University of Michigan–Ann Arbor

Department of Electrical Engineering and Computer Science

EECS 498 004 **Advanced Graph Algorithms**, Fall 2021

Lecture 15.0: The Cut-Matching Game: Randomized Cut Player

October 14 & 21, 2021

Instructor: Thatchaphol Saranurak

Scribe: Yi Tang

Recall: the Cut-Matching Game

In the cut-matching game, there are two players: the cut player and the matching player. The game starts with an empty graph G_0 with vertices V ($|V| = n$). In each round i, the cut player chooses a cut (A_i, B_i) (assuming $|A_i| \le |V|/2$), the matching player then chooses a maximal matching M_i between A_i and B_i (with size $|A_i|$), and the game sets $G_i = G_{i-1} \cup M_i$. The game ends when G_i is an expander (say an $\Omega(1/\text{polylog}(n))$-expander). Denote the total number of rounds by I and $G = G_I$.

The objective of the cut player is to minimize I and get an expander as fast as possible by cleverly choosing the cuts, while the matching player does the opposite and tries to delay the process as much as possible. We consider strategies/algorithms for the cut player under completely adversarial matching player.

The (lazy) random walk based on M_i is useful in analyzing the game strategies. In each step/round i of the random walk, from vertex u, if $u \notin M_i$ then the walk stays put; otherwise if $(u, v) \in M_i$ then the walk stays put with probability 1/2 and goes to v with probability 1/2.

Let $p_i(u, v)$ be the probability of reaching v at step i in the random walk starting from u, and $\mathbf{P}_i \in \mathbb{R}^{V \times V}$ be the matrix with value $p_i(u, v)$ at entry (u, v). Also let $\mathbf{M}_i$ be the lazy random walk matrix for matching M_i (see Figure 1 for the structure of $\mathbf{M}_i$). Then $\mathbf{P}_0 = \mathbf{I}$, and $\mathbf{P}_i = \mathbf{P}_{i-1}\mathbf{M}_i = \mathbf{M}_1 \cdots \mathbf{M}_i$. Note that $\mathbf{M}_i$ is symmetric and sparse and has only $\Theta(n)$ non-zero entries.

6 Randomized Near-Linear-Time Algorithm

Theorem 6.1 ([KRV09]). *There is a randomized $O(n \log^4 n)$-time algorithm for the cut player so that after $I = O(\log^2 n)$ rounds, $\Psi(G) = \Omega(1)$ (and thus $\Phi(G) = \Omega(1/\log^2 n)$).*

Proof. The randomized algorithm repeat the following subroutine I times: on round i,

- Sample a random (unit) vector $\mathbf{r}_i \in \mathbb{R}^V$ satisfying $\mathbf{r}_i \perp \mathbf{1}$.
- Compute $\mathbf{x} = \mathbf{P}_i^\top \mathbf{r}_i = \mathbf{M}_i \cdots \mathbf{M}_1 \mathbf{r}_i \in \mathbb{R}^V$.
- Return the *median cut* A_i that contains the $n/2$ vertices whose corresponding values x_u in $\mathbf{x}$ are the smallest.

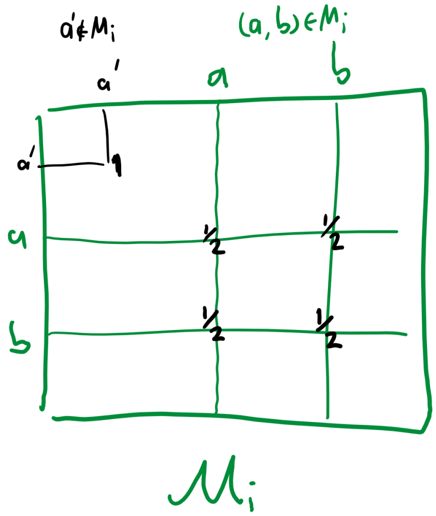

Figure 1: Structure of the (lazy) random walk matrix $\mathbf{M}_i$ for matching M_i.

