Algorithmics Exercise 1

Example 1 Covering the 1's

The problem is solvable in polynminal time because it could be reduce to the smallest vertex cover of a biparte graph.
König-Egerváry Theorem
We can transform the problem to a minimum vertex cover problem for a biartite graph.
Notation: Columns as $C = \{C_1, ..., C_m\}$ and the rows as $R = \{R_1, ..., R_n\}$ and the edges $E = \{(R_i, C_j)$ where $a_{ij} = 1\}$
Becasue there are no edges between verteses in C and no edges betwenn vertices in R. So it is possible to cover all $1`s$ with a lines. A small example:

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix} \tag{1}$$

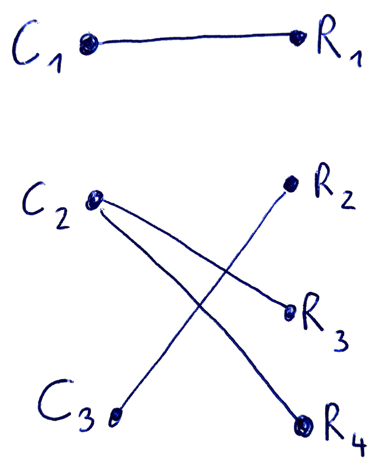

Figure 1: Graph

Example 2 Matrix rounding

This problen is similar to a Feasible matrix rounding (circulation problem with lower bounds). Given a $n \times m$ Matrix M where n is a even.

3,7	3,1	2,7	1,05	8,12
1,1	4,7	0,3	4,45	10,55
0,3	2,3	2,9	1,7	7,2
5,1	10,1	5,9	7,2	
15,2				

Figure 2: Feaseble Matrix

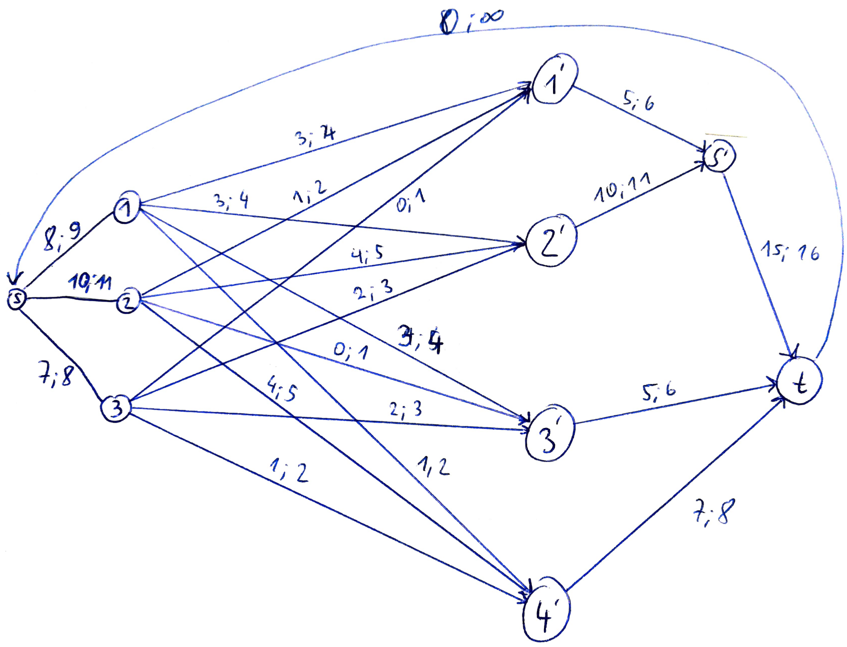

Figure 3: Graph

Example 3 Acyclic flows

a) acyclic flow:

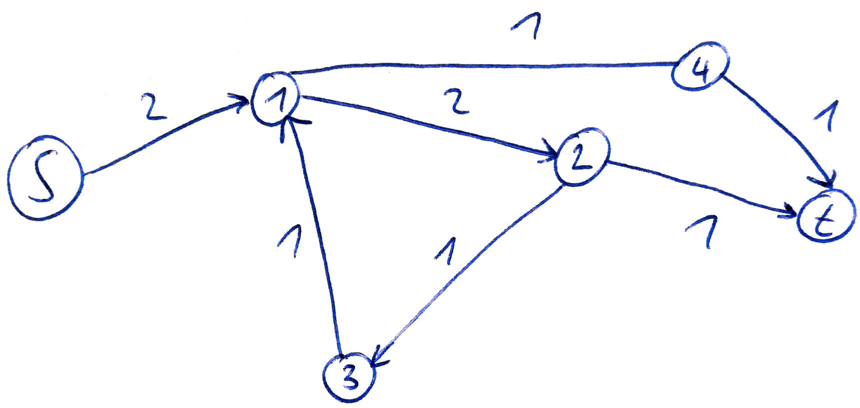

Figure 4: Graph

b)
Consider a maximum flow f, that is not acyclic.
Define E' the set of edges and V' the set of vertices in that cycle. So each vertices appears once and has one outgoing edge and one incoming edge in E'.
Denote $c' = \min\{f(e') \text{ where } e' \in E'\}$

$$f'(e) = f(e) - c' \;\; \forall e \in E'$$
$$f'(e) = f(e) - c' \;\; \forall e \in E \setminus E'$$

After that f' is still a flow because $\forall$ verteces in V':

$$\sum_{inv} f'(e) = \sum_{inv} f(e) - c' = \sum_{outv} f(e) - c' = \sum_{outv} f'(e)$$

Repeat this proced mutlible times to get an acycle flow.

Exercise 4 All-Different Assignments

We can transform this problem aslo in a Bipartit Matching problem. First we define L as a set of vertices, where each element represents a variable in X.
On the other hand we define R as a set of verteces, where each vertece represents on element in D.

$$(x_i, d_j) \in E \implies d_j \in D(x_i) \tag{2}$$

The maximum number of matchings are n. If there existst a all-different assignment $\rightarrow$ exists the following equivalence:

$$(x_i, d_j) \in M \implies f(x_i) = d_j \tag{3}$$

After that it is a Bipartite Matching Problem with the following Polynominal Time algorithm: To

1: Creat a graph G = (L ∪ R, E)
2: Write L with x_i
3: Write D with d_i
4: Wrtie $E(x_i, d_j)$ x_i and $d_j = D(x_i)$
5: Creat graph $G' = (L \cup R \cup \{s, t\}, E')$
6: Apply Ford-Fulkerson to G'
7: If max-flow is n $\rightarrow$ all-different assigmnent
8: Else no all-different

line 4 is a runtime of $O(mn)$
Creat the graph also in $O(mn)$ line 5
Ford-Fulkerson runs intotal with $O(m^2n^2)$
$m+n+2$ verteces and max $m \times n+2$. Finally we get $O(m^2n+n^2m+2mn+2m+2n+4) \rightarrow O(m^2n^2)$

Exercise 5 Significant Edges

A edge $e' = (u, v)$ of $N = \{V, E, c, s, t\}$ is a significant edge if the minimum cut of $N' = \{V, E, c', s, t\}$ where $c'(e') = c(e') - 1$ and $c'(e) = c(e)$ for all other edges than the minimum cut of N.

Explenarsion:
If the minimum cut of N' is less than 1 the minimum cut of N must contain $e' = (u, v)$ as on of the edges such that $u \in A$ and $v \in B$, as otherwise the minimum cut of N' would also be a minimum cut of N. Then the same cut with the capacity c instead of c' would be bigger by one, therefore the minimum cut of N' would also be a minimum cut of N thereby proving the implication.

With this assamtion we could design a algorythm who determine all significant edges (the maximum flow and the minimum cut have the same value)

Ford-Falkerson (V, E, ω, s, t) to determine maximal flow-value c
For $e \in E$ do:
$\omega(e) = \omega(e) - 1$
Ford-Fulkerson (V, E, ω, s, t) to determine new flow value c_n
If $(c_n < c)$ then $G = G \cup e$
$\omega(e) = \omega(e) + 1$
end for

Since this algorythm runs Ford-Fulkerson $m + 1$ times where m is the number of edges and Ford-Fulkerson is running in Plynominal time $O((m + 1)mnC)$

