# Exam problems for the course 'Network Programming' (DM817) Part B

Department of Mathematics and Computer Science
University of Southern Denmark

The problems are made available from the homepage no later than December 7, 2020.
The solutions must be returned by January 11, 2021 at 9.00 a.m.
You should hand in via SDU assignment in blackboard (use your name and birthdate but NOT the last 4 digits of the cpr number)

It is important that you explain how you obtain your answers and argue why they are correct. If you are asked to describe an algorithm, then you must supply enough details so that a reader who does not already know the algorithm can understand it (but you should not give pseudo code). You should also give the complexity of the algorithm when relevant. Note also that illustrating an algorithm means that one has to follow the steps of the algorithm meticulously (slavisk). When you are asked to give the best complexity you can find for a problem, you must say which (flow) algorithm you use to get that complexity and give a reference to its complexity (in the book) or prove it directly. All algorithms asked for should be polynomial in the size of the input. Unless otherwise stated, the numbers $n$ and $m$ always denote the number of vertices and arcs of the network or digraph in question.

As with the first part A of the exam problems, there are 100 points to earn in total for this part B of the problems. The final grade for the course will be based on an overall impression of your performance on both sets of problems.

**You may refer to results from Ahuja and Bang-Jensen and Gutin as well as results from the course page and from exercises that have been posed on the weekly notes, but not to material found elsewhere (this includes exercises that we have not done in the course). It is strictly forbidden to work in groups and any exchange of ideas and results before the deadline for handing in will be considered as exam fraud.**

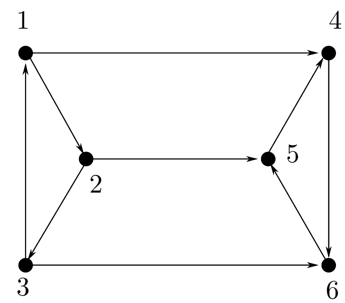


Figure 1: A digraph which is not 2-arc-strong.

## PROBLEM 1 (15 point)

Recall that a digraph $D = (V, A)$ is **$k$-arc-strong** if every non-trivial subset $X \subset V$ has out-degree at least $k$. We denote by $\lambda(i, j)$ the maximum number of arc-disjoint $(i, j)$-paths in $D$. Menger's Theorem implies that $D$ is $k$-arc-strong if and only if $\lambda(i, j) \geq k$ for every pair of distinct vertices $i, j \in V$.

### Question a:

Explain how to use maximum flow calculations to decide, for a given input digraph $D$ and a natural number $k$, whether $D$ is $k$-arc-strong. Give the best complexity you can find for this problem using the algorithm you propose.

### Question b:

Illustrate Frank's algorithm for making a directed multigraph $k$-arc-strong (BJG Section 7.6) on the digraph in Figure 1 when $k = 2$. You do not have to give all details. It is sufficient to explain in words how you decide which arcs can be deleted (to and from the new vertex $s$) and which splittings are admissible (e.g. based on values of $\lambda(i, j)$ for suitable pairs of vertices $i, j$. These values may be found by inspection, you don't need to run a flow algorithm). **Note that you should process the vertices in the order 1,2,3,4,5,6.**

## PROBLEM 2 (15 point)

Suppose we are given a network $\mathcal{N} = (V, A, \ell, u, b)$ for which no feasible flow exists.

### Question a:

Prove by a short argument that if we forget the balance constraints $b$, then there is always a feasible flow with respect to the lower bounds.

Suppose also that without the lower bound constraints there is a feasible flow which meets the balances and respects the capacities. Now we are interested in finding the minimum total amount we have to decrease the lower bounds in order obtain a network $\mathcal{N}' = (V, A, \ell', u, b)$ which has a feasible flow. That is, we want to minimize $\sum_{ij \in A}(\ell_{ij} - \ell'_{ij})$ over all networks $\mathcal{N}' = (V, A, \ell', u, b)$ which have a feasible flow.

**Question b:**

Formulate this problem as a convex cost flow problem. Remember to argue why your model is correct. Hint: compare with Ahuja Exercise 14.4.

**Question c:**

Given a network $\mathcal{N} = (V, A, \ell \equiv 0, u)$ and two feasible flows $x$ and $y$ such that $b_x \not\equiv b_y$ (that is, their balances differ for at least one vertex). Decsribe an algorithm for finding the smallest integer $\alpha$ such that we can transform $x$ into a flow $x'$ with $b_{x'} \equiv b_y$ and $|x_{ij} - x'_{ij}| \leq \alpha$ for all arcs $ij \in A$. Hint: work in $\mathcal{N}(x)$ and use convex cost flows.

**Question d:**

Show by a small example that $\alpha$ above is not necessarily equal to $\max\{|x_{ij} - y_{ij}| : ij \in A\}$.

