CS 598CSC: Topics in Graph Algorithms **Lecture date: 11/04/2024**
Instructor: Chandra Chekuri **Scribe: CC**

1 Fast Algorithms for Sparsest Cut

Let $G = (V, E)$ be a graph. We are interested in a fast algorithm to check if G is an expander. More generally we also want fast algorithms for Expansion, Conductance and Uniform Sparsest Cut. There are basically three approximation algorithms that we have discussed or mentioned. One is based on multicommodity flow and yields an $O(\log n)$-approximation [LR99]. The other is via SDP and yields an $O(\sqrt{\log n})$-approximation [ARV09]. For Conductance there is an $O(\sqrt{\text{OPT}})$-approximation via spectral method. The spectral method is very fast but does not yield good approximation in the regime when the conductance is low. The other two methods are based on solving slow mathematical programming relaxations. We discussed MWU based methods for fast solutions to LPs and one of the motivations for these algorithms was to speed up the max-concurrent flow LP for Uniform Sparsest Cut. However, even with several ideas the fastest way we know how to solve the LP takes $O(mn)$ time for a constant factor approximation.

In a beatiful work, Khandekar, Rao and Vazirani showed that one can approximate expansion to within an $O(\log^2 n)$-factor via $O(\log^2 n)$ calls to an (approximate) single-commodity s-t flow routine [KRV09]. We now have near-linear time algorithms graphs for a constant factor approximation to s-t flow in undirected graphs [Pen16], and an almost-linear time exact algorithm for s-t flow in directed graphs [CKL$^+$22]. KRV obtained their algorithm via the so-called Cut-Matching Game which has since become a powerful tool in several applications. The Cut-Matching Game has been further improved to yield an $O(\log n)$-approximation [OSVV08]. Arora and Kale [AK07] developed a more systematic approach via matrix-multiplicative weight updates to obtain fast primal-dual algorithms that yield $O(\log n)$ and $O(\sqrt{\log n})$-approximation algorithms; these also use appropriate flow sub-routines.

Here we discuss the approach and proof of KRV. We also take a flow-cut gap perspective and point out some other applications. These notes are based mainly on Kent Quanrud's exposition and from [KRV09].

2 Cut Matching Game

We want to check if $G = (V, E)$ is an γ-expander. We would like an algorithm that either returns a cut to prove that G is *not* a γ-expander or somehow proves that G is a γ/α-expander for some approximation factor $\alpha \geq 1$. We will assume that $\gamma = 1$ since we can scale the edge capacities by $1/\gamma$. If we have a routine for checking approximate expansing with a given threshold, we can do binary search to find an approximate sparsest cut.

To motivate the cut-matching game we recall the notion of well-linkedness and its connection to expansion.

Definition 1. *Let A, B be two disjoint sets of equal seize. An A-B linkage is a set of $|A|$ edge-disjoint paths such that each vertex in $A \cup B$ is the end-point of exactly one of the paths. We say that A and B are linked in G if they there is an A-B linkage.*

As we saw previously, one can check if A, B are linked by an s-t flow computation. Moreover a linkage can be viewed as creating a perfect matching between A and B where the edges of the matching correspond to the paths in the A-B linkage.

Definition 2. *Let $G = (V, E)$ be a graph. A set X is well-linked in G if for any two $A, B \subseteq X$ with $|A| = |B|$, the sets A, B are linked.*

We had seen the following lemma previously.

Lemma 1. *A graph $G = (V, E)$ is an expander iff V is well-linked in G.*

What is the advantage of the notion of well-linkedness? To prove that G is *not* an expander it suffices to exhibit an equal-sized bipartition (A, B) of V and one can check via an s-t flow computation if A, B are not linked. Of course it is much easier to exhibit a cut S which is not expanding. However, our goal here is to do the following. To verify that G is an expander we could generate several bipartitions (A_i, B_i), $i \in [k]$ and check each of them. If we find a violating linkage then we have found that G is not an expander. Otherwise we create some perfect matchings $M_1, M_2, \ldots, M_k$ where each M_i is routable in G. Let $H = (V, \sum_{i=1}^{k} M_i)$ be the multi-graph obtained by the union of the matchings. Our goal is to find a small number of bipartitions and corresponding perfect matchings in some *adaptive* fashion such that H becomes an expander. If k is small enough then we have shown that an expander H can be "embedded" in G with congestion k since each M_i is routable in G. Thus, we will certify that G is a $\frac{1}{k}$-expander. Thus, our goal is to minimize k while ensuring that H is an expander.

