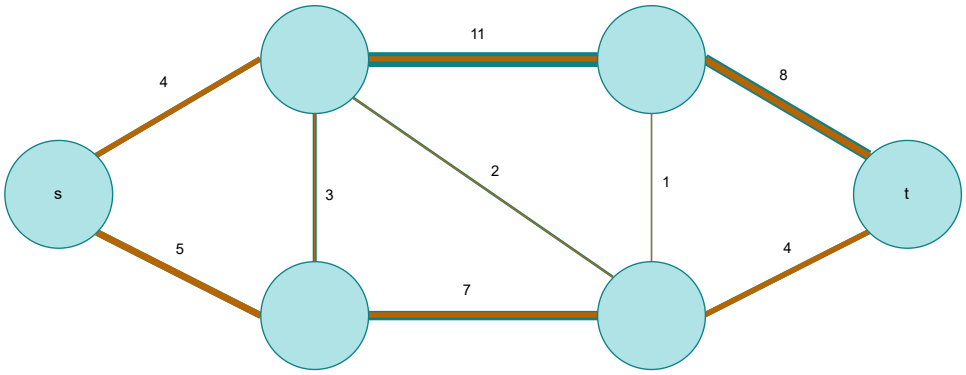
Flows, paths and cuts

Tomáš Turek



December 1, 2023

Contents

1 Introduction **2**
1.1 Network flow 2
1.2 Min s,t-cut 2

2 Multi-commodity problem **3**
2.1 Multi cut problem 3
2.2 Example 3
2.3 Preparation for algorithm 4
2.4 Pipe cut algorithm 6
2.5 How to solve LP 6
2.6 Is there any better approximation? 7

3 The Sparsest cut problem **8**
3.1 Concurrent multicommodity flow 8
3.2 Metric spaces 14
3.3 Applications 16
3.4 Minimum cut linear arrangement 17

4 Max cut **18**

5 Edge disjoint path problem **21**

1. Introduction

Firstly we remind some basics from flows and cuts.

1.1 Network flow

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: $\forall v \neq s,t \in V : \sum_{uv \in E} f(uv) - \sum_{vu \in E} f(vu) = 0$

1.2 Min s,t-cut

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 **multi-commodity flow** problem and **multi-cut** problem.

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):

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

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

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

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).

2.1 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**.

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

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

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.

2.2 Example

Before we continue we will take a look at a simple example (2.1). 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

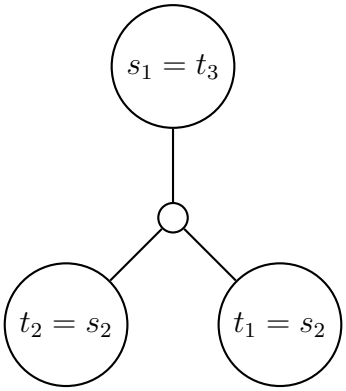


Figure 2.1: Example graph.

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.

2.3 Preparation for 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.

Algorithm 1 Pipe cut algorithm

Require: $\bar{G} = (\bar{V}, \bar{E})$.
Ensure: F multi cut.
1: $F \leftarrow \emptyset$
2: **for** $i = 1 \dots k$ **do**
3: **if** $s_i - t_i$ are still connected in $(\bar{V}, \bar{E} \setminus F)$ **then**
4: Choose $r \in (0, 1/2)$ by Lemma.
5: $F \leftarrow F \cup \delta(B_x(s_i, r))$
6: Remove edges inside $B_x(s_i, r)$ and $\delta(B_x(s_i, r))$.
7: **end if**
8: **end for**
9: **return** F

2.4 Pipe cut algorithm

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. Lets define C_i as the cost of the cut of the ball from iteration i and V_i as the volume of it. We know that $C_i \leq 2 \ln 2k \cdot V_i$.

$$\sum_i C_i \leq 2 \ln 2k \sum_i V_i \leq 2 \ln 2k \cdot 2\Phi = 4 \ln 2k \cdot \Phi = O(\log k)\Phi$$

□

For a single commodity we know that max flow = min cut. Where $\leq$ is trivial and $\geq$ is a little harder. This is a case of **exact duality**. On the other hand we already shown that this doesn't hold for multi-commodity, but what if we can define **approximate duality**.

Corollary. Max flow $\leq$ min cut $\leq O(\log k)$ max flow. For multi-commodity.

Proof. Because of the duality of LP1 and LP2 we know that max flow is the same as min fractional multi-cut. And because of the algorithm we know that the fractional multi-cut is in $O(\log k)$.

□

2.5 How to solve LP

There is still a problem with our LP which can have up to exponential many of constraints. But this can be solved fast by using **Ellipsoid algorithm** on $Ax \leq b$. Only think it needs is an so called **ORACLE** which is that for given $\bar{x}$, check whether $A\bar{x} \leq b$ and if not return a violated constraint.

In our case **ORACLE** is for each i find the shortest $s_i - t_i$ path w.r.t $\bar{x}$. This can be either 1 and we are happy or < 1 then this constraint is violated.

2.6 Is there any better approximation?

We will show that indeed this approximation is the best we can get. Firstly we will define a new property of graphs.

A graph $G = (V,E)$ is an **α-expander** if $\forall S \subseteq V, |S| \leq \frac{n}{2}, \delta(S) \leq \alpha|S|$.

We take as granted that it holds: 3-regular α-expanders exist for $\alpha > 0$. Now lets observe (2.2)that at most $1 + 3 \cdot 2^{l-1}$ vertices are reachable by a path of length $\leq l$.

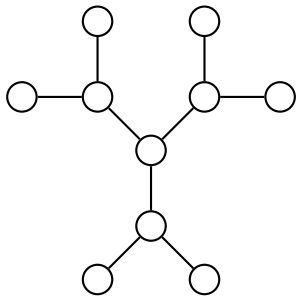


Figure 2.2: How to get the upper bound.

If we take $l = \log_2 \frac{n-2}{6} + 1$ so with the upper bound we get $1 + 3\frac{n-2}{6} = 1 + \frac{n-2}{2} = \frac{n}{2}$. And also we define an instance of multi-commodity problem: $T = \{\{u,v\}|\delta(u,v) > l\}$. A unit of flow consumes $\geq l$ units of volume of the entire system. Thus $|E| = O(n)$. Therefore max flow $\leq O(\frac{n}{l}) = O(\frac{n}{\log n})$. But for min cut we take the optimum $F \subseteq E$. Every path in $G = (V, E \setminus F)$ is $\leq \frac{n}{2}$ so min cut os $\Theta(n)$. Thus it is indeed tight.

3. The Sparsest cut problem

Same as before we have an undirected graph $G = (V,E)$ and k-pairs of sources and targets $(s_1,t_1),(s_2,t_2),\ldots,(s_k,t_k) \in V^2$. But we will introduce a new parameters $d_1,d_2,\ldots,d_k \in \mathbb{R}^+$ called **demands**.