## Problem 3 (20 point)

Recall that an out-branching rooted at $r$ in a digraph $D$ is just a spanning tree in the underlying graph of $D$ such that $r$ has a directed path to all other vertices using only arcs of the branching. We denote such an out-branching by $B_r^+$. This problem is about branching flows and minimum cost out-branchings.

Let $\mathcal{N} = (V, A, \ell \equiv 0, u \equiv n-1)$ be a network on $n$ vertices and $m$ arcs, all of which have capacity $n-1$. If $r \in V$ is a vertex of $\mathcal{N}$, then a **branching flow** from $r$ in $\mathcal{N}$ is a flow $x$ whose balance vector satisfies that $b_x(r) = n-1$ and $b_x(v) = -1$ for all $v \in V - r$. Recall that two flows $x, y$ in a network $\mathcal{N}'$ are **arc-disjoint** if $x_{ij} \cdot y_{ij} = 0$ for every arc $ij$ of $\mathcal{N}'$. Below we assume that all flows are integer valued, that is, $x_{ij}$ is an integer for every arc $ij \in A$.

**Question a:**

Let $r \in V$ be fixed. Prove that $\mathcal{N}$ has a branching flow from $r$ if and only if $r$ can reach all vertices by directed paths in $\mathcal{N}$, that is, there is an out-branching $B_r^+$ from $r$ in $\mathcal{N}$. You should also explain how to construct a branching flow from $r$ when one exists.

**Question b:**

Prove that $\mathcal{N} = (V, A, \ell \equiv 0, u \equiv n-1)$ has two arc-disjoint branching flows $x, x'$ from $r$ if and only if the capacity of every cut $(S, \bar{S})$ with $r \in S$ and $\bar{S} \neq \emptyset$ is at least $2n-2$. Hint: Use Edmonds' Branching theorem (Bang-Jensen and Gutin Theorem 9.5.1).

**Question c:**

Suppose now that we also have a non-negative cost function $c : A \rightarrow \mathbf{R}_0$ on the arcs so $\mathcal{N} = (V, A, \ell \equiv 0, u \equiv n-1, c)$. Describe an algorithm which, for a given vertex $r$ of $\mathcal{N}$, either produces a minimum cost branching flow from $r$ or a certificate that $\mathcal{N}$ has no branching flow from $r$. Try to obtain the best possible complexity for the problem. You should also illustrate your algorithm on the network that we obtain from digraph in Figure 2 by adding a capacity $4 = n-1$ to each arc in the digraph.

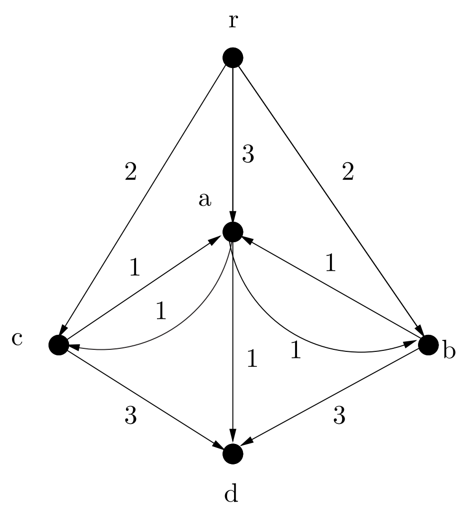


Figure 2: An arc-weighted digraph. The number on each arc is the cost of the arc.

**Question d:**

Recall that a minimum cost out-branching $B_r^+$ in an arc weighted digraph $D = (V, A, c)$ is an out-branching from $r$ such that $c(B_r^+)$ (the sum of the arc costs of $B_r^+$) is less than or equal to the cost of every other out-branching from $r$. Give a short description of an algorithm for finding a minimum cost out-branching in an arc-weighted digraph and illustrate the algorithm on the digraph in Figure 2.

**Question e:**

Let $\mathcal{N}(D) = (V, A, \ell \equiv 0, u \equiv n-1, c)$ be obtained from the arc-weighted digraph $D = (V, A, c)$ by giving each arc capacity $n-1$. Show by an example that the following algorithm will not always find a minimum cost out-branching from $r$: first find a minimum cost branching flow $x$ from $r$ such that all arc flows are integers and then return those arcs of $A$ for which $x_{ij} \neq 0$. Try to explain why the algorithm may fail.

## PROBLEM 4 ( 15 point)

This problem is about maximum weight closures in digraphs (Ahuja Section 19.2).

**Question a:**

Explain briefly how to reduce the problem of finding a maximum weight closure in a digraph $D = (V, A)$ with weights $\omega : V \rightarrow \mathbf{R}$ to a minimum cut problem in a suitable network.