In this context we recall the following fact. Suppose we take the union of 3 random perfect matchings on V. Then with high probability the resulting graph is an expander. However, well-linkedness cannot guarantee an arbitrary matching between A and B but only that *some* perfect matching exists — it is not under our control. The second issue is how to certify that H is an expander. For this purpose KRV show that H can route uniform multicommodity flow: a demand of $1/n$ between each ordered pair of vertices (u, v). This guarantees that H is an expander. Other methods use different ways to certify that H is an expander (for instance one can use spectral methods via Cheeger's inequality etc).

The Game: KRV set up the creation of H as an adaptive game between two players: a Cut Player and a Matching Player. The game goes in rounds. In each round the Cut Player generates a bipartition A_i, B_i of V (we will assume that these are equi-sized partitions through out this section). The Matching Player is required to output a perfect matching M_i between A_i and B_i. Let $H_i = (V, M_1 + \ldots + M_i)$. The goal of the Cut Player is to make H_i an expander as quickly as possible and the goal of the Matching Player is to delay it as much as possible.

Theorem 2. *There is an randomized adaptive strategy for the Cut Player such that H is an $\frac{1}{2}$-expander with high probability after $k = O(\log^2 n)$ rounds. Moreover, the certificate of H's expansion is a feasible routing of the uniform multicommodity flow over V.*

We leave the proof of the following corollary as an exercise since we already informally described the details earlier.

Corollary 3. *There is a randomized algorithm, that given G, either proves that G is not an expander or with high probability certifies that it is an $\Omega(\frac{1}{\log^2 n})$-expander using $O(\log^2 n)$ single-commoditly flow instances on G. Via binary search, there is a randomized algorithm for sparsest cut via $O(\log^3 n)$ single-commodity flows.*

The proof also yields the following alternate proof of the flow-cut gap although it is slightly weaker than the $O(\log n)$-bound we saw previously.

Corollary 4. *The multicommodity flow-cut gap for uniform instances is $O(\log^2 n)$.*

2.1 The Cut Player Strategy

We now describe the cut-player strategy. This is inspired by random-walk ideas but we will not explicitly mention the motivation and instead think of routing uniform multicommodity flow. The algorithm explicitly maintains a vector $b_u \in [0,1]^n$ for each vertex $u \in V$ such that $\sum_v b_u(v) = 1$. The entry $b_u(v)$ is the amount of u's flow that has reached v so far. Initially $b_u(v) = 1$ for $v = u$ and 0 if $v \neq u$ since all of u's one unit of flow is at u. The goal is to eventually have $b_u(v) \simeq 1/n$ for every u, v which corresponds to routing the (directed) uniform multicommodity flow. We will let b_u^i denote the vector for u after i iterations of the game.

For each u we also maintain a vector $x_u = b_u - \mathbf{1}/n$ (subtracts $1/n$ from each entry) which measures the distance of b_u to the target.

In the KRV game the cut player generates a partition (A_i, B_i) based only on the vectors $x_u i - 1$, $u \in V$. The Matching Player generates some M_i. The Cut Player than mixes the flow. in a very simple way. For each edge $uv \in M_i$ it sets

$$b_u^i = b_v^i = \frac{1}{2}(b_u^{i-1} + b_v^{i-1})$$

which also implies that

$$x_u^i = x_v^i = \frac{1}{2}(x_u^{i-1} + x_v^{i-1})$$

In other words for each matched pair $uv \in M_i$ we average their flow vectors and assign them both the same vector.

Claim 5. *We have $\sum_v b_u^i = 1$ for all i. Suppose H_{i-1} routes the demand matrix implied by the vectors b_u^{i-1}, $u \in V$. Then H_i routes the demand matrix implied by the the vectors b_u^i, $u \in V$.*

Proof. The first part is easy to verify.

Note that H_i differs from H_{i-1} via the matching M_i. For each $uv \in M_i$ we use the edge to exchange the flow at u and flow at v — $1/2$ capacity to send u's vector to v and $1/2$ capacity to send v's vector to u. ∎

The potential: To keep track of progress and to guide the generation of the cuts, the Cut Player maintains a potential

$$\phi = \sum_{u \in V} \|x_u\|^2 = \sum_u \sum_v \left(b_u(v) - \frac{1}{n}\right)^2.$$

We let ϕ_i denote the potential after i iterations.

Claim 6. $\phi_0 = n - 1$.