Firstly we will take a look at linear program for solving this problem for $k = 1$.

$$\begin{array}{rc} & \max f \\ & \sum_{p\in\mathcal{P}_{st}} x_p \geq f \cdot d_1 \\ \forall e \in E & \sum_{e\in\in\mathcal{P}_{st}} x_p \leq c(e) \\ & x \geq 0 \end{array}$$

Where $\mathcal{P}_{st}$ are all paths between s and t. We may see that the optimum of the max flow is the same as this optimum just divided by d_1. We will denote $\mathcal{P}_i = \mathcal{P}_{s_i,t_i}$.

3.1 Concurrent multicommodity flow

Thus we are getting this LP for all k commodities and k demands.

$$\begin{array}{rc} & \max f \\ \forall i \in [k] & \sum_{p\in\mathcal{P}_i} x_p \geq f \cdot d_i \\ \forall e \in E & \sum_{i=0}^{k} \sum_{e\in\in\mathcal{P}_i} x_p \leq c(e) \\ & x \geq 0 \end{array}$$

We will take a look at the matrix of this LP and after that find a dual program. But firstly we modify $\sum_{p\in\mathcal{P}_i} x_p \geq f \cdot d_i$ to $f \cdot d_i - \sum_{p\in\mathcal{P}_i} x_p \leq 0$. Then the matrix is as follows:

	f		$\mathcal{P}_1$			$\mathcal{P}_2$		...
1	d_1	-1	-1	...	0	...	0	...
2	d_2	0	0	...	-1	-1	...	0
$\vdots$								
k	d_k	0	0	...	0	0	0	...
e_1	0	1	0		1	0	0	...
$\vdots$								
$e_{\lvert E\rvert}$	0	0	1		1	0	1	...

Where for the first k lines are ≤ 0 and for edges it is $\leq c(e)$. We visualized the matrix and thus we can make the dual. We will have variables x_e for edges and y_i for $i \in [k]$. Thus the dual is:

$$\begin{array}{rc} & \min \sum_{e\in E} x_e c(e) \\ & \sum_{i=0}^{k} y_i d_i \geq 1 \\ \forall i \in [k] \forall p \in \mathcal{P}_i & \sum_{e\in p} x_e - y_i \geq 0 \\ & x,y \geq 0 \end{array}$$

Definition 1. *For $S \subseteq V$ we define $\delta(S) = \{\{u,v\} \in E : |\{u,v\} \cap S| = 1\}$ and then $I(S) = \{i \in [k] : |\{s_i,t_i\} \cap S| = 1\}$. Then the **sparsity** of S is*

$$\rho(S) = \frac{\sum_{e\in\delta(S)} c(e)}{\sum_{i\in I(S)} d_i}$$

Example. We will have a simple example where all capacities are 1 and all demands are 1. So we have the graph **??**.

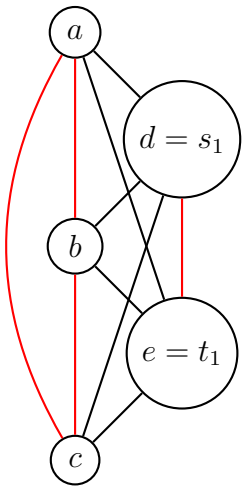


Figure 3.1: Sparse cut example.

The demands are the red edges. We may see that if we choose $S = \{c, e\}$ then $\sum_{e\in\delta(S)} c(e) = 3$ and $\sum_{i\in I(S)} d_i = 3$ therefore $\rho(S) = 1$.

We may see that each pair s_i t_i consumes at least 2 units of a flow of the network for a single unit of the flow. Then we set f as a max flow and see what we get. For example for paths $\mathcal{P}_1 = \{(d,a,e) = p_1, (d,b,e) = p_2, (d,c,e) = p_3\}$. $x_{p_1} + x_{p_2} + x_{p_3} \geq f \cdot d - 1 = f$. Thus the total volume consumed by a flow with objective value f is $\geq k2f = 8f$. Total volume of G is 6. Therefore $f \leq \frac{6}{8} = \frac{3}{4}$.

Maybe we can ask if there exist such a flow with this volume. We can obtain it by pushing $\frac{1}{4}$ from d to c on each path and $\frac{3}{8}$ between all other pairs on all paths. All edges are not over their capacities and we get $\frac{3}{4}$ for all demands.

Definition 2. *Now* $F \subseteq E : I(F) = \{i \in [k] : s_i, t_i$ *are in different components in* $(V, E \setminus F)\}$. *And* ***sparsity*** *of* F *be*

$$\rho(F) = \frac{\sum_{e\in F} c(e)}{\sum_{i\in I(F)} d_i}$$

Lemma 2. $\min_{S\subseteq V} \rho(S) = \min_{F\subseteq E} \rho(E)$.

Proof. The $\geq$ inequality can be easily seen if we set F to be $\delta(S)$. Now we need to show the other inequality. For given $F \subseteq E$, let $s_1, \dots s_l$ be the components of connectivity of $(V, E \setminus F)$. For that we will proof that $\min_{i\in[l]} \rho(s_i) \leq \rho(F)$. This will bw shown by a contradiction. Assume $\forall i$:

$$\frac{\sum_{e\in\delta(s_i)} c(e)}{\sum_{j\in I(s_i)} d_j} > \frac{\sum_{e\in F} c(e)}{\sum_{j\in I(F)} d_j}$$

$$\sum_{e\in\delta(s_i)} c(e) > \rho(F) \cdot \sum_{j\in I(s_i)} d_j$$

Now we sum all i inequalities.

$$\sum_{i=1}^{l} \sum_{e \in \delta(s_i)} c(e) > \rho(F) \cdot \sum_{i=1}^{l} \sum_{j \in I(s_i)} d_j$$

We can see that $\sum_{i=1}^{l} \sum_{e \in \delta(s_i)} c(e) = 2 \sum_{e \in F} c(e)$ because all edges are counted twice and similarly $\sum_{i=1}^{l} \sum_{j \in I(s_i)} d_j = 2 \sum_{j \in I(F)} d_j$. So we get:

$$\sum_{e \in F} c(e) > \rho(F) \cdot \sum_{j \in I(F)} d_j$$

Which is a contradiction. So for each F we can find s_i that satisfies the inequality. □

Now we can use this for integer program and then use relaxation. The program will look like this:

$$\begin{array}{rc} & \min \frac{\sum_{e \in E} c(e) x_e}{\sum_{i=1}^{k} d_i y_i} \\ \forall i \in [k] \forall p \in \mathcal{P}_i & \sum_{e \in p} x_e \geq y_i \\ & \sum_{i=1}^{k} d_i y_i \geq 1 \\ \forall e \in E & x_e \in \{0,1\} \\ \forall i \in [k] & y_i \in \{0,1\} \end{array}$$