The correctness and efficiency of the algorithm will be analyzed throughout this section. □

The randomized algorithm still closely follows the heuristics of choosing a *balanced sparse cut*. Firstly, the median cut is always a balanced cut. Moreover, if there exists a sparse cut S, then similar to the situation in the power method (though here the random walk mixes by merely M_i instead of the entire graph in round i), the values x_u in $\mathbf{x}$ for $u \in S$ and $u \in V \setminus S$ would be well-separated with high probability, and hence the median cut would contain the sparse cut S.

6.1 ℓ_2 Potential

For vertex u and round i, define the ℓ_2 *potential* by

$$\Pi_i(u) = \|\mathbf{P}_i(\cdot, u) - \mathbf{1}/n\|_2^2 .$$

Also define the total ℓ_2 potential by $\Pi_i = \sum_u \Pi_i(u)$.

Note that $\Pi_i(u) \geq 0$, and the equality holds for and only for uniform $\mathbf{P}_i(\cdot, u)$, so the potential measures how well-spread the distribution $\mathbf{P}_i(\cdot, u)$ is.

By calculation, $\Pi_0(u) = (1 - 1/n)^2 + (n-1)/n^2 = (n-1)/n$, and $\Pi_0 = n - 1$.

6.2 Small Potential Implies Done

Proposition 6.2. *If $\Pi_I \leq 1/(4n^2)$, then $\Psi(G) \geq 1/4$.*

Proof. Note that Π_I is a sum of squares, so if $\Pi_I \leq 1/(4n^2)$, then $p_I(u, v) \geq 1/n - 1/(2n) = 1/(2n)$ for all u, v. Then by Lemma 3.1 in the previous lecture, $K/2 \preceq^{\text{flow}} G$ and thus $K/2 \preceq^{\text{cut}} G$. This implies that for any cut S,

$$|\partial_G S| \geq \frac{|S| \cdot |V \setminus S|}{2n} \geq \frac{1}{4} \min\{|S|, |V \setminus S|\} .$$

I.e., $\Psi(G) \geq 1/4$. □

6.3 Potential Reduction = Total Squared Distance between Matched Vertices

Lemma 6.3. *The potential decrease in round i is precisely*

$$\Pi_{i-1} - \Pi_i = \frac{1}{2} \sum_{(u,v)\in M_i} \|\mathbf{P}_{i-1}(\cdot, u) - \mathbf{P}_{i-1}(\cdot, v)\|^2 .$$

Proof. Let $\mathbf{x}_{i,u} = \mathbf{P}_i(\cdot, u) - \mathbf{1}/n$. Then $\Pi_i = \sum_u \|\mathbf{x}_{i,u}\|^2$. For $u \notin M_i$, $\mathbf{x}_{i,u}$ is the same as $\mathbf{x}_{i-1,u}$ and the potential for u remains the same. For $(u, v) \in M_i$, the potential decrease for u, v is $\|\mathbf{x}_{i-1,u}\|^2 + \|\mathbf{x}_{i-1,v}\|^2 - 2\|(\mathbf{x}_{i-1,u} + \mathbf{x}_{i-1,v})/2\|^2$. Observe the identity $a^2+b^2-2((a+b)/2)^2 = (a-b)^2/2$. By applying the identity coordinate-wise, the potential decrease for u, v is precisely $\|\mathbf{x}_{i-1,u} - \mathbf{x}_{i-1,v}\|^2/2$. The claim follows by summing over all $(u, v) \in M_i$. □

6.4 Intuition of the Remaining Analysis

Imagine $\mathbf{P}_i \in \mathbb{R}^{V\times V}$ as an embedding of the vertices in the n-dimensional space, with u mapped to $\mathbf{P}_i(\cdot, u)$. By Lemma 6.3, if the points for the vertices are "well-separated" (see the left of Figure 2), and the cut is a good separation, then there would be large potential decrease (no matter how the matching behaves) as the distances between vertices on two sides are large.