Proof. Recall that b^0_u has only 1 in the row for u and 0's else where. Thus $\|x^0_u\|^2 = (1-1/n)^2 + (n-1)/n^2 = 1 - 1/n$ and hence $\phi_0 = n(1-1/n) = n-1$. ■

Following is easy to see.

Claim 7. *Suppose $\phi_i \leq 1/(4n^2)$. Then $b^i_u(v) \geq \frac{1}{2n}$ for every u, v.*

Thus, if $\phi_i \leq 1/(4n^2)$ then we have effectively routed a near-uniform multicommodity flow in H_i and H_i is an expander. We leave the following as an exercise.

Claim 8. *Suppose $\phi_i \leq 1/(4n^2)$ then H_i is a $\frac{1}{2}$-expander.*

Reducing the potential via a matching: Based on the preceding claims, the goal of the Cut Player is to reduce the potential from the initial potential of n to $1/(4n^2)$. For this KRV prove the following.

Lemma 9. *Given current flow vectors $b^{i-1}_u, x^{i-1}_u, u \in V$ there is a randomized algorithm that produces a partition (A_i, B_i) such that for* any *perfect matching M_i between A_i and B_i, the potential after mixing according to M_i satisfies the following:*

$$E[\phi_i] \leq (1 - \frac{1}{c \log n})\phi_{i-1} + O(\frac{1}{n^{10}})$$

where c is some sufficiently large fixed constant.

Note that the algorithm uses only the information about the flow routed so far and nothing about the previous matchings. The preceding lemma immediately imply that the Cut Player can finish the game in $O(\log^2 n)$ rounds because it needs to reduce the potential from n to $1/(4n^2)$, and it can reduce the potential by a multiplicative $(1 - 1/(c \log n))$ factor in each iteration.

Thus, it remains to prove the lemma. To motivate the cut-generation strategy we work backwards in the analysis to first see what effect does a specific matching M have in reducing the potential. The lemma below

Lemma 10. *Suppose M_i is a perfect matching on V in iteration i. Then the change/reduction in potential is given by:*

$$\phi_i = \phi_{i-1} - \frac{1}{2} \sum_{uv \in M_i} \left\|x^{i-1}_u - x^{i-1}_v\right\|^2.$$

Note that the formula is exact and clean.

Proof. To simplify the notation we drop superscripts and use y_u for the vector for u after mixing according to matching M_i, and x_u at start of iteration i. We let $M = M_i$. We have $\phi_i = \sum_{u \in V} \|y_u\|^2$ and $\phi_{i-1} = \sum_{u \in V} \|x_u\|^2$.

Fix a vertex $u \in V$ and say it is matched with v in M. Then $y_u = \frac{1}{2}(x_u + x_v)$. Therefore,

$$\begin{aligned}
\|y_u\|^2 &= \left\|\frac{1}{2}(x_u + x_v)\right\|^2 \\
&= \frac{1}{4}\|x_u\|^2 + \frac{1}{4}\|x_v\|^2 + \frac{1}{2}\langle x_u, x_v \rangle \\
&= \frac{1}{2}\|x_u\|^2 + \frac{1}{2}\|x_v\|^2 - \frac{1}{4}\|x_u - x_v\|^2
\end{aligned}$$

Summing up over all $u \in V$ and using the fact that M is perfect matching we obtain the desired claim:

$$\phi_i = \phi_{i-1} - \frac{1}{2} \sum_{uv \in M_i} \left\| x_u^{i-1} - x_v^{i-1} \right\|^2.$$

■

Remark 11. *The preceding proof did not use any property of the vectors x_u. We only used the fact that the new vector y_u was defined by averaging with its partner in the perfect matching M. Thus, this is a purely geometric theorem.*

From the preceding lemma one sees that the potential reduction for an edge uv is proportional to $\|x_u - x_v\|^2$ which is the square of the Euclidean distance between x_u and x_v. Thus the Cut Player should somehow force the Matching Player to pair vertices that are far apart. Note however that the Cut Player can only give a partition (A_i, B_i) and the Matching Player has control of the actual matching it generates. To draw inspiration we take a geometric view from the preceding lemma. What happens if the vectors x_u are in one dimension? In this case we think of each x_u as a scalar and it corresponds to a point on the real line. What is a good cut strategy? One natural idea would be to sort the points on the line and pick the first half as A and second half as B. We prove the following lemma which shows that the potential reduction is substantial in this case!