At least one edge has to be removed from each path. Plus we assume that $d_i \geq 1$. Now we could just put $x_e \geq 0$ and $y_i \geq 0$. But the thing is that we don't have a linear function in the objective function. What if we have a vector $(x,y) \to (\alpha x, \alpha y)$ for $\alpha > 0$. You can see that the feasible solution don't change and also the objective is the same. So we could put $\alpha = \frac{1}{\sum_{i=1}^{k} d_i y_i}$ and we know that the $\sum_{i=1}^{k} d_i y_i = 1$. Thus the linear program will be:

$$\begin{array}{rc} & \min \sum_{e \in E} c(e) x_e \\ \forall i \in [k] \forall p \in \mathcal{P}_i & \sum_{e \in p} x_e \geq y_i \\ & \sum_{i=1}^{k} d_i y_i = 1 \\ \forall e \in E & x_e \geq 0 \\ \forall i \in [k] & y_i \geq 0 \end{array}$$

Before we continue we remind ourselves the Manhattan distance $||z||_1 = \sum_{j=1}^{k} |z_j|$. This is indeed a metric, which means that it is non-negative, symmetric and triangular inequality holds.

Lemma 3. *Let f be a mapping $f : V \to \mathbb{R}^d$ for some $d > 0$ and let*

$$\begin{array}{lrcl} \forall \{u,v\} \in E : & \hat{x}(\{u,v\}) & = & ||f(u) - f(v)||_1 \\ \forall i \in [k] : & \hat{y}(i) & = & ||f(s_i) - f(t_i)||_1 \\ & \beta & = & \sum_{i=1}^{k} d(i) \hat{y}(i). \end{array}$$

*Then $\left(\frac{\hat{x}}{\beta}, \frac{\hat{y}}{\beta}\right)$ is feasible solution. Also this is called **solution induced by** f. And we will denote $\left(\frac{\hat{x}}{\beta}, \frac{\hat{y}}{\beta}\right) = (x', y')$.*

Proof. We need to show that all conditions of LP are satisfied. Easily the non-negativity still holds. Also

$$\sum_{i=1}^{k} y'(i) d(i) = \sum_{i=1}^{k} \frac{\hat{y}(i)}{\beta} d(i) = 1$$

where the last equality holds by the definition of β. Lastly we need to check that $\sum_{e\in E} x(e) \geq y(i)$ of our LP still holds. This can be easily proven by the fact that $||\cdot||_1$ is metric so in particular triangular inequality is satisfied and by induction on the length of the path we would prove it. Also keep in mind that scaling by β doesn't change anything for the whole inequality since it is on both sides. □

Lemma 4 (A)**.** *Let (x',y') be a solution induced by $f : V \to \mathbb{R}^d$. Then one can find in polynomial time cut $S \subseteq V$ of sparsity $\rho(S) \leq \sum_{e\in E} x'(e)c(e)$.*

Lemma 5 (B)**.** *Given any feasible solution (x,y) of LP, one can construct a mapping $f : V \to \mathbb{R}^d$ (by random algorithm with high probability) which induces a solution $(\bar{x}, \bar{y})$ s.t.*

$$\sum_{e\in E} c(e)\bar{x}(e) = O(\log k) \sum_{e\in E} c(e)x(e)$$

Theorem 2. *There exist a randomized polynomial-time algorithm for the sparsest cut problem that is $O(\log k)$-approximation.*

Proof. By Lemma B (5) we generate $(\bar{x}, \bar{y})$ and then by Lemma A (4) we construct the cut. □

Proof of Lemma A 4. Given $f : V \to \mathbb{R}^d$ let

$$\forall u,v \in V \quad \mu(u,v) \quad = \quad ||f(u) - f(v)||_1$$

For $S \subseteq V$ we define $\forall u,v \in V$

$$\mu_S(u,v) = \begin{cases} 1 & \text{iff } |\{u,v\} \cap S| = 1 \\ 0 & \text{otherwise} \end{cases}$$

This will be called **cut mapping** and we can easily see that it is non-negative, symmetric and triangular inequality is satisfied thus it is metric. Before we continue we will use another lemma.

Lemma 6 (lemma)**.** *$\forall S \subseteq V$ $\exists \lambda_S \geq 0$ s.t. $\forall u,v \in V : \mu(u,v) = \sum_{S\subseteq V} \lambda_S \mu_S(u,v)$. Moreover $|\{S|\lambda_S > 0\}| \leq n \cdot d$.*

Proof of lemma 6. Consider the contribution of the first coordinate to $\mu(u,v)$: order the vertices according to f_1 where $f = (f_1, f_2, \ldots, f_d)$, s.t. $f_1(v_1) \leq f_1(v_2) \leq \cdots \leq f_1 v_n$. Now let $S(l) = \{v_1, \ldots, v_l\}$ for $l \in [n]$. Consider any two vertices v_i,v_j s.t. $i > j$.

$$f_1(v_i) - f_1(v_j) = \sum_{l=j}^{i-1} (f_1(v_{l+1}) - f_1(v_l)) = \sum_{l=1}^{n-1} (f_1(v_{l+1}) - f_1(v_l))\mu_{S(l)}(v_i, v_j)$$

Where $(f_1(v_{l+1}) - f_1(v_l)) = \lambda_{S(l)}$. This can be used to prove this for all dimensions $f_2, \ldots, f_d$ thus it is true for f. □

Observation. *For any non-negative numbers $a_1, \ldots, a_n$ and positive numbers $b_1, \ldots, b_n$ holds:*

$$\frac{\sum_{i=1}^n a_i}{\sum_{i=1}^n b_i} \geq \min_{i\in[n]} \frac{a_i}{b_i}$$

Proof of observation. By a contradiciton assume $\forall j$:

$$\frac{\sum_{i=1}^{n} a_i}{\sum_{i=1}^{n} b_i} < \frac{a_j}{b_j}$$

$$b_j \frac{\sum_{i=1}^{n} a_i}{\sum_{i=1}^{n} b_i} < a_j$$

$$\sum_j b_j \frac{\sum_{i=1}^{n} a_i}{\sum_{i=1}^{n} b_i} < \sum_j a_j$$

Where the last line is summing all the inequalities together. We get a contradiction. *Note that geometricaly that can be represent as vectors and values of the* tan *function and it would state that there is a* tan *smaller of one of the vectors than the sum of them.* □

Now we continue to proof the Lemma A.

$$\sum_{e \in E} c(e)x'(e) = \frac{\sum_{e \in E} c(e)x'(e))}{\sum_{i=1}^{k} y'(i)d(i)}$$

