Mathematical Aspects of Public Transportation Networks
Problem Set 9

So 2018

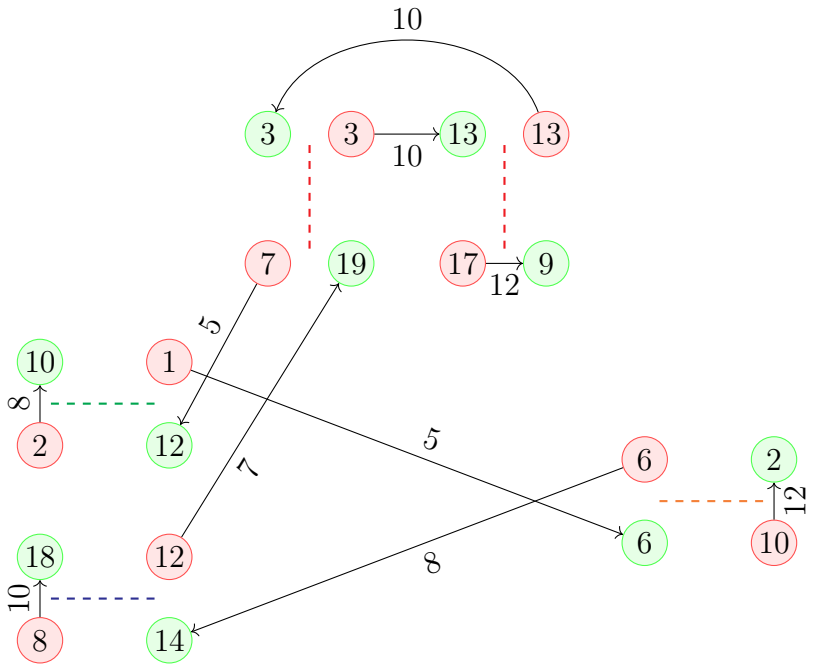
Dimitrios Bogiokas - BMS
FU ID: 5048996

Exercise 1

(a) A minimum cost circulation covering each driving activity includes (except for every driving activity) a perfect matching on every station. In Köpenick the perfect matching trivially contains the only transfer activity. In Lichtenberg and in Faltenberg there are two perfect matchings in each to choose from and one can easily find the ones that are of minimum cost. With a little help of a C++ program, which is included, one can see that in Schöneweide every minimum weight matching has weight 37. Thus we have with respect to x:

$$\begin{aligned} \min &= (4 \cdot 44 + 2 \cdot 34 + 50 + 51 + 2 \cdot 24) + 12 + (10 + 8) + (10 + 10) + 37 \\ &= 393 + 12 + 18 + 20 + 37 = 480 \end{aligned}$$

(b) Using the same program, we also get all particular matchings achieving cost 37 in Schöneweide. Thus one minimum-weight perfect matching of (V, E_t) is:



The dashed lines just indicate where are the network lines.

(c) An optimal periodic vehicle schedule requires $\frac{480}{20} = 24$ vehicles.

(d) There is a bijection between the perfect matchings of (V, E_t) and the distinct periodic vehicle schedules in $\mathcal{E}$. Thus, the number of the latter equals with

$$b(K_{1,1}) \cdot b(K_{2,2})^2 \cdot b(K_{5,5})$$

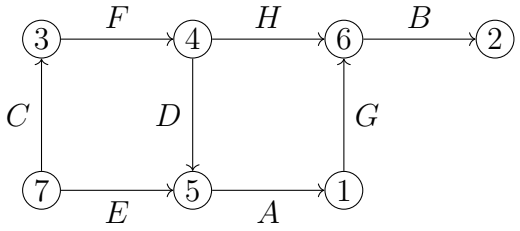
where $b(G)$ is the number of different perfect matchings of G. If $G = K_{t,t}$, $b(G)$ is the number of permutations of $[t]$, i.e. $b(K_{t,t}) = t!$. This gives us:

$$1! \cdot (2!)^2 \cdot 5! = 480$$

different periodic vehicle schedules in $\mathcal{E}$.

Exercise 2

(a) Each column of the matrix has exactly one 1 and one -1 entry. Thus, A is the incidence matrix of the following oriented graph and thus totally unimodular.



where the numbers correspond to the rows and the letters to the columns.

(b) For this, we again needed a program. The programm you got almost checks every possible determinant, of growing size. The following arguments helped reduce the case spaces a bit, but the algorithm is still exponential, which suffices to find an 7×7 submatrix with determinant not in $\{-1, 0, 1\}$, but not to find every one of them without spending some good weeks.