**Question b:**

How can we use the algorithm above to find a maximum weight closure of minimum size (that is we want to minimize $|X|$ over all maximum closures)? Explain also how we can find a maximum closure of maximum size. What is the complexity of your algorithm? Hint: compare with BJG exercise 3.35.

**Question c:**

Explain how to modify the polynomial algorithm for finding a maximum closure so that you can find the maximum weight of a subset $X \subseteq V$ where $d^+(X) \leq 1$ (instead of $d^+(X) = 0$ as for a closure). State the complexity of your algorithm. Can you generalize your solution to a polynomial algorithm for finding the maximum weight of a subset whose out-degree is at most $k$ for some constant $k$?

**Question d:**

Explain how to modify the algorithm above so that you can find the maximum weight of a subset $X \subseteq V$ where $d^+(X) = 1$. Hint: how can you make sure a certain vertex is always (respectively, is never) contained in the optimal set?

## PROBLEM 5 (35 point)

In this problem digraphs may have parallel arcs. The underlying graph (denoted $UG(D)$) of a digraph $D$ is the multigraph $G = (V, E)$ which we obtain by replacing each arc $u \rightarrow v$ by an edge $uv$ (so $|E| = |A|$ and $pq \in E$ if and only if at least one of the arcs $pq, qp$ is in $A$).

Recall the following theorem, due to Edmonds, which characterizes digraphs for which there are $k$ arc-disjoint out-branchings $B^+_{r,1}, \ldots, B^+_{r,k}$ all rooted at the same vertex $r$. See BJG Section 9.5.

**Theorem 0.1 (Edmonds' Branching Theorem)** *A digraph $D = (V, A)$ with $r \in V$ contains $k$ arc-disjoint out-branchings rooted at $r$ if and only if*

$$d^-_D(X) \geq k \qquad \forall X \subseteq V - r \tag{1}$$

**Question a:**

Explain how to check in polynomial time, for a given input digraph $D = (V, A)$ and $r \in V$, whether (1) holds.

Suppose below that $D = (V, A)$ does not have $k$ arc-disjoint out-branchings from a prescribed $r \in V$. Our goal will now be to either add new arcs or reverse some arcs so that the resulting digraph $D'$ satisfies (1).

**Question b:**

Show that if we can add some set of $p$ arcs to $D$ so that (1) holds in the new digraph, then we can also add $p$ arcs, all of which start in $r$ so that (1) holds in the new digraph.

**Question c:**

Show that a digraph $D' = (V, A')$ has $k$ arc-disjoint out-branchings rooted at $r$ if and only if the digraph $D^* = (V, A' \cup A^*)$, which we obtain from $D'$ by adding $k$ parallel arcs from $v$ to $r$ for each $v \in V - r$, is $k$-arc-strong.

**Question d:**

Prove, that the minimum number, $\beta_k(D)$, of new arcs that we must add to $D$ in order to get a digraph which satisfies (1) is equal to the minimum[1] number $\beta$ which satisfies

$$\beta \geq \sum_{X \in \mathcal{F}} (k - d^-(X)), \tag{2}$$

for every subpartition $\mathcal{F}$ of $V - r$.

**Question e:**

Show that using Frank's algorithm for augmenting the arc-connectivity of a digraph (Bang-Jensen and Gutin, 2001, Section 7.6) we can solve the problem of finding the minimum number of new arcs we need to add to $D$ such that the resulting digraph satisfies (1). You may formulate a modified version of the algorithm, directly tailored to the branching problem if you wish.

**Question f:**

Formulate the problem of deciding whether we can reorient some arcs of $D$ so that (1) holds in the resulting digraph $D'$ as a submodular flow problem. Remember to argue why your formulation is correct.

**Question g:**

Argue that it is possible to reorient some arcs of $D$ so that (1) holds in the resulting digraph $D'$ if and only if $UG(D)$ contains $k$ edge-disjoint spanning trees.

**Question h:**

Describe, based on the observations above, a polynomial algorithm for deciding whether a given undirected graph $G = (V, E)$ contains $k$ edge-disjoint spanning trees (your algorithm does **not** have to find the trees if they exist!).

**Question i:**

Use the observations above to describe an algorithm $\mathcal{A}$ which given a graph $G = (V, E)$ that does not have $k$-edge-disjoint spanning trees, finds a minimum cardinality set $E'$ of new edges to add to $G$ so that the resulting graph $G' = (V, E \cup E')$ has $k$ edge-disjoint spanning trees. Hint: use minimum cost submodular flows on an orientation of $G$ plus some new edges.

[1] That is, there is some subpartition $\mathcal{F}$ of $V - r$ for which we have equality in (2).