Which is just a division by 1 from the conditions in LP. Then by lemma:

$$= \frac{\sum_{e \in E} c(e) \sum_{S \subseteq V} \lambda_S \mu_S(e)}{\sum_{i=1}^{k} d(i) \sum_{S \subseteq V} \lambda_S \mu_S(s_i,t_i)} =$$

$$= \frac{\sum_{S \subseteq V} \lambda_S \sum_{e \in E} c(e)\mu_S(e)}{\sum_{S \subseteq V} \lambda_S \sum_{i=1}^{k} d(i)\mu_S(s_i,t_i)} \geq \min_{S \subseteq V} \rho(S)$$

The last part is due to the previous observation and the fact that $\sum_{e \in E} c(e)\mu_S(e) = a_S$ and $\sum_{i=1}^{k} d(i)\mu_S(s_i,t_i) = b_S$ taken means $\frac{a_S}{b_S} = \rho(S)$. □

Now we will be proving the Lemma A 5. For that we denote $T = \{s_i | i \in [k]\} \cap \{t_i | i \in [k]\}$ and without loss of generality assume that $|T| = 2^\tau$ (we can add arbitrary sources and targets that are essentially the same). Let us denote $d_x(u,v)$ the length of the x-shortest $u - v$ path. Where x is the result of our LP. For $A \subseteq V : d_x(A,u) = \min_{v \in A} d_x(v,u)$.

Also we put $L = q \log(k)$ where q is some constant to be decided later on and k is for number of commodities. Also $d = L \cdot \tau = O(\log^2(k))$. For $t = 1, \ldots, \tau$ and $l = 1, \ldots, L$: let A_{tl} be a set that is constructed by $2^{\tau - t}$-times selecting uniformely at random $v \in V$.

Definition 3. $\forall v \in V : f_{tl}(v) = d_x(v, A_{tl})$.

Note that both A_{tl} and f_{tl} are not dependent on l. One can say that l is for repeating the selection.

Lemma 7. $\forall \{u,v\} \in E : ||f(u) - f(v)||_1 \leq d \cdot x(u,v)$.

Proof. We proceed by the definition and some algebra.

$$||f(u) - f(v)||_1 = \sum_{t=1}^{\tau} \sum_{l=1}^{L} |f_{tl}(u) - f_{tl}(v)| = \sum_{t=1}^{\tau} \sum_{l=1}^{L} |d_x(u, A_{tl}) - d_x(v, A_{tl})|$$

Now lets take a look at these inequalities which follows from the triangle inequalities.

$$\begin{array}{rcl} d_x(u, A_{tl}) & \leq & x(u,v) + d_x(v, A_{tl}) \\ d_x(v, A_{tl}) & \leq & x(u,v) + d_x(u, A_{tl}) \\ & \Downarrow & \\ d_x(u, A_{tl}) - d_x(v, A_{tl}) & \leq & x(u,v) \\ d_x(v, A_{tl}) - d_x(u, A_{tl}) & \leq & x(v,u) \end{array}$$

Which leads to $|d_x(u,A_{tl}) - d_x(v, A_{tl})| \leq x(u,v)$ and thus getting the last inequality:

$$\sum_{t=1}^{\tau} \sum_{l=1}^{L} |d_x(u, A_{tl}) - d_x(v, A_{tl})| \leq \tau \cdot L \cdot x(u,v) = d \cdot x(u,v)$$

□

Lemma 8. *With probability $\geq 1/2$: $\forall i \in [k]$ holds that*

$$||f(s_i) - f(t_i)||_1 \geq \frac{L}{88} y_i.$$

Before proving this lemma we will take a look, how useful it is. $\beta = \sum_{i=1}^{k} d(i) \cdot ||f(s_i) - f(t_i)||_1 = \Omega(\log k) \cdot \sum_{i=1}^{k} d(i) y_i = \Omega(\log k)$ where y_i is from our LP and thus $\sum_{i=1}^{k} d(i) y_i$ is equal to 1. The second equality is from the lemma before. And now from the lemma even before that we get $\leq \sum_{e \in E} c(e) \cdot d \cdot x(e) = d \sum_{e \in E} c(e) x(e)$ which is the objective function result of our LP. Thus $= O(\log^2(k)) \sum_{e \in E} x(e) c(e)$. But that is scaled by β thus the objective value of the solution induced by f is $\leq O(\frac{\log^2(k)}{\log(k)} \sum_{e \in E} c(e) x(e) = O(\log k) \sum_{e \in E} x(e) c(e)$ and so we have $O(\log k)$-approximation. So this proves the Lemma B 5.

Proof. To prove the lemma we will prove a simple version that for fixed $i \in [k]$ with probability $\geq 1 - 1/2k$ it holds that

$$||f(s_i) - f(t_i)||_1 \geq \frac{L}{88} y_i.$$

We can easily see that for doing this for all $i \in [k]$ the lemma will follow. To prove this we will define few more things.

$$\forall v \in \{s_i, t_i\} : B_x(v, r) = \{w \in T | d_x(v,w) \leq r\}$$

$$\forall v \in \{s_i, t_i\} : B_x^{\circ}(v, r) = \{w \in T | d_x(v,w) < r\}$$

Now we will look at this sequence of radii. $r_0 = 0$,

$$r_t = \min\{r > 0 : |B_x(s_i,r)| \geq 2^t \wedge |B_x(t_i,r)| \geq 2^t\}$$

$$\hat{t} = \min\left\{t | r_t \geq \frac{y(i)}{4}\right\}$$

and also redefine $r_{\hat{t}} = \frac{y(i)}{4}$. This is define with respect to LP. And also it means that $B_x(s_i, r_{\hat{t}}) \cap B_x(t_i, r_{\hat{t}}) = \emptyset$.

Now we observe that for $A_{tl} \subseteq V : A_{tl} \cap B_x^{\circ}(s_i, r_t) = \emptyset \Leftrightarrow d_x(s_i, A_{tl}) \geq r_t$. And also $A_{tl} \cap B_x(t_i, r_{t-1}) \neq \emptyset \Leftrightarrow d_x(t_i, A_{tl}) \leq r_{t-1}$. Let E_{tl} be the event such that $A \cap B = \emptyset$ and $A \cap G \neq \emptyset$ where $B = B_x^{\circ}(s_i, r_t)$ and $G = B_x(t_i, r_{t-1})$.