Lemma 12. *Suppose $z_u, u \in V$ are scalars which correspond to points on the real line and suppose $\sum_{u \in V} z_u = 0$. Let A, B be a partition of V where A corresponds to the first half of V in sorted order of z_u values and B the second half. Then, for* any *perfect matching M between A and B, we have*

$$\sum_{uv \in M} (z_u - z_v)^2 \geq \sum_{u \in V} z_u^2.$$

Proof. Suppose we sort the points on the line and let $\beta \in \mathbb{R}$ which corresponds to the median. Hence $z_u \leq \beta$ for all $u \in A$ and $z_u \geq \beta$ for all $u \in B$. Let M be any perfect matching between A and B. Fix a matched pair $uv \in M$. Then $(z_u - z_v)^2 = (z_u - \beta - (z_v - \beta))^2$. Since $z_u \leq \beta$ and $z_v \geq \beta$, we have $(z_u - \beta - (z_v - \beta))^2 = (|z_u - \beta| + |z_v - \beta|)^2$. Summing up over all pairs in M,

$$\begin{aligned}
\sum_{uv \in M} \sum_{uv \in M} (z_u - z_v)^2 &= \sum_{uv \in M} (|z_u - \beta| + |z_v - \beta|)^2 \\
&= \sum_{u \in V} z_u^2 - 2\beta \sum_{u \in V} z_u + n\beta^2 \\
&\geq \sum_{u \in V} z_u^2.
\end{aligned}$$

We used in the fact that $\sum_{u \in V} z_u = 0$ in the final inequality. ■

Algorithm for Cut Player via Random Projection: The final ingredient to use Lemmas 10 and the insight via the one-dimensional case in Lemma 12. We have vectors $x_u, u \in V$ where each $x_u \in \mathbb{R}^n$ and $\sum_v x_u(v) = 0$. We reduce the problem to the one dimensional case by using the well-known idea of random projections that map points in high dimensions to a line. The following lemma can be shown by properties of Guassian random variables.

Lemma 13. *Let $g \sim \mathcal{N}(0,1)^n$-dimensional Guassian vector and let $x \in \mathbb{R}^n$. Let $z = \langle g, x \rangle$. Then*

- $E[z] = 0$
- $E[z^2] = \|x\|^2$
- *For any $C \geq 1$ there is some constant $c > 1$ such that $\mathbf{P}[z^2 \geq c \log n \cdot \|x\|^2] \leq \frac{1}{n^C}$.*

The Cut Player algorithm in iteration i is now the following.

- Pick a random Guassian vector g. For each $u \in V$ let $z_u = \langle g, x_u^{i-1} \rangle$.
- Sort V according to z_u values and let A_i be the first half and B_i be the second half.
- Give the partition (A_i, B_i) to the Matching Player.
- Let M_i be the matching returned.
- For each $uv \in M_i$ set $x_u^i = x_v^i = \frac{1}{2}(x_u^{i-1} + x_v^{i-1})$.

Note that the partition (A_i, B_i) is random.

Lemma 14. *Let M_i be any perfect matching between A_i and B_i. Then*

$$E[\sum_{uv \in M_i} (z_u - z_v)^2] \geq \phi_{i-1}.$$

Proof. Using Lemma 12, we see that

$$\sum_{uv \in M_i} (z_u - z_v)^2 \geq \sum_{u \in V} z_u^2 - 2\beta \sum_{u \in V} z_u$$

where β is the median of the numbers $z_u, u \in V$. Taking expectations on both sides (since the z values are random based on the choice of the random vector g) and using Lemma 13 (first two properties) we have

$$E[\sum_{uv \in M_i} (z_u - z_v)^2] \geq E[\sum_{u \in V} z_u^2] - 2\beta E[\sum_{u \in V} z_u] \geq \sum_{u \in V} \|x_u\|^2 = \phi_{i-1}.$$

■

Note that $(z_u - z_v)^2 = \langle g, x_u^{i-1} - x_v^{i-1} \rangle^2$ by Lemma 13 we have with high probability, $\langle g, x_u^{i-1} - x_v^{i-1} \rangle^2 \leq c \log n \cdot \left\| x_u^{i-1} - x_v^{i-1} \right\|^2$. For now, assume that this happens deterministically. Then, via the preceding lemma and Lemma 10

$$E[\phi_{i-1} - \phi_i] = \frac{1}{2} E[\sum_{uv \in M_i} \left\| x_u^{i-1} - x_v^{i-1} \right\|^2] \geq \frac{1}{c \log n} E[\sum_{uv \in M_i} (z_u - z_v)^2] \geq \frac{1}{c \log n} \phi_{i-1}.$$

This implies that

$$E[\phi_i] \leq (1 - \frac{1}{c \log n}) \phi_{i-1}.$$

The argument is not complete because we converted the high probability bound into a deterministic guarantee in the above inequalities while using expectations when needed. To overcome this we need to do a little bit of extra work which is why we get the weaker bound with a slight additive bound.

