Overview: Shortest Augmenting Paths

Lemma 54

The length of the shortest augmenting path never decreases.

Lemma 55

After at most O(m) augmentations, the length of the shortest augmenting path strictly increases.



14. Jan. 2024 440/450

Shortest Augmenting Paths

Define the level $\ell(v)$ of a node as the length of the shortest *s-v* path in G_f (along non-zero edges).

Let L_G denote the subgraph of the residual graph G_f that contains only those edges (u, v) with $\ell(v) = \ell(u) + 1$.

A path P is a shortest s-u path in G_f iff it is an s-u path in L_G .



edge of L_G

cage o. 26

Overview: Shortest Augmenting Paths

These two lemmas give the following theorem:

Theorem 56

The shortest augmenting path algorithm performs at most O(mn) augmentations. This gives a running time of $O(m^2n)$.

Proof.

- We can find the shortest augmenting paths in time $\mathcal{O}(m)$ via BFS.
- $ightharpoonup \mathcal{O}(m)$ augmentations for paths of exactly k < n edges.

Harald Räcke

11.2 Shortest Augmenting Paths

14. Jan. 2024

441/450

In the following we assume that the residual graph \mathcal{G}_f does not contain zero capacity edges.

This means, we construct it in the usual sense and then delete edges of zero capacity.

Shortest Augmenting Path

First Lemma:

The length of the shortest augmenting path never decreases.

After an augmentation G_f changes as follows:

- ▶ Bottleneck edges on the chosen path are deleted.
- ► Back edges are added to all edges that don't have back edges so far.

These changes cannot decrease the distance between s and t.



Shortest Augmenting Paths

Theorem 57

The shortest augmenting path algorithm performs at most $\mathcal{O}(mn)$ augmentations. Each augmentation can be performed in time $\mathcal{O}(m)$.

Theorem 58 (without proof)

There exist networks with $m=\Theta(n^2)$ that require $\Omega(mn)$ augmentations, when we restrict ourselves to only augment along shortest augmenting paths.

Note:

There always exists a set of m augmentations that gives a maximum flow (why?).

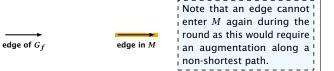
Shortest Augmenting Path

Second Lemma: After at most m augmentations the length of the shortest augmenting path strictly increases.

Let M denote the set of edges in graph L_G at the beginning of a round when the distance between s and t is k.

An s-t path in G_f that uses edges not in M has length larger than k, even when using edges added to G_f during the round.

In each augmentation an edge is deleted from M.



Shortest Augmenting Paths

When sticking to shortest augmenting paths we cannot improve (asymptotically) on the number of augmentations.

However, we can improve the running time to $\mathcal{O}(mn^2)$ by improving the running time for finding an augmenting path (currently we assume $\mathcal{O}(m)$ per augmentation for this).

Shortest Augmenting Paths

We maintain a subset M of the edges of G_f with the guarantee that a shortest s-t path using only edges from M is a shortest augmenting path.

With each augmentation some edges are deleted from M.

When M does not contain an s-t path anymore the distance between s and t strictly increases.

Note that *M* is not the set of edges of the level graph but a subset of level-graph edges.



11.2 Shortest Augmenting Paths

448/450

Analysis

Let a phase of the algorithm be defined by the time between two augmentations during which the distance between s and t strictly increases.

Initializing M for the phase takes time $\mathcal{O}(m)$.

The total cost for searching for augmenting paths during a phase is at most O(mn), since every search (successful (i.e., reaching t) or unsuccessful) decreases the number of edges in M and takes time $\mathcal{O}(n)$.

The total cost for performing an augmentation during a phase is only $\mathcal{O}(n)$. For every edge in the augmenting path one has to update the residual graph G_f and has to check whether the edge is still in M for the next search.

There are at most n phases. Hence, total cost is $\mathcal{O}(mn^2)$.

Suppose that the initial distance between s and t in G_f is k.

M is initialized as the level graph L_G .

Perform a DFS search to find a path from s to t using edges from M.

Either you find t after at most n steps, or you end at a node vthat does not have any outgoing edges.

You can delete incoming edges of v from M.

וחו	ПП	Harald Räcke
	┚╚┖	Harald Räcke

11.2 Shortest Augmenting Paths

449/450