We may observe that if E_{tl} happens then $|f_{tl}(s_i) - f_{tl}(t_i)| = |d_x(s_i, A_{tl}) - d_x(t_i, A_{tl})| \geq r_t - r_{t-1}$. We will look at the probability of happening this.

$$\begin{aligned} \Pr[E_{tl}] &= \Pr[A_{tl} \cap G \neq \emptyset | A_{tl} \cap B = \emptyset] \Pr[A_{tl} \cap B = \emptyset] \\ &\geq \Pr[A_{tl} \cap G \neq \emptyset] \Pr[A_{tl} \cap B = \emptyset] \end{aligned}$$

Let us assume wlog s_i defines r_t.

$$\Pr[A_{tl} \cap B = \emptyset] = \left(1 - \frac{|B|}{|V|}\right)^{2^{\tau-t}} \geq \left(1 - \frac{2^t}{2^\tau}\right)^{\frac{2^\tau}{2^t}} \geq \frac{1}{e} \geq \frac{1}{4}$$

$$\Pr[A_{tl} \cap G \neq \emptyset] = (1 - \Pr[A_{tl} \cap G = \emptyset]) = 1 - \left(1 - \frac{|G|}{|V|}\right)^{2^{\tau-t}} \geq$$

$$\geq 1 - \left(1 - \frac{2^{t-1}}{2^\tau}\right)^{\frac{2^\tau}{2^{t-1}} \frac{1}{2}} \geq 1 - \left(\frac{1}{e}\right)^{\frac{1}{2}} \geq \frac{4}{11}$$

Thus the $\Pr[E_{tl}] \geq \frac{1}{11}$. Now we fix $t = \{1, \ldots, \tau\}$ and define:

$$X_{tl} = \begin{cases} 1 & \text{iff } E_{tl} \text{ occurs} \\ 0 & \text{otherwise} \end{cases}$$

For $l = 1, \ldots, L$ let $\mu = \mathsf{E}\left[\sum_{l=1}^{L} X_{tl}\right]$. We may observe that $\mu \geq \frac{L}{11}$ by linearity of E. We now may use the Chernoff bound.

$$\Pr\left[\sum_{l=1}^{L} X_{tl} \leq \frac{\mu}{2}\right] \leq e^{\frac{-\mu}{8}} \leq e^{\frac{-q \log k}{88}} \leq e^{-\log 2k - \log\log 2k} = \frac{1}{2k \log 2k}$$

Where there is hidden analysis to proper choice of q. If $\sum_{l=1}^{L} X_{tl} \geq \frac{\mu}{2}$ then

$$\sum_{l=1}^{L} |f_{tl}(s_i) - f_{tl}(t_i)| \geq \sum_{l=1}^{L} X_{tl}(r_t - r_{t-1}) \geq \frac{L}{22}(r_t - r_{t-1})$$

Therefore with probability $\geq 1 - \frac{\tau}{2k \log 2k} \geq 1 - \frac{1}{2k}$ $\forall t \in [\hat{t}]$ the previous statement holds. Thus

$$||f(s_i) - f(t_i)||_1 \geq \frac{L}{88} y_i = \frac{L}{88} 4 \sum_{t=1}^{\tau} (y_t - y_{t-1})$$

□

3.2 Metric spaces

Some of basic definitions are for metric spaces which reader may already know, but we will remind it once again.

Definition 4. *Metric space* (M,d) *when* $d : M \times M \to \mathbb{R}^+$ *and*

(i) $\forall x,y \in M : d(x,y) \geq 0$ *and* $d(x,y) = 0 \Leftrightarrow x = y$,

(ii) $\forall x,y \in M : d(x,y) = d(y,x)$,

(iii) $\forall x,y,z \in M : d(x,z) \leq d(x,y) + d(y,z)$.

We may already know some examples. One is for $G = (V,E)$ and for $x : E \to \mathbb{R}^+$ the metric is $d(z,y) = \min_{z-y \text{ paths}} \sum_{e \in P} x(e)$. This means that (V,d) is a metric system.

Definition 5. *Let (X,d) and $(Y,\bar{y})$ be metric spaces. An injective function $f : X \to Y$ is D-embedding for some $D \geq 1$, if $\exists r > 0$ such that $\forall x,y \in X$ the following holds*

$$r \cdot d(x,y) \leq \bar{d}(f(x), f(y)) \leq D \cdot r \cdot d(x,y).$$

Definition 6. *The* inf *of D values satisfying the above property is called* ***distortion*** *of f.*

Theorem 3 (Bourgain, 1985)**.** *Every n-point metric space (V,d) can be embedded in $(\mathbb{R}^p, l_1)$ with distortion $O(\log n)$ with $p = O(\log^2 n)$. Where $l_1 = ||x||_1 = \sum_i x_i$.*

We remind ourselves what we did. We constructed $f : V \to \mathbb{R}^+$ and $p = O(\log^2 k)$ such that

(i) $\forall u,v \in V : ||f(u) - f(v)||_1 \leq p \cdot d_x(u,v)$,

(ii) $\forall i \in [k] : ||f(s_i) - f(t_i)||_1 \geq \Omega(\log k) d_x(s_i, t_i)$.

So if we take $T = V$, think about all pair of vertices as commodities. Everything in the proof still works. So we technically proved this theorem before.

Now the question one can ask is: *Is the analysis tight?* For the answer we may recall that in 3-regular graphs, there are $\Omega(n^2)$ pairs of vertices at distance $\Omega(\log k)$. That was for the multi-commodity system. Lets consider 3-regular β-expanders (i.e. $\delta(S) \geq \beta|S|, \forall S \subseteq V, |S| \leq |V|/2$).

With that consider the instance: Commodity for each pair of vertices and set all $d = 1$. This will lead to

$$\min_{S \subseteq V} \frac{E(S, V \setminus S)}{I(S)} \geq \min_{S \subseteq V} \frac{\beta|S|}{|S||V \setminus S|} \geq \frac{\beta}{|V|} = \Omega(1/n)$$

then the max concurrent flow is at most the available capacity $O(n)$ divided by what unit of flow consumes. Thus get

$$\frac{O(n)}{\Omega(n^2 \log n)} = O\left(\frac{1}{n \log n}\right)$$

which means that it is tight. Alternatively integrality gap of our LP is $\Omega(\log n)$. Also it implies that the asymptotic optimality of the theorem is tight. Otherwise if there was better version we could use that for better approximation of our LP which is a contradiction. Also there exists $O(\sqrt{\log n})$-approximation for sparsest cut using positive semidefinite programming.

Corollary. Max flow $\leq$ min cut $\leq O(\log k)$ max flow. For sparsest cut problem.

3.3 Applications

Definition 7. *Cut $(S, V \setminus S)$ is b-balanced (for some $b \leq 1/2$) if*

$$bn \leq |S| \leq (1-b)n$$

where $G = (V,E)$ and $|V| = n$.

1/2-balanced is called **bisection**. Also there is a problem for finding a b-balanced cut minimizing the number of edge between $E(S, V \setminus S)$ (*cost of cut*). This problem is generally NP-hard.

Theorem 4. *If there is a b-balanced cut T in $G = (V,E)$, then for any $b' < \min\{1/3, b\}$ one can find in polynomial time b'-balanced cut of cost $O\left(\frac{E(T,V\setminus T)\log n}{b-b'}\right)$.*

Proof. First we define the algorithm.