Exercise 6 Cute Subsets

This problem can be solved again by reducing it to a maximum flow problem.
1) Adding a source and a sink to the vertex-set
2) Add a vertex for each element in $s_i \in S$. 3) Add an edge from the source to each for these vertices with capacity 1.
4) Add a vertex for each set $A_1, ...A_m$.
5) Add edges in a way such that following equivalence holds:

$$\forall s_i \in S \ \forall A_i (s_i \in A_j \implies (s_i, A_j) \in E) \tag{4}$$

Assign capacity 1 to each for these edges.
6) Add an edge between each A_i and the sink with capacity 1.
If the maximum flow equals m then the edges in the flow give us a cut subset.

Ford-Fulkerson soves this maxflow-problem in $O((m+n)(mn) \times 1)$
$m+n$ verteces and $m \times n$ edges and the maximal capacity is 1.
This is obviosly also $O((m+n)^3)$ so it runs polynominally on $m+n$.

For the validation assume there is a maximum flow for m. We can get a cut subset from the flow in the following way:

$$\forall i = 1, ..., m \ a_i = s_j \implies f(s_j, A_i) = 1 \tag{5}$$

Since the flow to each s_i is exactly one such edge for each s_i. Because the flow from each A_i to the sink is limited by one, there must be one for s_j for each A_i, because else the maximum flow could not be m.

This equivalence holds in both directions and if we have a cut subset we can make a maximum flow of m using the above equivalence.

Exercise 7 Unique Flows

When we proof the following lemma then the probel is also solved: There is a unique maximum flow if changing the capacity of any of the edges, with $f(e) > 0$ in the maximum flow to $c(e) = f(e) - 1$ reduces teh maximum flow. (Only for acalcic graphs where capacities are positive integers).

Proof by contraposition:
If there is more than one maximum flow then changing the capacity of one of the edges $f(e) > 0$ to $c(e) = f(e) - 1$ does not reduce the maximum flow.
Let $f' \neq f$
Then we can take any e such that $f(e) > f'(e)$ and reduce it's capacity and still have hte maximum flow f' with the same value.
Obviously either an edge with $f(e) > f'(e)$ or an edge with $f'(e) > f(e)$ must exist because else the two flows would be the same. For all vertices:

$$\sum_{e into v} f(e) = \sum_{e out of v} f(e)$$

Assume taht no edge with $f(e) > f'(e)$ exists. Since the flows are bigger going into the vertex for f' than for f the flows going out must also be bigger. Since there are no cycles and $f'(e) \geq f(e)$ this would carr on until the sink, meaning that the flow f' has a bigger value than f contradiction our assumption that both are maximum flows. Therefor an e with $f(e) > f'(e)$ must exist.

Using the above lemma we can decide whether any given acyclic flow network has a unique maximum flow in the following way:

Algorithm 1 PPO

```
Ford-Fulkerson(N)
Let E' be the edges e such that f(e) > 0 in the computed maximum flow and c be the value of the flow
For e' ∈ E' do
c'(e') = f(e') − 1 and c'(e) = c(e) for all other
Compute Ford-Fulkerson(N' = (V, E, c', s, t))
if the maximum flow value is c output ("There are multiple maximum flows")
Elseif c'(e) = c(e) ∀e ∈ E
End for
"There is a unique maximum flow"
```

Since we call Ford-Fulkerson at most $m + 1$ times where m is the number of edges, the algorithm runs in polynominal time.

Exercise 8k-Edge Partitions

To proof this we will use the "Theorem of König": every k-regular bipartite gaph has a perfect matching.
Proof: Induction Basis: Case $k = 1$ is trivial.

Induction step:
Consider a $k + 1$-regular bipartite graph G. Form the theorm of König we know that there exists a perfect matching M for this graph.
Consider the graph $G' = (V, E \setminus M)$. Since we removed a perfect matching this is a k-regular bipartite graph and there exists a partition of the edges $E \setminus M$ into $E_1, ..., E_k$ such that each vertex is only incident to at most one edge of each partition. Simply let $E_{k+1} := M$ to get a $k+1$-partition as obviously no vertex can be incident to two edges in E_{k+1} as it is a perfect matsching.