Flows, paths and cuts

Tomáš Turek

9. 10. 2023

1. Introduction

1.1 Basics

Firstly we remind some basics from flows and cuts. **Flow** is defined on a **network**. Network is on a oriented graph $G = (V,E)$ and it has two special vertices $s,t \in V$ called source and target. Also we have a capacity, which is a mapping $c : E \to \mathbb{R}_0^+$. Flow then is a mapping $f : E \to \mathbb{R}_0^+$ which has two properties.

1. $\forall e \in E : f(e) \leq c(e)$
2. Kirchoff's law: $\forall v \neq s,t \in V : \sum_{uv \in E} f(uv) - \sum_{vu \in E} f(vu) = 0$

Now we also remind ourselves another term which is a s,t-cut. Which is a $M \subset E$ such that no s,t-path exists in $G' = (V, E \setminus M)$.

These basic terms can be generalized to a **multicommodity flow** problem and **multicut** problem.

1.2 Multi commodity problem

On a graph $G = (V,E)$ and k tuples $(s_1,t_1),(s_2,t_2),\ldots,(s_k,t_k)$ inV^2 known as **commodities** we can also define a network and a flow. Same as before we have a capacity $c : E \to \mathbb{R}_0^+$. These commodities are shared on the same resources.

We may define $\mathcal{P}_i$ as all paths between s_i and t_i. And also $\mathcal{P} := \bigcup_{i=1}^{k} \mathcal{P}_i$.

For a single commodity flow problem we could define an *linear program* (LP):

$$\begin{array}{rl} & \max \sum_{p \in \mathcal{P}_{st}} f_p \\ \forall e \in E & \sum_{p:e \in p \in \mathcal{P}_{st}} f_p \leq c(e) \\ & f \geq 0 \end{array}$$

From this we may define an linear program denoted as **LP1** for multi commodity problem.

$$\begin{array}{rl} & \max \sum_{p \in \mathcal{P}} f_p = \sum_{i=1}^{k} \sum_{p \in \mathcal{P}_i} f_p \\ \forall e \in E & \sum_{j=1}^{k} \sum_{p:e \in p \in \mathcal{P}_j} f_p \leq c(e) \\ & f \geq 0 \end{array}$$

Alternatively we could define it by a single variables for flows on each edge. For multi commodity problem we would have k flows which adds up to one single flow. Thus we can see that the problem is in P (polynomial).

1.3 Multi cut problem

As a cuts for single flow we may define somewhat similiar definition. Such that between every tuple of s_i and t_i there is no such path. We will also look at a linear program which will be denoted as **LP2**.

$$\forall e : x_e = \begin{cases} 1 & e \text{ in cut} \\ 0 & \text{otherwise} \end{cases}$$

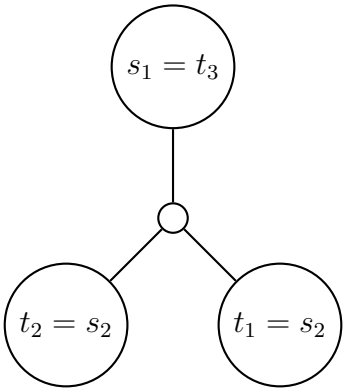
$$\begin{aligned} \phi := & \quad \min \sum_{e\in E} x(e)c(e) \\ \forall i \in [k] \forall p \in \mathcal{P}_i & \quad \sum_{e\in p\in\mathcal{P}_i} x(e) \geq 1 \\ \text{(ILP)} & \quad x(e) \in \{0,1\} \\ \text{(LP)} & \quad x(e) \geq 0 \end{aligned}$$

First ILP is integer linear program which is generally NP hard. Thus we will look on the relaxation of LP program. We may observe that we don't need to specify that $x(e) \geq 1$.

Now we may see that indeed **LP1** and **LP2** are dual programs. Thus resulting in knowing that the maximum flow is the same as minimum fractional cut.

1.3.1 Example

Before we continue we will take a look at a simple example. The graph is as follows. All capacities are equal to 1.



We may observe that the max flow is $|f| = \frac{3}{2}$ because we can put $\frac{1}{2}$ on every edge. Then multi-cut is 2 since we need to remove always at least two edges. But minimum fractional multi-cut is only $\frac{3}{2}$ because we only have to "cut"half of each edge. Thus it is the exact same result as max flow.

1.3.2 Algorithm

We will show an algorithm which will give an approximate result of a multi-cut problem. We may look at it like we would cut off some parts of the graph which are close to the sources and continue.