Algorithm 2 Find b'-balanced cut.

Require: Graph G.
Ensure: b'-balanced cut.
1: $i := 0$, $G_i = G$, $S = \emptyset$
2: **while** $|V(G_i)| > (1-b')|V|$ **do**
3: find an approximation of the sparsest cut in G_i and denote it as $S_i \subseteq V(G_i)$
4: let $G_{i+1} = G_i[V(G_i) \setminus S_i]$, $S = S \cup S_i$, $i = i+1$
5: **end whilereturn** S

where in the sparsest cut problem is on the network where all vertices are terminals and demands are 1.

Correctness of the algorithm: Before the last iteration it is true that $|S| < b'n$. In the last iteration at most $|V(G_i)|/2$ are added to S. Therefore at the end

$$\leq |S| + \frac{n-|S|}{2} = \frac{n+|S|}{2} < \frac{(1+b')n}{2} \leq (1-b')n$$

Where the last inequality is due the value $b' \leq 1/3$ and the fact that $1 + b' \leq 2 - 2b'$. Also because it ended $|S| \geq b'n$ so it is indeed a b'-balanced cut.

Approximation of the cost: Consider an optimal b-balanced cut $(T, V \setminus T)$. In each iteration $|T \setminus S| \geq (b-b')n$. What is the sparsity of the cut $T \setminus S$? Lets denote $\text{opt} = E(T, V \setminus V)$. The sparsity is

$$\leq \frac{\text{opt}}{b-b')n(1-b)n} \leq \frac{2\text{opt}}{(b-b')n^2}$$

so the sparsity of the $O(\log n)$-approximation S_i found by the algorithm is

$$\frac{E(S_i, V_i \setminus S_i)}{|S_i||V_i \setminus S_i|} \leq O(\log n)\frac{\text{opt}}{(b-b')n^2}$$

which means that $E(S_i, V_i \setminus S_i) \leq O(\log n)\frac{\text{opt}}{(b-b')n^2}|S_i|$. Now we sum it up.

$$E(S, V \setminus S) \leq \sum_i E(s_i, V_i \setminus S_i) \leq O(\log n)\frac{\text{opt}}{(b-b')n^2}\sum_i |S_i| = O(\log n)\frac{\text{opt}}{b-b'}$$

□

3.4 Minimum cut linear arrangement

Given $G = (V,E)$, find ordering $v_1, \ldots, v_n$ of the vertices such that

$$\max_{i \subset [n]} E(\{v_1, \ldots, v_i\}, \{v_i, \ldots, v_n\}) \text{ is minimized.}$$

Observation. $OPT \geq \min$ *bijection of* $G =: \mathcal{B}$.

Proof. For any ordering $E(\{v_1, \ldots, v_{n/2}\}\{v_{n/2+1}, \ldots, v_n\}) \geq B$. □

Algorithm 3 Find minimum cut linear arrangement

Require: Graph G
Ensure: Minimum cut linear arrangement
1: Find a 1/3-balanced cut of G and denote it as (L,R) by the previous algorithm.
2: Solve the problem recursively for L,R.

Observation. *The depth of recursion is* $O(\log n)$.

Observation. $E(L,R) \leq O(\log n) \cdot B$.

And now we would like to get similiar bound for all the levels of recursion. For that consider G_i. Lets denote B_i the bijection of G_i and OPT_i the optimum solution for G_i. Then $B_i \leq OPT_i \leq OPT$, therefore in our solution

$$\forall i \in [n], E(\{v_1, \ldots, v_i\}\{v_{i+1}, \ldots, v_n\}) \leq O(\log n) \cdot O(\log n) \cdot OPT$$

because the first O is for number of recursion calls and the second O is approximation of the size of each balanced cut. Altogether it is equal to $O(\log^2 n)OPT$.

Theorem 5. *The approximation ratio of the algorithm is* $O(\log^2 n)$.

Definition 8. *Crossing number of the graph is the number of intersections of edges (the minimum). For planar graphs it is 0 and for not planar it is* ≥ 1.

This can be also solved by the algorithm above.

4. Max cut

We have been talking about minimal cuts the whole time. Now we will consider somewhat opposite problem. That is for given graph $G = (V,E)$ we want to find $S \subseteq V$ such that $E(S, V \setminus S)$ is maximized.

For this problem we may introduce a **randomized algorithm** which is simple. For every vertex choose if it is in S or in $V \setminus S$ with probability 1/2. Then $\mathsf{E}[|E(S, V \setminus S)|] = \frac{|E|}{2} \geq \frac{OPT}{2}$ because the probability of edge being in the cut is exactly one half, since there are four options where u and v may land, but in two scenarios they are in the same part and in the rest they are on the opposite sites.

Now we would like to talk about 0,878...-approximation. Firstly we will label our vertices. WLOG: $V = \{1, 2, \ldots, n\}$. Set $\forall i \in V : y_i^2 = 1$. Now think about an edge ij. How can we express with this representation of the graph that ij is in the cut? Think about

$$y_i \cdot y_j = \begin{cases} 1 & \text{on the same side} \\ -1 & \text{on different sites} \end{cases}$$

and from this we would make

$$\frac{1 - y_i}{2} \cdot y_j = \begin{cases} 0 & \text{on the same side} \\ 1 & \text{on different sites} \end{cases}.$$

So with this we can introduce a maximalization problem:

$$\max \frac{1}{2} \sum_{\{i,j\} \in E} (1 - y_i y_j)$$

Altogether we can define a **quadratic formulation** for max-cut problem. Every part was already mentioned, but just to gather it on one place.

$$\begin{gathered} \forall i \in V : y_i^2 = -1 \\ \max \tfrac{1}{2} \textstyle\sum_{\{i,j\} \in E} (1 - y_i y_j) \\ y_i \in \mathbb{R} \end{gathered}$$

We may see that this program is not very good for solving. This means we will relax it to a **vector program** and we will denote it as VP for future usage.

$$\begin{gathered} \max \tfrac{1}{2} \textstyle\sum_{\{i,j\} \in E} (1 - y_i^T y_j) \\ y_i^T y_i = 1 \\ \forall i : y_i \in \mathbb{R} \end{gathered}$$

The intuition behind it is that we start with 1 dimensional ball (which are line segments [−1,1]) and then we will continue to higher dimensions. In the VP we use vectors instead of usual numbers. We will continue with changing the program. Now it will be to semi-positive programming or SPD for short. This formulation is as follows.

$$\begin{gathered} \max \tfrac{1}{2} \textstyle\sum_{\{i,j\} \in E} (1 - Y_{ij}) \\ \forall i : Y_{ij} = 1 \\ Y \text{ is positive semi-definite} \end{gathered}$$

When it is written in this form it can be solved in polynomial time. Just for a reminder we introduce a definition of PSD.