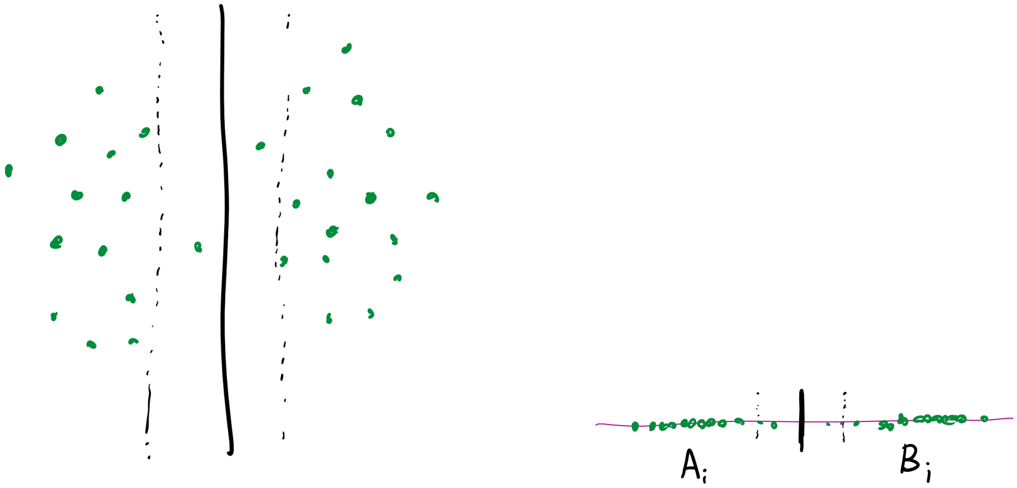

Figure 2: Left: Example of well-separated points, where the bold line is a good separation. Right: Random projection preserves distances well, so a good separation after projection is also a good separation before projection.

For efficiency, instead of computing a good separation in the n-dimensional space, we project onto a 1-dimensional subspace in a random direction $\mathbf{r}$ (see the right of Figure 2). This gives $\mathbf{x} = \mathbf{P}_i^\top \mathbf{r}$, with u mapped to $x_u = \mathbf{r}^\top \mathbf{P}_i(\cdot, u)$. Random projection has the property of preserving pairwise distances well. So the median cut after projection should approximate a good separation.

6.5 Random Projection Preserves Distances

Lemma 6.4 (Gaussian behavior of projections). *For any vector $\mathbf{v} \in \mathbb{R}^d$ and random unit vector $\mathbf{r} \in \mathbb{S}^{d-1}$,*

1. $\mathrm{E}_{\mathbf{r}}[(\mathbf{r}^\top \mathbf{v})^2] = \|\mathbf{v}\|^2/d$;
2. $\Pr_{\mathbf{r}}[(\mathbf{r}^\top \mathbf{v})^2 \geq \alpha \cdot \|\mathbf{v}\|^2/d] \leq e^{-\alpha/4}$, *for all* $\alpha \leq d/16$.

Proof. By Johnson-Lindenstrauss lemma. □

Note that all the involved vectors $\mathbf{P}_i(\cdot,u) - \mathbf{1}/n$ as well as $\mathbf{P}_i(\cdot,u) - \mathbf{P}_i(\cdot,v)$ live in the $(n-1)$-dimensional subspace orthogonal to $\mathbf{1}$. So by constraining $\mathbf{r} \perp \mathbf{1}$, we can apply the above lemma to the $(n-1)$-dimensional subspace. Also note that by setting $\alpha = \Theta(\log n)$ the probability in the above lemma is inverse polynomial. Then we have the following lemma as a corollary.

Lemma 6.5. *It holds that:*

1. $\|\mathbf{P}_{i-1}(\cdot,u) - \mathbf{P}_{i-1}(\cdot,v)\|^2 = (n-1)\,\mathrm{E}[(x_u - x_v)^2]$, *and* $\|\mathbf{P}_{i-1}(\cdot,u) - \mathbf{1}/n\|^2 = (n-1)\,\mathrm{E}[x_u^2]$;
2. $\|\mathbf{P}_{i-1}(\cdot,u) - \mathbf{P}_{i-1}(\cdot,v)\|^2 \geq (n-1)/\Theta(\log n) \cdot (x_u - x_v)^2$ *with high probability* $1 - 1/\mathrm{poly}(n)$.

6.6 Median Cut is Good

Lemma 6.6. *With median cut,* $(n-1)\,\mathrm{E}[\sum_{(u,v)\in M_i}(x_u - x_v)^2] \geq \Pi_{i-1}$.