2.2 Background on Guassian Random Variables and Random Projections

A Guassian/Normal random variable with mean μ and variance σ^2, denoted by $\mathcal{N}(\mu, \sigma^2)$ is a one dimensional probability distribution with density function $f(x) = \frac{1}{\sqrt{2\phi\sigma^2}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$. It is a symmetric distribution around μ. We are particularly interested in the standard Normal distribution $\mathcal{N}(0,1)$ with mean 0 and variance 1. A d-dimensional (standard) Guassian random variable, denoted by $\mathcal{N}(0,1)^d$ is a d-dimensional random vector where each coordinate is an *independent* random variable distributed as $\mathcal{N}(0,1)$. Suppose Z is a d-dimensional Guassian random variable. Then its density function is $(\frac{1}{\sqrt{2\pi}})^d e^{-(\sum_{i=1}^d x_i^2)/2}$. Note that it is centrally symmetric.

A simple and powerful property about Guassian random variables is that the sum of (two or more) independent Guassian random variables is distributed as a Guassian random variable.

Lemma 15. *Suppose $Z_1 \sim \mathcal{N}(\mu_1, \sigma_1^2)$ and $Z_2 \sim \mathcal{N}(\mu_2, \sigma_2^2)$ are independent random Guassian random variables. Then $Z = Z_1 + Z_2$ is distributed according to $\mathcal{N}(\mu_1 + \mu_2, \sigma_1^2 + \sigma_2^2)$.*

Corollary 16. *Let X and Y be independent random variables. Suppose $X \sim \mathcal{N}(0,1)$ and $Y \sim \mathcal{N}(0,1)$. Let $Z = aX + bY$ where a, b are arbitrary real numbers. Then $Z \sim \mathcal{N}(0, a^2 + b^2)$.*

Concentration of Guassian random variable and its square: The Guassian random variable is symmetric around its mean and decays exponentially. One can ask for tail bounds and the following is well known. Suppose $Z \sim \mathcal{N}(0, \sigma^2)$.

$$\mathbf{P}[Z > \beta] \leq e^{\frac{-\beta}{2\sigma^2}}$$

Suppose Z is a Guassian random variable $\mathcal{N}(0, \sigma^2)$. We consider Z^2 the square of the Guassian random variable. This is now a positive random variable. It has the χ^2 distribution. Note that its mean is σ^2. It also exhibits strong concentration arounds its mean. The following upper tail bound is known when $Z \sim \mathcal{N}(0,1)$

$$\mathbf{P}[Z^2 \geq 1 + 4t] \geq e^{-t}$$

Using the preceding properties one can prove Lemma 13.

References

[AK07] Sanjeev Arora and Satyen Kale. A combinatorial, primal-dual approach to semidefinite programs. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 227–236, 2007.

[ARV09] Sanjeev Arora, Satish Rao, and Umesh Vazirani. Expander flows, geometric embeddings and graph partitioning. *Journal of the ACM (JACM)*, 56(2):1–37, 2009.

[CKL+22] Li Chen, Rasmus Kyng, Yang P Liu, Richard Peng, Maximilian Probst Gutenberg, and Sushant Sachdeva. Maximum flow and minimum-cost flow in almost-linear time. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 612–623. IEEE, 2022.

[KRV09] Rohit Khandekar, Satish Rao, and Umesh Vazirani. Graph partitioning using single commodity flows. *Journal of the ACM (JACM)*, 56(4):1–15, 2009.

[LR99] T. Leighton and S. Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *Journal of the ACM (JACM)*, 46(6):787–832, 1999. Conference version is from 1988.

[OSVV08] Lorenzo Orecchia, Leonard J Schulman, Umesh V Vazirani, and Nisheeth K Vishnoi. On partitioning graphs via single commodity flows. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 461–470, 2008.

[Pen16] Richard Peng. Approximate undirected maximum flows in o (m polylog (n)) time. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 1862–1867. SIAM, 2016.