Definition 9. *We say that a symmetric matrix $A \in \mathbb{R}^{n\times n}$ is positive semi-definite if* $\forall x : x^T Ax \geq 0$.

Observation. *A is positive semi-definite* $\Leftrightarrow \exists$ *matrix $U \in \mathbb{R}^{n\times n}$ s.t. $A = U^T U$.*

Observation. *VP = PSD*

Proof. This is from the above observation because we can put the semi-positive matrix Y to a multiplication of two matrices which will look like this:

$$\begin{pmatrix} \cdots & y_1 & \cdots \\ & \vdots & \\ \cdots & y_n & \cdots \end{pmatrix} \begin{pmatrix} \vdots & & \vdots \\ y_1 & \cdots & y_n \\ \vdots & & \vdots \end{pmatrix}$$

□

Algorithm 4 Algorithm for max cut

Require: Graph G
Ensure: S max cut of G.
1: Solve SDP.
2: Interpret it as a solution of the VP $\rightarrow v_1, \ldots, v_n \in \mathbb{R}^n$.
3: Sample uniformly at random $r \in \{x \in \mathbb{R}^n : x^2 = 1\}$
4: **return** $S = \{i \in V : v_i^T r \geq 0\}$.

We know that both v_i and r are unit vectors. So $\cos\alpha = v_i^T \cdot r$. Where the cos function can be seen on a picture 4.1 to visualize how it looks.

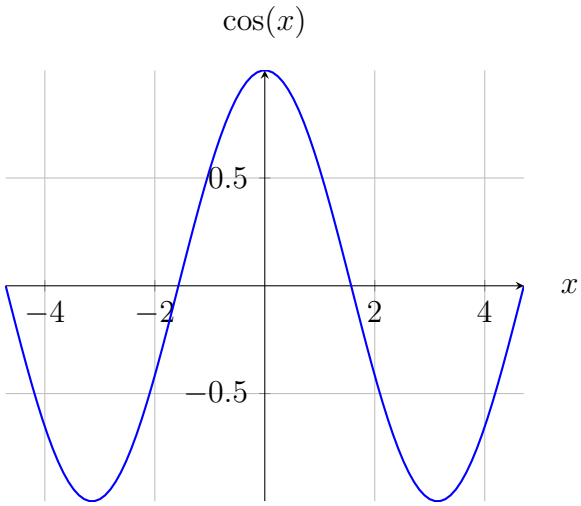


Figure 4.1: Cosine function.

Now we will take a step back and try to achieve the mysterious number for the approximation ratio. Let θ_{ij} be tha angle between v_i and v_j. Then for $\{i,j\} \in E$, its contribution to the objective (in VP) is $\frac{1-\cos\theta_{ij}}{2}$.

Lemma 9 (no proof)**.** *For each $x \in \langle 0, \pi\rangle$ and $\alpha = 0{,}87856$ it holds that*

$$\frac{x}{\pi} \geq \alpha \frac{1-\cos x}{2}$$

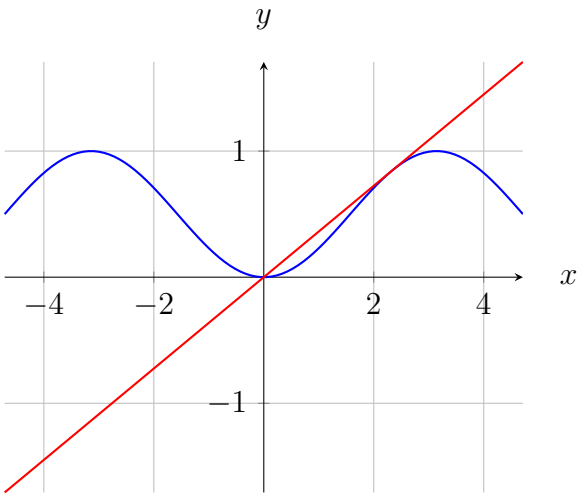


Figure 4.2: The red function is for $x/(\pi \cdot \alpha)$ and blue for $(1 - \cos(x))/2$.

The meaning of it is shown on the picture 4.2.

Lemma 10. *For* $\{i,j\} \in E$ $\Pr[i \text{ and } j \text{ are separated}] = \frac{\theta_{ij}}{\pi}$.

Proof. Consider the projection of r to the plane defined by v_i, v_j. Let W be the objective value of our solution:

$$\begin{aligned}
\mathsf{E}[W] &= \sum_{\{i,j\}\in E} \frac{\theta_{ij}}{\pi} && \text{(by linearity of expectation)} \\
&\geq \alpha \sum_{\{i,j\}\in E} \frac{1-\cos(\theta_{ij})}{2} && \text{(by the first lemma)} \\
&= \alpha \sum_{\{i,j\}\in E} \frac{1-v_i^T v_j}{2} && \textbf{(this is our objective functions of VP)} \\
&\geq \alpha OPT && \text{(because VP is a relaxation)}
\end{aligned}$$

□

5. Edge disjoint path problem

Some readers may already know what is edge disjoint path problem and also some basic algorithms. But we will have a brief introduction to this topic.

- **INPUT**: $G = (V,E)$ and $(s_i, t_i) \in V^2$ for all $i \in [k]$.
- **OUTPUT**: $I \subseteq [k]$ and an $s_i - t_i$ path P_i for each $i \in I$, s.t. the selected paths are edge disjoint.
- **OBJECTIVE**: $\max |I|$.

It is known that this particular problem is NP-hard. So we will again show some approximation to this problem. *Note: for any fix k it is solvable in polynomial time on undirected graph. But for directed graphs it is NP-hard for $k = 2$.* We will introduce an greedy algorithm that has an parameter.

Algorithm 5 Greedy algorithm with a catch for parameter $\sqrt{m}$

$I = \emptyset$
while $\exists i \notin I$ and $\exists s_i - t_i$ path in G, s.t. $|P_i| \leq \sqrt{m}$ **do**
    $I = I \cup \{i\}$, keep P_i, $G = G \setminus P_i$
end while

We will denote OPT as the optimal solution of the problem. It will be either a set of paths or set of indexes. Then we will denote $OPT_S = \{P \in OPT \mid |P| \leq \sqrt{m}\}$, where the length of a path is set as the number of edges. Then $OPT_L = OPT \setminus OPT_S$ and ALG as the set given by the algorithm.

Now take the set $OPT_S \setminus ALG$. That is path between s_i and t_i is in this set if there exists $s_j - t_j$ path obtained by the algorithm which shares an edge. This path has length at most $\sqrt{m}$ and there are $|ALG|$ paths. Thus altogether $|OPT_S \setminus ALG| \leq \sqrt{m}|ALG|$.

