Advanced Graph Algorithms and Optimization Spring 2024

Classical Algorithms for Maximum Flow

R. Kyng & M. Probst Gutenberg *Problem Set 7 — Monday, April 15th*

The exercises for this week will not count toward your grade, but you are highly encouraged to solve them all. This exercise sheet has exercises related to week 7. We encourage you to start early so you have time to go through everything.

To get feedback, you must hand in your solutions by 23:59 on April 25. Both hand-written and LaTeX solutions are acceptable, but we will only attempt to read legible text.

Exercise 1: Implementing Field Preservations for Cut-Link Tree Rotations

In Chapter 12, Section 12.3, we described the operations $\text{PLink}(u, v)$ and $\text{PCut}(u, v)$ by doing $O(\log n)$ tree rotations (in expectation). However, we omitted the details. Here we ask you to give the pseudo-code for the operation $\text{PTreeRotation}(v, w)$ where it is assumed that on input v is the parent of w and the operation manipulates the tree over the path such that v and w change position as described in the script. For simplicity, you are allowed to assume that w is the left child of v in the treap $\mathcal{T}_p$, and that the nodes $\mathit{left}_{\mathcal{T}_p}(v), \mathit{right}_{\mathcal{T}_p}(v), \mathit{left}_{\mathcal{T}_p}(w), \mathit{right}_{\mathcal{T}_p}(w)$ exist. Accompany your pseudo-code with a brief analysis that confirms that the run-time is indeed $O(1)$.

Solution. The pseudocode is given in algorithm 1. Clearly, all updates in the algorithm (changing pointers and updating $\Delta \mathit{min}$ and $\Delta \mathit{cost}$ values can be done in $O(1)$ hence the total runtime is $O(1)$.

Exercise 2: Blocking Flows on Expander Graphs

Some classical algorithms for maximum flow have the unexpected behaviour that they often terminate much faster than their worst-case guarantee would suggest. One such algorithm is the blocking flow algorithm. The goal of this exercise is to show that expander graphs are one class of graphs for which blocking flow works particularly well.

In this exercise we develop an approximate s-t maximum flow algorithm for a ϕ-expander graph $G = (V, E)$ with n vertices and m edges. Each edge has unit capacity. Show the following.

1. Assume that the amount of s-t demand D is at most $1 - \epsilon$ of the maximum flow, i.e. the s-t min-cut is at least $1/(1 - \epsilon)D$. Show that $h = O(\frac{\log m}{\phi\epsilon})$ to send D flow from s to t in that case.

Solution. Recall that running h iterations of blocking flow guarantees that the distance from s to t in the residual graph is at least $h+1$ if not all the demand has been routed by Lemma 11.3.1 in the script . Let $\boldsymbol{f}$ denote the (integral) flow computed after h iterations of blocking flow. Assume for a contradiction that $\boldsymbol{f}$ does not route D flow.

Algorithm 1: PTreeRotation(v,w)

```
/* Recall we assume $left_{\mathcal{T}_p}(v) = w$; and $w$ and $v$ both have two children.   */
/* We first change the parent pointer for the parent that supports the subtree
   (if there is one).                                                                           */
if $parent_{\mathcal{T}_p}(v) \neq NULL$ then
    $p \leftarrow parent_{\mathcal{T}_p}(v)$
    if $left_{\mathcal{T}_p}(p) = v$ then
        $left_{\mathcal{T}_p}(p) \leftarrow w$
    end
    else
        $right_{\mathcal{T}_p}(p) \leftarrow w$
    end
end
/* Compute roots of the subtrees rooted at the children of $v$ and $w$.                     */
$a \leftarrow left_{\mathcal{T}_p}(w)$.
$b \leftarrow right_{\mathcal{T}_p}(w)$.
$c \leftarrow right_{\mathcal{T}_p}(v)$.
/* Change the tree structure according to the rotation.                                      */
$right_{\mathcal{T}_p}(w) \leftarrow v$.
$left_{\mathcal{T}_p}(v) \leftarrow b$.
/* Update $\Delta cost()$ and $\Delta min()$ to new variables                                */
$\Delta c(w) \leftarrow cost(v) - cost(v) + cost(w)$.
$\Delta c(v) \leftarrow \Delta cost(v) + \min\{\Delta min(b) + \Delta min(w), \Delta min(c)\}$
$\Delta m(w) \leftarrow \Delta min(v)$
$\Delta m(v) \leftarrow \Delta min(v) + \min\{\Delta min(c), \Delta min(w) + \Delta min(b)\}$.
$\Delta m(a) \leftarrow \Delta min(a) - \Delta min(w)$.
$\Delta m(b) \leftarrow \Delta min(b) - \Delta min(w) + \Delta m(v)$.
$\Delta m(c) \leftarrow \Delta min(c) - \Delta m(v)$.
/* Update actual fields                                                                      */
$\Delta cost(w) \leftarrow \Delta c(w)$
$\Delta cost(v) \leftarrow \Delta c(v)$
$\Delta min(w) \leftarrow \Delta m(w)$
$\Delta min(v) \leftarrow \Delta m(v)$
$\Delta min(w) \leftarrow \Delta m(w)$
$\Delta min(w) \leftarrow \Delta m(w)$
$\Delta min(w) \leftarrow \Delta m(w)$
```

Consider the following algorithm. Initialize $S_0 = \{s\}$. Then update $S + i + 1 \leftarrow S_i \cup \{v | \exists u \in S_i \text{ s.t. } (u, v) \in G_{\boldsymbol{f}}\}$ for h iterations $i = 0, \ldots, (h-1)/2$. Notice that the final set S does not contain t since we only ran the algorithm for h steps. Now consider the volume of S_i in G as a function of the volume of S_{i-1}. Clearly $\mathrm{vol}_G(S_0) \geq 1$, otherwise there would not exist any flow routing out of s. Then, we know that the cut $E_G(S_i, V \setminus S_i)$ contains at least $\phi\,\mathrm{vol}_G(S_i)$ edges as long as $\mathrm{vol}_G(S_i) \leq \mathrm{vol}_G(V)/2$.

We now argue that a large fraction of these edges are also contained in the residual graph. An edge is not in the residual graph if it routes out flow. This can happen for two reasons: 1) the flow originated at s or 2) the flow got routed across the cut in opposite direction first. For every edge of type two there is a paired edge that is in the cut, and therefore there are at least $\frac{\phi}{2}\,\mathrm{vol}_G\, S_i$ edges also in the residual graph. Therefore the volume of set S_i is at least $\min((1+\phi\epsilon/2)^i, 1+\mathrm{vol}_G(V)/2)$. Therefore after $(h-1)/2 = O(\log m/\phi\epsilon)$ iterations the set S_i contains half the graph. The symmetric argument from t yields a contradiction to the assumption that the distance between s and t is at least $h+1$. This concludes our proof.