Proof. Suppose η is the median in $\mathbf{x}$. Note that with median cut, M_i is a perfect matching. Then by calculation,

$$\sum_{(u,v)\in M_i}(x_u - x_v)^2 \geq \sum_{(u,v)\in M_i}(x_u - \eta)^2 + (x_v - \eta)^2 = \sum_u (x_u - \eta)^2 = \sum_u x_u^2 - 2\eta\sum_u x_u + n\eta^2 \geq \sum_u x_u^2 \,.$$

where the last ineqaulity is because $\sum_u x_u = 0$ as $\mathbf{x} = \mathbf{P}_{i-1}^\top \mathbf{r} \perp \mathbf{1}$. Hence by Lemma 6.5, Item 1,

$$(n-1)\,\mathrm{E}\Big[\sum_{(u,v)\in M_i}(x_u - x_v)^2\Big] \geq (n-1)\,\mathrm{E}\Big[\sum_u x_u^2\Big] = \sum_u \|\mathbf{P}_{i-1}(\cdot,u) - \mathbf{1}/n\|^2 = \Pi_{i-1}\,.$$ □

6.7 Summary of Correctness

Putting together Lemma 6.3, Lemma 6.5, Item 2, and Lemma 6.6,

$$\begin{aligned}\mathrm{E}[\Pi_{i-1} - \Pi_i] &= \frac{1}{2}\,\mathrm{E}\Big[\sum_{(u,v)\in M_i}\|\mathbf{P}_{i-1}(\cdot,u) - \mathbf{P}_{i-1}(\cdot,v)\|^2\Big] \\ &\geq \frac{1}{\Theta(\log n)} \cdot (n-1)\,\mathrm{E}\Big[\sum_{(u,v)\in M_i}(x_u - x_v)^2\Big] - \frac{\mu}{\mathrm{poly}(n)} \\ &\geq \frac{1}{\Theta(\log n)} \cdot \Pi_i - \frac{\mu}{\mathrm{poly}(n)}\,.\end{aligned}$$

Here $\mu = \max_{\mathbf{r}}(n-1)/\Theta(\log n) \cdot \sum_{(u,v)\in M_i}(x_u - x_v)^2 \leq \Theta(n^2/\log n)$ (as $|x_u| \leq \|\mathbf{P}_i(\cdot,u)\|_2 \leq \|\mathbf{P}_i(\cdot,u)\|_1 = 1$). So with large enough hidden degree in the term $\mathrm{poly}(n)$ (which in turn comes from large enough hidden constant in $\alpha = \Theta(\log n)$), the term $\mu/\mathrm{poly}(n)$ remains $1/\mathrm{poly}(n)$.

Then we have

$$\mathrm{E}[\Pi_i] \leq \Big(1 - \frac{1}{\Theta(\log n)}\Big) \cdot \Pi_{i-1} + \frac{1}{\mathrm{poly}(n)}\,.$$

This implies after $I = O(\log^2 n)$ rounds, with high probability, $\Pi_I \leq 1/(4n^2)$ and thus $\Psi(G) \geq 1/4$.

6.8 Near-Linear Time

Note that $\mathbf{M}_i$ has $\Theta(n)$ non-zero entries. Then $\mathbf{x} = \mathbf{P}_i^\top \mathbf{r} = \mathbf{M}_i \cdots \mathbf{M}_1 \mathbf{r}$ can be computed in $O(n \cdot i)$ time. Moreover it takes $O(n \log n)$ time to sort and find the median cut in $\mathbf{x}$ (or actually $O(n)$ time to find the median cut without a thorough sorting). Hence the total running time of the randomized algorithm is $\sum_{i=1}^{I} O(n \cdot i) = O(nI^2) = O(n \log^4 n)$.

7 State of the Art

There is an improvement to the randomized algorithm where we have $\Psi(G) = \Omega(\log n)$ instead of $\Psi(G) = \Omega(1)$ at the end, while still using $I = O(\log^2 n)$ rounds (so that $\Phi(G) = \Omega(1/\log n)$ instead of $\Phi(G) = \Omega(1/\log^2 n)$), summarized in the following theorem.

000

Theorem 7.1 ([OSVV08])**.** *There is a randomized $\widetilde{O}(n)$-time algorithm for the cut player so that after $I = O(\log^2 n)$ rounds, $\Psi(G) = \Omega(\log n)$ (and thus $\Phi(G) = \Omega(1/\log n)$).*

We have seen a slow deterministic algorithm and a fast randomized algorithm, and there is also an algorithm which almost gives the best of both worlds, summarized in the following theorem.