Given $\bar{G} = (\bar{V}, \bar{E}), (s_i, t_i), k, c : E \to \mathbb{R}_0^+$ and solution x of **LP2**. We define:

$$B_x(s_i, r) := \{u \in \bar{V} | d_x(s_i, u) \leq r\}$$

As a ball around s_i with diameter w.r.t x. $d_x(s_i, u) :=$ the length of the shortest $s_i u$ path in $\bar{G}$ w.r.t. the edge length $x(e)$.

$$\delta(B_x(s_i, r)) = \{\{u,v\} \in \bar{E} : |\{u,v\} \cap B_x(s_i, r)| = 1\}$$

$$V_x(s_i, r) = \frac{\phi}{k} + \sum_{\{u,v\}\in\bar{E}, u,v\in B_x(s_i,r)} c(e)x(e) + \sum_{\{u,v\}\in\delta(B_x(s_i,r)):u\in B_x(s_i,r)} c(e)(r - d_x(s_i, u))$$

This we call a volume of a ball. We will denoted as a function of r $f(r)$. The last sum is of all edges which are only partly inside the ball. Next we also define the following.

$$C_x(s_i, r) = \sum_{\{u,v\} \in \delta(B_x(s_i,r))} c(u,v)$$

This will be denoted as a function $g(r)$. We may see some really nice properties these functions have. For instance $f(r)$ is a growing function which is increasing linearly and then it jumps to another point. On the other hand $g(r)$ is constant at some parts and then it jumps to a certain point, these jumps are for both function in the exact same spots. Next we can see that for some nice points it holds that $f'(r) = g(r)$.

Lemma 1. *For each $i \in [n]$ s.t. $s_i, t_i \in \bar{V}$ there exist $r \in (0, 1/2)$ s.t.*

$$\frac{C_x(s_i, r)}{V_x(s_i,r)} \leq 2 \ln 2k.$$

Proof. By contradiction. For fixed i $\forall r \in (0, 1/2)$, $\frac{f'(r)}{f(r)} > 2 \ln 2k$. We will have $r_0 = 0 < r_1 < r_2 < \cdots < r_{l-1} < 1/2 = r_l$ which are the values where $f(r)$ is not continuous (there are these "jumps"). First we consider (r_j, r_{j+1}) for some j.

$$\frac{f'(r)}{f(r)} = (\ln f(r))'$$

So $\forall r \in (r_j, r_{j+1})$, $(\ln f(r))' > 2 \ln 2k$. We will compute the integral over all of these values. We may see that the right side is just a constant so we get

$$\int_{r_j}^{r_{j+1}} (\ln f(r))' > (r_{j+1} - r_j) 2 \ln 2k.$$

In r_{j+1} there may be jump so we instead take the $\lim_{r \to r_{j+1}^-} f(r) = f^-(r_{j+1})$. Note that $f^-(r_{j+1}) \leq f(r_{j+1})$.

$$\ln f(r_{j+1}) - \ln f(r_j) \geq \ln f^-(r_{j+1}) - \ln f(r_j) = \int_{r_j}^{r_{j+1}} (\ln f(r))' > (r_{j+1} - r_j) 2 \ln 2k.$$

Now we sum our inequality over all intervals $(r_j, r_{j+1}), j = 0, \ldots, l$. We will only have the very ends because the rest will be once added and once removed.

$$\sum_{j=0}^{l-1} (\ln f(r_{j+1}) - \ln f(r_j)) > \sum_{j=0}^{l-1} (r_{j+1} - r_j) 2 \ln 2k$$

$$\ln f(r_l) - \ln f(r_0) > (r_l - r_0) 2 \ln 2k$$

$$\ln f(1/2) - \ln(0) > \ln 2k$$

$$\ln \frac{f(1/2)}{\frac{\Phi}{k}} > \ln 2k$$

$$\frac{f(1/2)}{\frac{\Phi}{k}} > 2k$$

$$f(1/2) > 2\Phi$$

As the volume of the entire pipe system is at most 2Φ it means that we have a contradiction.

□

Note that choosing $1/2$ is not necessary for the proof, but for the algorithm to work. Because if we choose $1/2$ it means that no s_j, t_j will both be in a ball for the index i. That is because the length w.r.t x of paths from s_j to t_j need to be at least 1.

1.4 Pipe cut algorithm

INPUT: $\bar{G} = (\bar{V}, \bar{E})$.
OUTPUT: F multi cut.

$F \leftarrow \emptyset$
for $i = 1 \dots k$ **do**
  if $s_i - t_i$ are still connected in $(\bar{V}, \bar{E} \setminus F)$ **then**
    Choose $r \in (0, 1/2)$ by Lemma.
    $F \leftarrow F \cup \delta(B_x(s_i, r))$
    Remove edges inside $B_x(s_i, r)$ and $\delta(B_x(s_i, r))$.
  end if
end for
return F

There are few things to talk about. To get r we will check all "almost ends"of all intervals. The time complexity is polynomial since everything that is inside the code is polynomial. Correctness of the algorithm is easily observable since no pair p_i,t_i is inside some other ball and all balls will separate pairs s_j, t_j. Other thing to consider is what is the approximation ratio?

Theorem 1. *Approximation ratio of the Pipe cut algorithm is* $O(\log k)$.

Proof. Not yet done.

□