Next we may see that $|OPT_L| \leq \sqrt{m}$, because we have m edges and each one of them is at least $\sqrt{m}$ long. Now we may conclude altogether following result.

$$|OPT| \leq |OPT_L| + |OPT_S \setminus ALG| + |ALG| \leq O(\sqrt{m})|ALG|$$

Now one can see where the catch in the algorithm is. Consider that there are no such short paths. The algorithm will output no path at all. To fix this we need to change the algorithm such that it will always output at least one path. If there is none then OPT is 0 as well.

Algorithm 6 Greedy $(\sqrt{m})$

$I = \emptyset$
while $\exists i \notin I$ and $\exists s_i - t_i$ path in G, s.t. $|P_i| \leq \sqrt{m}$ **do**
    $I = I \cup \{i\}$, keep P_i, $G = G \setminus P_i$
end while
if $I = \emptyset$ **then**
    Connect any $s_i - t_i$ path if possible.
end if

Thus we have shown an algorithm that is a $\sqrt{m}$-approximation. Now we consider running the same algorithm but we change the parameter from $\sqrt{m}$ to $n^{2/3}$. Can we obtain $n^{2/3}$-approximation?

Theorem 6 (Khana, Chedari)**.** *Given an instance of the sum multi-commodity flow problem $G = (V,E), (s_i, t_i) \in V^2$ for all $i \in [k]$ such that $(\forall i) d(s_i, t_i) \leq l$, then the max multi-commodity flow is $O(\frac{n^2}{l^2})$.*

Before proving this we will show the consequences for our problem. Lets use the algorithm Greedy($n^{2/3}$). Assume there $\exists P_i \in ALG, |P_i| \leq n^{2/3}$. Otherwise we use the theorem on the network obtained by G and setting all capacities to one. Then all edges are at least $n^{2/3}$ length so we get the max multi-commodity flow is $O(\frac{n^2}{n^{4/3}}) = O(n^{2/3})$. Therefore it means if we choose just one path the approximation ratio will still be $O(n^{2/3})$.

Denote $OPT_{easy} = \{P \in OPT \mid \exists Q \in ALG : Q \cap P \neq \emptyset\}$. With this we know that $|OPT_{easy}| \leq n^{2/3}|ALG|$ by the same argument as it was already mentioned before.

We will look at $\forall (s_i, t_i) \in (OPT \setminus OPT_{easy}) \setminus ALG$. What can we say about such $d(s_i, t_i)$ at the end of the loop of the algorithm. Clearly because it was not chosen either there is some intersection with another path, but this is remove by OPT_{easy}, so the other option is only that $d(s_i, t_i) > n^{2/3}$. Hence $|(OPT \setminus OPT_{easy}) \setminus ALG| \leq O(n^{2/3})$ by the theorem and the same argument which was already mentioned. Altogether we have:

$$|OPT| \leq |OPT_{easy}| + |(OPT \setminus OPT_{easy}) \setminus ALG| + |ALG| = O(n^{2/3})|ALG|$$

Now we only need to proof the theorem since it is the base of our arguments for obtaining $O(n^{2/3})$-approximation algorithm.

Proof. We will split the vertices into two sets:

1. **low degree vertex** are when $\deg(v) \leq \frac{6n}{l}$
2. **high degree vertex** are when $\deg(v) > \frac{6n}{l}$

Also we will assume l is a multiple of 6. Otherwise it get lost in the O notation. To finish the proof we will use an observation.

Observation. *Any $s_i - t_i$ path (denote it as $s - t$) uses at least $l/6$ low degree vertices.*

Proof of observation. Consider running BFS on the graph starting from s. We denote $L_i = \{u \in V : d(s,u) = i\}$. Note that edges are only within one layer or only between adjacent layers. Let B_i be a block of three consecutive layers $\{L_{3i}, L_{3i+1}, L_{3i+2}\}$. Because the length to t is at least l then there is at least $l/3$ blocks. Assume that $< l/6$ layers consists of only low degree vertices. Otherwise the observation obviously holds. Now discard all blocks containing a layer of only low degree vertices. As there are $\geq l/3$ blocks at least $\geq l/6$ blocks remain. Then the smallest remaining block is of size $\leq \frac{n}{l/6} = 6n/l$ which can be seen by pigeonhole principle. For the vertices in the middle layer we know all neighbors are within the block. Therefore it is a low degree vertex. This is a contradiction because we still have a block having one layer with low degree vertices only. □

Now for the theorem we know a **unit** of flow between any pair $s_i - t_i$ consumes $\mathbf{\Omega(l)}$ cpacity of edges adjacent to low degree vertices. And the total capacity adjacent to low degree vertices is $\leq n \deg(v) \leq n(6n/l) = O(n^2/l)$. Which gives us $O(n^2/l^2)$. □

This is an example of greedy algorithm and the fact that using it with different parameter may result in better approximation, but the analysis is way harder. We also saw using flows to limit paths, but this has also its limits. We will show a counterexample a graph called **Brick wall**.

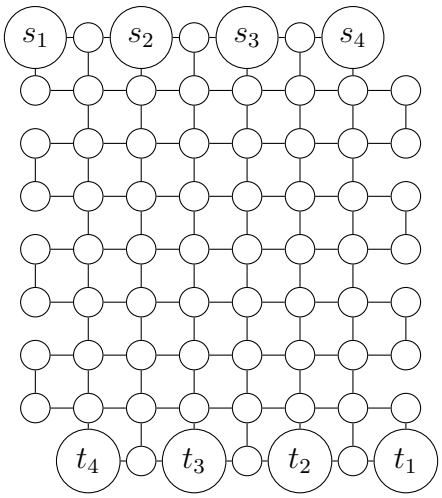
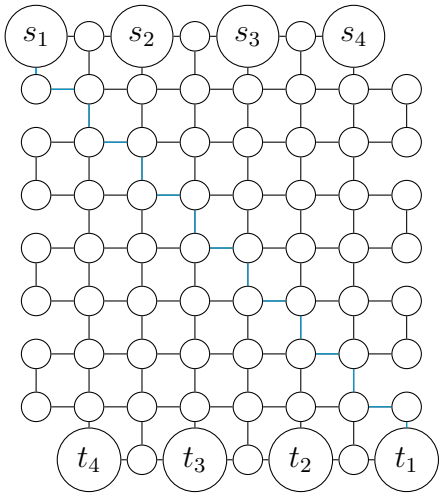


Figure 5.1: Example of a **brick wall** graph for $k = 4$.

The graph you may see at the picture 5.1 is planar. Also it can be generalized to k where there are k pairs of bricks underneath. The edge disjoint problem optimum is 1, which we can see on picture 5.2a. And the max multi flow optimum is at least $k/2 = O(\sqrt{n})$, which can be seen on picture 5.2b.



(a) Edge disjoint problem.

(b) Max multi flow optimum.

Figure 5.2: Example of optimalization problems.