Theorem 7.2 ([CGL+20])**.** *There is a deterministic $O(n^{1+o(1)})$-time algorithm for the cut player so that after $I = O(\log n)$ rounds, $\Phi(G) = \Omega(1/n^{o(1)})$.*

This leaves open the following question.

Question 7.3. Is there a deterministic $\widetilde{O}(n)$-time algorithm for the cut player so that after $I =$ polylog(n) rounds, $\Phi(G) = \Omega(1/\text{polylog}(n))$?

An affirmative answer to this question would improve a lot of $O(m^{1+o(1)})$-time deterministic algorithms to $\widetilde{O}(m)$.

Moreover, in the randomized algorithm, it is not clear how much precision we need for the random unit vector $\mathbf{r}$; in particular, the following question asks whether "one bit per dimension" could suffice.

Question 7.4. In the randomized algorithm, can we simply sample $\mathbf{r} \in \{-1, 1\}^V$ instead (and translate it so that $\mathbf{r} \perp \mathbf{1}$)?

8 Further Topics

8.1 Directed Graphs

One can generalize the cut-matching game to direct graphs, where in each round i:

- The cut player chooses two disjoint subsets of vertices A_i, B_i with the same size.
- The matching player chooses two perfect matchings $M_i^{\rightarrow}, M_i^{\leftarrow}$ in both directions A_i to B_i and B_i to A_i.
- Set $G_i = G_{i-1} \cup M_i^{\rightarrow} \cup M_i^{\leftarrow}$.

One can also define the *directed expansion* to be

$$\Psi_G(S) = \frac{\min\{|E(S, V \setminus S)|, |E(V \setminus S, S)|\}}{\min\{|S|, |V \setminus S|\}} .$$

Under this notion of expansion, the deterministic and the randomized algorithms can be generalized to direct graphs using the same idea: entropy-based deterministic algorithm [BPGS20] and ℓ_2-norm-based randomized algorithm [Lou10].

8.2 Lower Bounds

Theorem 8.1 ([OSVV08]). *In order to get $\Psi(G) \geq \Omega(1)$, we need $I \geq \Omega(\sqrt{\log n})$.*

Theorem 8.2 ([She09]). *In order to get $\lambda_2(\mathbf{L}_G) \geq \Omega(1)$, we need $I \geq \Omega(\log n / \log\log n)$.*

Question 8.3. How does the lower bound on I change when we require only $\Psi(G) \geq \Omega(1/\text{polylog}(n))$?

9 Exercises

Exercise 9.1. Improve the algorithms for the cut player so that the resulting graphs not only are expanders but also have approximately a given degree profile $\mathbf{d}$.

Solution. First consider the case where $\mathbf{d} = R \cdot \mathbf{1}$, where R is the number of rounds in the game, i.e. we want uniform-degree expanders. The randomized algorithm already satisfies this requirement as it always chooses balanced cuts. The deterministic algorithm can also be easily modified to satisfies this requirement: in each round i, instead of choosing a cut A_i decided by the algorithm, we choose a balanced cut $A'_i \supseteq A_i$ instead. Observe that this does not break the entropy-based analysis in Lemma 5.2 in the previous lecture.

For general $\mathbf{d}$, discretize it by $\mathbf{d} \approx R \cdot \mathbf{w}$. A first attempt could be to duplicate the vertices according to $\mathbf{w}$, run the (*uniform-degree*) cut-matching game for R rounds, and contract the vertices correspondingly at the end. However an issue is that there might be many edges among the duplicate vertices and the contraction would mess up the degree profile. Even if those edges are retained as self-loops, the running time of the cut player would depend on $\mathbf{d}(V)$, which could be huge and thus the dependency is undesired.

Instead, one can consider the weighted cut-matching game that exactly fits the purpose:

- The cut player finds a $\frac{1}{4}$-*balanced* sparse cut (A'_i, B'_i), where $\mathbf{w}(A'_i) \geq \frac{1}{4}\mathbf{w}(V)$, then chooses the cut (A_i, B_i) where $A_i \supseteq A'_i$ where $\mathbf{w}(A_i) = \mathbf{w}(B_i)$.
- The matching player chooses a $\mathbf{w}$-*matching* between A_i and B_i, which is a bipartite subgraph with degree profile $\mathbf{w}$.

