

CSCI 2510 - Problem set 8

Shay Mozes
shay@cs.brown.edu

Problem 18.3

We give a polynomial time algorithm for the integral multicommodity flow on unit capacity trees of arbitrary height.

Let T be an (arbitrarily rooted) tree T with k pairs of terminal nodes s_i, t_i for distinct commodities ($1 \leq i \leq k$) ($s_i \neq t_i$). Let T_v denote the subtree of T rooted at node v . We define a recursive procedure $\text{solve}(T_v)$ that returns f_v , the maximum value of an integral multicommodity flow within T_v , along with a set of commodities C_v , such that for each commodity $i \in C_v$, exactly one of s_i, t_i belongs to T_v and there exist a feasible integral multicommodity flow in T_v with value f_v in which commodity i is routed from s_i (or t_i) to v . In other words, i is such that commodity i can be sent out of T_v while still attaining the maximum flow value within T_v .

If v is a leaf then $\text{solve}(T_v)$ returns zero and C_v is the empty set unless v is one of the terminal nodes of commodity i , in which case $C_v = \{i\}$. For an internal node v , $\text{solve}(T_v)$ is computed as follows.

Let v_1, \dots, v_n be the children of v in T .

- For each $1 \leq j \leq n$ recursively compute f_{v_j} and C_{v_j} .
- If v is a terminal for commodity i , set C_v to $\{i\}$.
- Define G to be the graph on nodes v, v_1, \dots, v_n , where there is an edge between two nodes u, w if $C_u \cap C_w$ is non-empty. In polynomial time, find a maximum matching M in G . Set f_v to $|M| + \sum_j f_{v_j}$
- In a similar manner, for each $1 \leq j \leq n$ define G_j to be the graph on nodes $v, v_1, \dots, v_{j-1}, v_{j+1}, \dots, v_n$ where there is an edge between two nodes u, w if $C_u \cap C_w$ is non-empty. In polynomial time, find a maximum matching M_j in G_j . Let ϕ_j be $|M_j| + \sum_j f_{v_j}$, and let γ_j be the set $\{i \in C_{v_j} \text{ s.t. exactly one of } s_i, t_i \text{ belongs to } T_v\}$.
- Set C_v to be the union of γ_j for all j such that $\phi_j = f_v$. If v is a terminal for commodity i that is not paired within T_v , add i to C_v as well.

running time analysis: Since the recursion is invoked exactly once for each node, and since each level of the recursion takes polynomial time, the overall running time is polynomial as well.

correctness: Note that since the direction of the flow does not affect its feasibility, there is no real reason to distinguish the source terminals from the sink terminals, we therefore use s_i and t_i rather freely in our proof. We prove the correctness of the procedure by induction on the height of the tree. The base case is correct since for a single leaf node v , the flow is zero and the construction of C_v is trivial. We next assume correctness for the children $v_1 \dots v_n$ of an internal node v , and show that f_v and C_v are computed correctly.

We first show that there exists a feasible flow with value f_v . Let $e = (v_j, v_{j'})$ be an edge in the matching M , and let i be the corresponding commodity in the non empty intersection of C_{v_j} and $C_{v_{j'}}$. By the inductive assumption there exist feasible flows for $T_{v_j}, T_{v_{j'}}$ with values $f_{v_j}, f_{v_{j'}}$, respectively. Moreover, a flow of commodity i from s_i in T_{v_j} through v_j then to $v_{j'}$ and on to t_i in $T_{v_{j'}}$ is feasible. Thus we get a flow of value $f_{v_j} + f_{v_{j'}} + 1$. Summing over all edges in the matching, and adding all flows for non-matched subtrees we get a feasible flow of value exactly $|M| + \sum_j f_{v_j} = f_v$. The case where v is part of the matching has to be handled separately, but follows by a similar argument.

Next we argue that f_v is at least as large as the maximum flow OPT. Assume the contrary and consider the children of v . If the flow within each T_{v_j} in OPT is equal to f_{v_j} , then, by the inductive hypothesis, both OPT and our flow can only direct one of the commodities in C_{v_j} through the edge (v_j, v) . Since our algorithm considers a maximal matching corresponding to these commodities, OPT cannot be better than the solution obtained by our procedure in this case. Assume, therefore that the flow in OPT within T_{v_j} is strictly smaller than f_{v_j} for some j . Now OPT may gain by directing some commodity not in C_{v_j} from v_j to v , a possibility that our procedure does not consider. However, this contributes at most one more unit of flow to OPT which cancels out with the suboptimal flow within T_{v_j} . Thus f_v is at least as large as OPT.

Finally we have to show that the set C_v is constructed properly. Assume that there exists a commodity i such that $s_i \in T_{v_j}, t_i \notin T_v$ which can be routed to v while maintaining a flow of f_v in T_v . There are three cases:

- s_i is in some T_{v_j} and $i \in C_{v_j}$. In this case either $\phi_j = f_v$, in which case i is included in C_v by our procedure, or $\phi_j < f_v$, which contradicts the assumption that i can be routed to v while maintaining a flow of f_v .
- s_i is in some T_{v_j} and $i \notin C_{v_j}$. This means, by the inductive hypothesis, that i cannot be routed to v_j while maintaining a flow of f_{v_j} in T_{v_j} . We construct a flow within T_v with value strictly greater than f_v by replacing the suboptimal flow within T_{v_j} with the maximum flow that has value f_{v_j} . The flow remains feasible since we only change the flow within T_{v_j} and no longer route any commodity from v_j to v . It does not reduce the flow outside T_{v_j} since i does not have both terminal nodes in T_v , so it cannot contribute to the flow in T_v . We obtained a feasible flow with value greater than f_v , contradicting the optimality of f_v .
- v itself is a terminal node for commodity i . In this case i is added to C_v explicitly by our procedure.

□