be any vertex. Then for the edge $e = \{v^i, v^o\} \in \tilde{E}$, we get:

$$\sum_{l \in \tilde{\mathcal{L}}_e} f_l = 0 < 1 = f_e^{\min}$$

This means, that this cannot be true and leaves us only with the latter case. We first prove that this path $L$ is a Hamilton path in $\tilde{G}$. Indeed, let $v \in V$. Then, for the edge $e = \{v^i, v^o\} \in \tilde{E}$ we have:

$$f_e^{\min} \leq \sum_{l \in \tilde{\mathcal{L}}_e} f_l \leq f_L = 1 = f_e^{\min}$$

where the second inequality is an equality only if $L \in \tilde{\mathcal{L}}_e$. Since in the expression above every inequality is an equality, we have that $e$ is an edge in the path $L$. In particular, both $v^i$ and $v^o$ are vertices in this path. Next notice that $\tilde{G}$ is a bipartite graph (colour every vertex of the form $v^i$ red and everyone of the form $v^o$ blue), which means in particular, that the vertices of the path $L$ alternate between these two forms, if one moves along the path. Since the two parts of $\tilde{G}$ have the same amount of vertices, namely $n = |V|$, we know that $L$ starts with some vertex of the form $v^i$ and ends with one of the form $u^o$, or the other way around. Without loss of generality, we will assume that $L$ is of the former type. Indeed, if $L$ is of the latter type, just reversing the traversing direction of $L$, gives us a new path of the former type, satisfying the same assumptions.

Since all edges of the form $\{v^i, v^o\}$ are edges of $L$, we make the following two observations:

- It can never have the case that $(\ldots, u^o, v^i, w^o, \ldots)$ is a piece of the path for $u \neq v \neq w$ (Because otherwise $v^i$ would have degree at least 3 in the path). Thus, at least one of $u, w$ are the same as $v$ (in fact exactly one).
- It cannot be the case that $(v^i, u^o, \ldots)$ is the start of the path for $v \neq u$ (Because otherwise $v^i$ would have degree at least 2 in the path, but it is an endpoint).

These facts mean, that $L$ has the following form:

$$(v_1^i, v_1^o, v_2^i, v_2^o, \ldots, v_n^i, v_n^o)$$

for some ordering $v_1, v_2, \ldots, v_n$ of the vertices in $V$. Define then $\bar{L}$ in $G$ to be the walk $(v_1, v_2, \ldots, v_n)$. This is indeed a directed walk, since $\{v_j^o, v_{j+1}^i\}$ is in $\tilde{E}$ iff $(v_j, v_{j+1})$ is in $E$. Then we can easily see that $\bar{L}$ is in fact a path, because $L$ was already vertex-distinct, which gives in particular that $v_i \neq v_j$ for every $i \neq j$. So, we proved that $\bar{L}$ is a Hamiltonian path in $G$.

For completeness reasons, it should also be mentioned that this reduction from the directed Hamiltonian path problem to the 1-line generation pool problem is computable in polynomial time. Indeed, on any graph input we need linear time to define the new graph and the capacity functions.

**Exercise 2** The Steiner connectivity problem is finding a vector $x \in \{0,1\}^{\mathscr{P}}$, which minimizes

$$\sum_{P \in \mathscr{P}} x_P c_P$$

under the constrains that

$$T \subseteq \bigcup_{\substack{P \in \mathscr{P} \\ x_P = 1}} V(P)$$

and that

$$\bigcup_{\substack{P \in \mathscr{P} \\ x_P = 1}} P \text{ is connected}$$

We will prove that a vector $x \in \{0,1\}^{\mathscr{P}}$ solves the Steiner connectivity problem iff it is a solution for the given integer program, up to zero-weight paths. This means that there might exist some $x \in \{0,1\}^{\mathscr{P}}$ solving the given integer program and not solving the Steiner connectivity problem, but we will prove that in this case there exists some $x' \leq x$, such that

$$0 = x'_P < x_P = 1 \quad \Rightarrow \quad c_P = 0$$

Since the quantity that should be minimized is the same, it suffices to prove that the above mentioned constrains are satisfied by some $x$ (up to zero-valued paths), iff $x$ satisfies the constrain:

$$\sum_{P \in \mathscr{P}(W)} x_P \geq 1 \qquad \forall W \subseteq V,\ \emptyset \neq W \cap T \neq T$$

($\Rightarrow$) Let $x \in \{0,1\}$ satisfying the two constrains of Steiner connectivity problem and let $W \subseteq V$, with $\emptyset \neq W \cap T \neq T$. Since $W \cap T \neq T$, there exists some terminal $t \notin W$ and since $W \cap T \neq \emptyset$, there exists some terminal $t' \in W$. Due to the first constraint, $t, t' \in \bigcup_{\substack{P \in \mathscr{P} \\ x_P = 1}} V(P)$, which in particular means that there exist nodes of the path collection both in $W$ and not in $W$. Since the union of paths is connected in $G$, there exists some path $P$ with some vertex in $W$ and some not in $W$. Thus, this path $P$ has an edge $e$, with $|e \cap W| = 1$, i.e. $P \in \mathscr{P}(W)$.

