Max Wisniewski
Paul Podlech

Dozent : Panos Giannopoulos

Exercise 1 : *Multicut Problem in Trees*

Problem and Definitions

Problem:
In this exercise we will solve the follwing problem.
Given a tree $T(V, E)$ and k pairs of vertices s_i, t_i and costs $c_e \geq 0$ for each edge $e \in E$. The task is to find a minimum-cost set of Edges $F \subset E$ such that for each pair s_i, t_i s_i is in a different component than t_i in the graph $G(V, E \setminus F)$.

To reformulate the last criterion, this means, there exists no path from s_i to t_i, or even better, from every path from s_i, t_i in T there has to be at least one edge in F.

With this we can formulate the IP

$$\begin{array}{rrcll} \min & \sum_{e \in E} c_e x_e & & & \\ \text{subject to} & \sum_{e \in P_i} x_e & \geq & 1, & 1 \leq i \leq k \\ & x_e & \in & \{0,1\} & e \in E. \end{array}$$

This matches exactly our definition, as formulated above.
In this IP P_i is the set of edges on the path from s_i to t_i (In a tree there exists exactly one path from one vertex to another) and x_e is the variable that denotes, whether e is in F or not.

Next we will formulate the Primal and the Dual LP with which we then will work.

Primal LP

$$\begin{array}{rrcll} \min & \sum_{e \in E} c_e x_e & & & \\ \text{subject to} & \sum_{e \in P_i} x_e & \geq & 1, & 1 \leq i \leq k \\ & x_e & \geq & 0 & e \in E. \end{array}$$

Dual LP

$$\begin{array}{rrcll} \max & \sum_{i=1}^{k} y_i & & & \\ \text{subject to} & \sum_{i: e \in P_i} y_i & \leq & c_e, & e \in E \\ & y_i & \geq & 0 & e \in E. \end{array}$$

As one can see and assume from the simple Cut Problem, the dual is a Multiflow, such that on no edge there is more flow, than the capacity of the edge.

Root, Height and Ancestor
In the following we assume, that the tree is rooted at a arbitrary vertex r. From this

root node we can define the $depth(v)$ of a node $v \in E$ by the number of edges from r to v. The depth can be computed for every node by a simple iteration over the tree with the runtime of $O(n)$.

Next we will donate the *Lowest-Common-Ancestor* $lca(s_i, t_i)$ as the vertex $v = \underset{u \in P_i}{\text{argmin}}\{depth(u)\}$. This is namly the vertex on the path that is nearest to the root. This nodes can be found will computing the depth without increasing the runtime.

The Algorithm

We will solve this problem by applying the Primal-Dual-Method.

The algorithm will work as follows:
We start with an empty set of edges. Next we will iterate over all $lca(s_i, t_i)$ for every $1 \leq i \leq k$ from the one with the highest depth to the lowest one. Next we try to increment the flow y_i of the given vertex $lca(s_i, t_i)$ we took, until some constraint is met with equality. This edge we will then take into the solution set. In the last step we will remove, this time from the root to the bottom, every edge $e \in F$, such that $F \setminus \{e\}$ stays feasible.

Let $\mathcal{A}$ be the following algorithm

```
F ← ∅
I ← {1,..,k}
y_i ← 0   ∀i ∈ I
t ← 0
WHILE I ≠ ∅ connected DO
    t ← k + 1
    i ← argmax_{j∈I} lca(s_j, t_j)

    e_t ← argmin_{e_t∈P_i} {c_{e_t} − Σ_{j:e_t∈P_j} y_j}

    Δ ← c_{e_t} − Σ_{j:e_t∈P_j} y_j

    f_i ← f_i + Δ
    I ← I \ {j | e_t ∈ P_j}
    F ← F ∪ {e_t}

    R ← F
FOR z FROM k DOWNTO 1 DO
    IF R \ {e_z} is feasable THEN
        R ← R \ {e_z}
RETURN R
```

The Proof

In this part we will show, that the follwoing holds

Claim 1. *The algorithm $\mathcal{A}$ has a runtime in P.*

Claim 2. *The algorithm $\mathcal{A}$ is a 2-approximation-algorithm for the multicut problem in trees.*

Proof 1:

We see, that we have at most k iterations of the while loop, because we cut every iteration at least one pair.
In each iteration we first search the maximal depth, which we can obtain in $k \log k$ overall iterations (MaxHeap).
We can optain the e_t in P time. The path from s_i to t_i is for every pair at most $n = |V|$ (-1). This is the amount of elements we have to search for the minimum next.
For each of these n elements we have to find all Path P_j that contains the observed edge. This can be done in ineffciently in $k \cdot n$ for every edge (k Paths maximal and n elements in each). So we find the e_t in $O(n^2 k)$. Δ can be found in less time as easily seen. The rest of the actions needs constant time, or at least less than the given.
So the first loop has runtime $O(n^2 k^2)$. The next loop checks, for pair, whether we can delete one of the at most k taken edges.
We can loop at each path in $O(n)$ of k paths and this at most k times which leads us to $O(nk^2)$.
So this algorithm runs in $O(n^2 k^2)$ which obviosly lays in P.

Proof 2:

At first we observe, that F is feasable.
In line 5 of the algorithm, the loop terminates, only if there exists no pair of vertices in the set I.
In the loop body we remove an index only if we met one edge with equality in the dual. Due to complemantry slagness rule we know, that we have the edge in the Primal and so we removed an edge from the unique way from a the pair i.
We conclude, that the set F was feasable after the first loop. The seconde loop, will only remove an edge, if the set still remains feasable.
So the resulting set R must be feasable.

To proof the approximation we will show the following

Claim 3. *For any pair $y_i > 0$ $1 \leq i \leq k$ $|F \cap P_i| \leq 2$ that is for any non-zero flow there are at most 2 edges in F on the flow-path.*

Claim 4. *For F $\sum_{e \in F} c_e \leq 2OPT$ holds.*

Proof 3: Suppose y_i set to nonzero while we consider $v := lca(s_i, t_i)$. Therefore no edge of P_i has been cut before. Any P_j $j \neq i$ considered afterwards has its $lca(s_j, t_j)$ in the same depth or lower than v. Therefore if P_j shares an edge with P_i $lca(s_j, t_j)$ is an ancestor of v. Thus if we cut a new edge for P_j which is on P_i the Algorithm removes the lower edges, since they are not neccessary for any previous (s_k, t_k) the upper one could not be removed, because it will leave the P_j without a cut and (s_i, t_i) will be cut by the new edge too. Hence follows the claim.

Proof 4:

$$\sum_{e \in F} = \sum_{e \in F} \sum_{i: e \in P_i} y_i = \sum_{1 \leq i \leq k} |P_i \cap F| y_i \leq 2 \sum_{1 \leq i \leq k} y_i \leq 2OPT$$

This follows by construction, Claim 3 and duality.