- First, such a submatrix should have at least one row from the region involving the identity matrices. Otherwise the determinant would be the product of the determinants of two submatrices of A and thus in $\{-1, 0, 1\}$.
- Every row from the region with the identity matrices should hit both diagonals, i.e. have two non-zero entries. Indeed, if all entries are zero, then the matrix has determinant 0 and if there is only one of them non-zero, the determinant equals $\pm$ the determinant of the matrix without this row and column. We avoided this case, because we were looking for minimal submatrices.

- There should be at least one column not hitting the diagonal of the respective identity matrix. Indeed, if all columns had a non-zero entry in the region of the identity matrices, then the resulting matrix would look (up to a row-permutation) like:

$$\det \left(\begin{array}{c|c} A' & 0 \\ 0 & A'' \\ \hline I_k & I_k \end{array} \right)$$

where k is some appropriate integer and both $\begin{pmatrix} A' \\ 0 \end{pmatrix}$ and $\begin{pmatrix} 0 \\ A'' \end{pmatrix}$ are some square matrices, where A', A'' are submatrices of A, on the same column set. The above determinant, due to the fact that I_k permutes with itself equals:

$$\det \begin{pmatrix} A' \\ -A'' \end{pmatrix} = (-1)^k \det \begin{pmatrix} A' \\ A'' \end{pmatrix}$$

which is the determinant of a square submatrix of A, up to a sign and thus has a value in $-1, 0, 1$

The program first chooses some num to be the number of rows and columns of the submatrix. Then, it chooses some kI $\in \left[1, \frac{\text{num}}{2}\right]$ to be the number of choosen rows in the identity-matrices region. Then we choose kI of 8 to be these rows, which means that we automatically have chosen 2kI columns as well (we want for these rows to hit both diagonals). Next, we choose num $-$ kI rows, among the 14 rows above, to complete the row choice and num $-$ 2kI columns, among the ones we did not choose yet to complete the column choice. Then it just checks the determinant. The first matrix it outputs, for num $= 7$ and kI $= 2$ is the following:

$$M = \begin{array}{c} 4 \\ 6 \\ \\ \\ \\ F \\ G \end{array} \left(\begin{array}{ccc|cccc} 1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 \\ \hline 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{array} \right) \begin{array}{c} \\ \\ 1 \\ 4 \\ 5 \\ \\ \end{array}$$

$$\begin{array}{ccc|cccc} F & G & H & A & D & F & G \end{array}$$

with $\det(M) = 2$. The numbers and the letters refer to the rows and columns of the original matrix A.

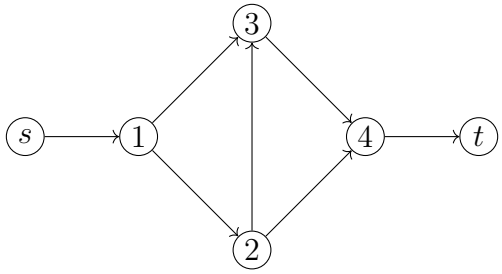
(c) Due to the Thm of Hoffman and Kruskal (1956) totally unimodular matrices A are characterized by the fact that the extreme points of $\{x|Ax \leq b\}$ are integer, for any integer b. This means, that for the 22×16 matrix B of (b) there exists some $c \in \mathbb{Z}^{16}$ and $b \in \mathbb{Z}^{22}$ such that:

$$\arg\max_{x \in \mathbb{R}^{22}} \{c^T x : Bx \leq b\} \not\subseteq \mathbb{Z}^{16}$$

or at least, it should in theory. Let:

$$A = \begin{pmatrix} 1 & -1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & -1 \end{pmatrix}$$

This matrix is the incident matrix of the following directed graph, having removed the rows corresponding to vertices s, t:



It is easy to see that A is TU and that the construction of (b) for A is TU iff the construction of (b) for A' is TU, where A' is A without the first and the last column. The program 2c.cpp proves that the matrix

$$B' = \begin{pmatrix} A' & 0 \\ 0 & A' \\ I & I \end{pmatrix}$$

is not totally unimodular. The code in 2c.sh tries to solve a general max 2-commodity flow in the above graph, with common source s and common target t, with positive integer (and random) cost, balance, capacity, many times in a row but unfortunately fails to find one with non-integer solution, even though the matrix B of the constraints is exactly:

$$B = \begin{pmatrix} A & 0 \\ 0 & A \\ I & I \end{pmatrix}$$

and thus not TU.