($\Leftarrow$) For the inverse, let $x \in \{0,1\}$ satisfying the constrain of the integer program. And let $t \in T$. For the set $W = \{t\}$ apply the integer-programing constrain, which gives us exactly that there exists some path $P$, with $x_P = 1$ passing through $t$, i.e. $t \in \bigcup_{\substack{P \in \mathscr{P} \\ x_P = 1}} V(P)$, giving us the inclusion $T \subseteq \bigcup_{\substack{P \in \mathscr{P} \\ x_P = 1}} V(P)$. Let

$$C \subseteq \bigcup_{\substack{P \in \mathscr{P} \\ x_P = 1}} P$$

be a connected component of the union of paths, containing at least one terminal $t \in T$. If there exists some $t' \in T \setminus C$, then for $W = C$, using the integer program constrain, we get some path $P \in \mathscr{P}(C)$ with $x_P = 1$ having vertices both in $C$ and outside $C$, wich contradicts the choice of $C$. Thus, $T \subseteq C$. Let $x' \in \{0,1\}^{\mathscr{P}}$, with

$$x'_P = \begin{cases} x_P & , \quad P \subseteq C \\ 0 & , \quad \text{otherwise} \end{cases}$$

Then, obviously $x' \leq x$ and if any path is satisfying $0 = x'_P < x_P = 0$, then $P$ cannot include any terminals in its vertices, thus ommiting $P$ also gives a valid solution to the linear program. In fact ommiting $P$ gives a strictly better solution, w.r.t. the minimization, unless $c_P = 0$, which is exactly the case described above.

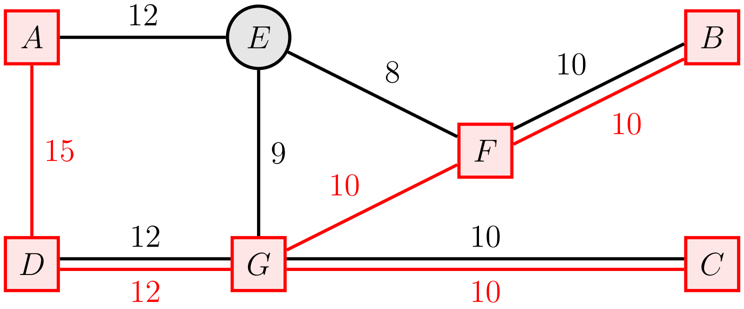
**Exercise 3**

(a) Let $G$ be the given graph and $G^c$ be its metric closure, w.r.t. the given edge costs $c$. For this we used the Floyd-Warshall algorithm in the C++ program you can find attached. This returned the following distance matrix:

| | $A$ | $B$ | $C$ | $D$ | $E$ | $F$ | $G$ |
|---|---|---|---|---|---|---|---|
| $A$ | 0 | 25 | 31 | 15 | 12 | 20 | 21 |
| $B$ | 25 | 0 | 15 | 32 | 13 | 10 | 20 |
| $C$ | 31 | 15 | 0 | 22 | 19 | 20 | 10 |
| $D$ | 15 | 32 | 22 | 0 | 21 | 22 | 12 |
| $E$ | 12 | 13 | 19 | 21 | 0 | 8 | 9 |
| $F$ | 20 | 10 | 20 | 22 | 8 | 0 | 10 |
| $G$ | 21 | 20 | 10 | 12 | 9 | 10 | 0 |

which is of course symmetric. Moreover, we notice that the smallest distance between two vertices joined by an edge is the cost of this edge.

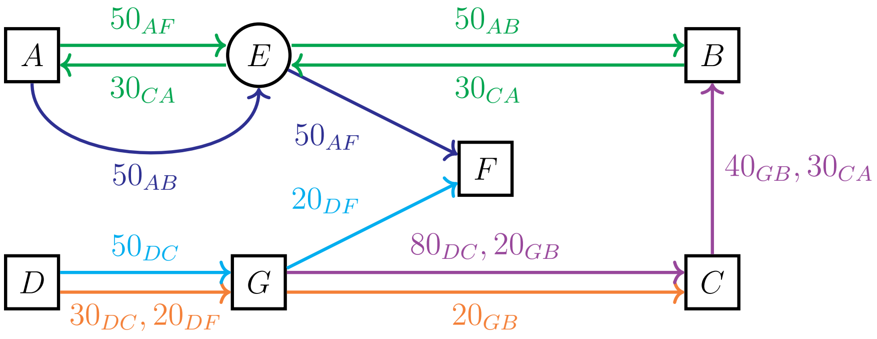
Then, we know that it suffices to find the cost of a Minimum Weight Spanning Tree in $G^c$ and one in $G^c \setminus v_E$ and compare their costs. Theoretically after that, we should compute the induced trees in $G$, but the program so happens to output MWSTs with edges also existing in the original graph (with the same cost, as we just mentioned). Thus, the two spanning trees of interest are:



with red colour a MWST of cost 57 in $G^c \setminus v_E$ and with black a MWST of cost 61 in $G^c$.

Of these two, the red one has less cost and thus this is a Steiner tree of $G$ of minimum cost.

(b) For this we used a C++ program to generate an MIP minimizing the sum of the frequences each line is used. The two directions of each line $l$ were regarded as different lines, numbered as $l$ and $l + 7$. Unfortunately, optimal solutions of this MIP did not in general avoid loops of imaginary passengers, traveling on empty spaces in the lines used for solving the program. I removed one such loop by hand from the SCIP solution, to get the following valid line plan:



where, the labels are the number of people traveling on each line and the subscripts denote the demands these people satisfy. Also with different colours we denote different lines:

|  |  |  |
|---|---|---|
| Line 2 | AEB | Green |
| Line 3 | AEF | Blue |
| Line 4 | DGC | Orange |
| Line 6 | BCGF | Purple |
| Line 7 | DGF | Cyan |

In the solution above, we use Line 2 twice, Line 3 once, Line 4 once, Line 6 twice and Line 7 once. This makes a total of 7 lines used.