The deterministic and the randomized algorithm can be generalized to this weighted case. For the randomized algorithm, we need to define the matrix $\mathbf{M}_i$ for lazy random walk on M_i in the way such that for each edge $\{a, b\} \in M_i$, add $1/(2\deg(a))$ to $\mathbf{M}_i(a, a), \mathbf{M}_i(a, b)$ and add $1/(2\deg(b))$ to $\mathbf{M}_i(b, b), \mathbf{M}_i(b, a)$; otherwise if $a \notin M_i$ then let $\mathbf{M}_i(a, a) = 1$. By running the $\mathbf{w}$-weighted cut-matching game for R rounds, an expander with approximate degree profile $\mathbf{d}$ can be obtained. The running time for the randomized algorithm is proportional to the maximum nonzero entries in $\mathbf{M}_i$, $i.e.O(\max_i \mathbf{w}(A_i) + n)$ whence the running time is $\widetilde{O}(\max_i \mathbf{w}(A_i) + n)$. □

Exercise 9.2. Improve the algorithms for the cut player so that they can handle matching players choosing *fractional matchings*. A fraction matching between A_i and B_i is a weighted bipartite graph where each edge can have fractional weight and the degrees of vertices in A_i ($|A_i| \leq |B_i|$) are exactly 1 and those of B_i are at most 1.

Solution. For the deterministic algorithm, we don't need to change anything except that we now find the balanced sparse cut in the weighted graph; and for randomized algorithm, we need to define the matrix $\mathbf{M}_i$ for lazy random walk on M_i in the way such that for each edge $\{a, b\} \in M_i$ with weight α, add $\alpha/2$ in entry $\mathbf{M}_i(a, a), \mathbf{M}_i(b, b), \mathbf{M}_i(a, b), \mathbf{M}_i(b, a)$; otherwise if a is not matched in M_i then let $\mathbf{M}_i(a, a) = 1$. This can be verified by redoing all the proofs, but conceptually, a factional matching is a convex combination of integral matchings, and thus the random walk over factional matchings can only mix faster than the random walk over integral matchings.

The running time of the (randomized) algorithm becomes $\widetilde{O}$(max # of edges in fractional matching) = $\widetilde{O}(m)$ instead of $\widetilde{O}(n)$, since the fractional matchings may contain more edges whence $nnz(\mathbf{M}_i)$ =4(# of edges in fractional matching)+$O(n) = O(m)$. □

References

[BPGS20] Aaron Bernstein, Maximilian Probst Gutenberg, and Thatchaphol Saranurak. Deterministic decremental reachability, SCC, and shortest paths via directed expanders and congestion balancing. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science*, pages 1123–1134. IEEE Computer Soc., Los Alamitos, CA, [2020] ©2020.

[CGL$^+$20] Julia Chuzhoy, Yu Gao, Jason Li, Danupon Nanongkai, Richard Peng, and Thatchaphol Saranurak. A deterministic algorithm for balanced cut with applications to dynamic connectivity, flows, and beyond. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science*, pages 1158–1167. IEEE Computer Soc., Los Alamitos, CA, [2020] ©2020.

[KRV09] Rohit Khandekar, Satish Rao, and Umesh Vazirani. Graph partitioning using single commodity flows. *J. ACM*, 56(4):Art. 19, 15, 2009. Preliminary version in STOC 2006.

[Lou10] Anand Louis. Cut-matching games on directed graphs. *CoRR*, abs/1010.1047, 2010.

[OSVV08] Lorenzo Orecchia, Leonard J. Schulman, Umesh V. Vazirani, and Nisheeth K. Vishnoi. On partitioning graphs via single commodity flows. In *STOC'08*, pages 461–470. ACM, New York, 2008.

[She09] Jonah Sherman. Breaking the multicommodity flow barrier for $O(\sqrt{\log n})$-approximations to sparsest cut. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science—FOCS 2009*, pages 363–372. IEEE Computer Soc., Los Alamitos, CA, 2009.