University of Michigan–Ann Arbor

Department of Electrical Engineering and Computer Science

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

Lecture 23: Boundary-Linked Decomposition

November 30, 2021

Instructor: Thatchaphol Saranurak

Scribe: Ashwin Sreevatsa

1 Formal definitions

1.1 Well-linked Sets

Intuition: **Well-linked sets** of graphs are subsets of the vertices of a graph that are well-connected.

Let $G = (V, E)$ be a graph and $T \subseteq V$ is a set of terminals. T is **α-well-linked** in G if, for any disjoint sets $A, B \subseteq T$, we have $\text{mincut}_G(A, B) \geq \alpha \min\{|A|, |B|\}$.

Note that for any cut $(S, \overline{S})$ in G, $\delta_G(S) = \text{mincut}_G(S, \overline{S}) \geq \text{mincut}_G(S \cap T, \overline{S} \cap T) \geq \alpha \min\{|S \cap T|, |\overline{S} \cap T|\}$.
Also note that V is α-well-linked $\iff \Psi(G) \geq \alpha$.

We can also introduce the concept of well-linkedness to edges. Consider a graph $G = (V, E)$, construct a **split graph** $G' = (V', E')$ from G by converting each edge $e = (u, v) \in E$ from graph G into $(u, x_e), (x_e, v)$ in G'. Note that $V' = V \cup \{\text{the set of split nodes of } G\}$, $E' = \{(u, x_e), (x_e, v) \text{ for each edge } e = (u, v) \in E\}$. (Figure 1)

Let $X_E = \{x_e | e \in E\}$ be the set of split nodes of G. Then, we can show that $\Phi(G) \geq \phi \iff X_E$ is ϕ-well-linked in G'.
We can also consider well-linked sets from the **flow perspective**. Let K be the **all-to-all demand** between T when K is the **d**-product graph, with $\text{d}(u) = 1$ if $u \in T$ and $\text{d}(u) = 0$ otherwise.

If $K \preccurlyeq^{\text{flow}} G$, then for any demand H where $V(H) = T$ and H has maximum degree $O(1)$, we have $H \preccurlyeq^{\text{deg}} O(1)K \implies H \preccurlyeq^{\text{flow}} O(1)K \preccurlyeq^{\text{flow}} O(1)G$.
This means that if the all-to-all demand between T is routable, then any demand between T where each node exchanges at most $O(1)$ units of flow must be routable with $O(1)$ congestion.

Exercise 1.1. Given a graph G and a vertex set T, show an algorithm for polylog(n)-approximating the well-linkedness of T.

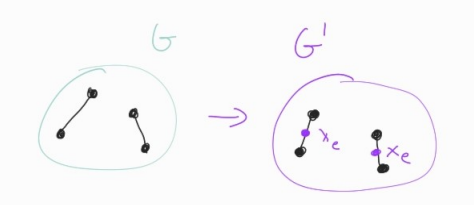

Figure 1: An example of a graph with split edges

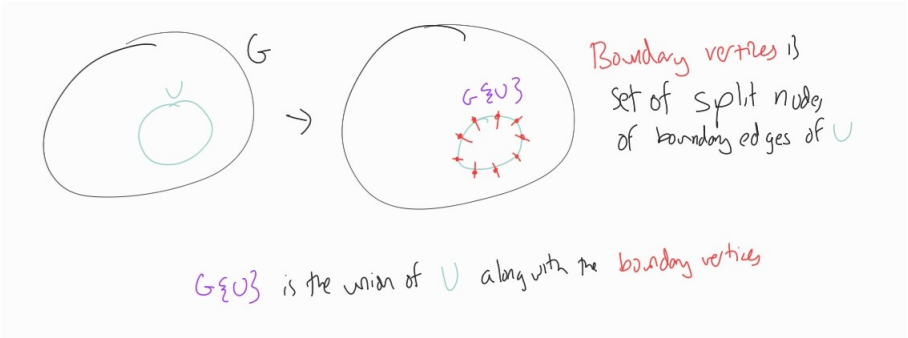

Figure 2: Boundary vertices of a graph with vertex subset U

Proof. We have that T is α-well-linked in $G \iff$ the all-to-all demand between T is routable in G with congestion $O(\frac{\log n}{\alpha})$ from a previous exercise. Since we have an algorithm for finding the minimum congestion routable flow for the all-to-all demand in T, and this value will simply be a log(n) approximation of the well-linkedness. □

1.2 Boundary-linked Graphs

Let $G = (V, E)$ be a graph with $U \subseteq V$ a subset of the vertices. Then $G\{U\}$ is the union of the subgraph U and its **boundary vertices** (figure 2). Formally, for every edge $e = (u, v) \in E$ with $u \in U$ and $v \in V/U$, create a new vertex x_e such that $(u, x_e), (x_e, v) \in E$ and $(u, x_e) \in G\{U\}$. (Figure 2)

Let $\partial_G \langle U \rangle = V(G\{U\}) \setminus U$ be the set of boundary vertices. Then, we say $G\{U\}$ is α-boundary-linked if $\partial_G \langle U \rangle$ is α-well-linked in $G\{U\}$.

An additional formulation of this notion of boundary-linkedness is with respect to flow:

- the all-to-all demand between $\partial_G \langle U \rangle$ is routable with congestion $\tilde{O}(1/\alpha)$, or equivalently
- constant-degree expanders on $\partial_G \langle U \rangle$ is embeddable into G with congestion $\tilde{O}(1/\alpha)$.

2 Boundary-linked Decomposition

Intuition: we want to take some induced subgraph $G\{U\}$ with boundary vertices and partition U into $U_1, U_2, ..., U_k$ to produce a new set of induced subgraphs $G\{U_1\}, G\{U_2\}, ..., G\{U_k\}$. This will

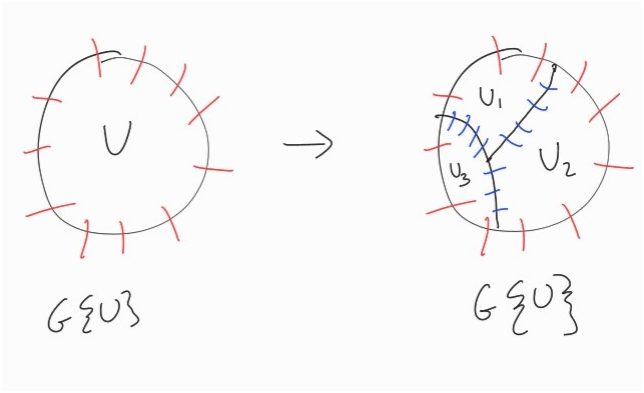

Figure 3: Boundary-linked Decomposition

be an α**-boundary-linked decomposition** (figure 3) if the new induced subgraphs need to satisfy 2 constraints:

1. $G\{U_i\}$ is α-boundary-linked
2. There aren't too many new boundary edges
   (a) Formally we want the total number of new boundary edges to be $\sum_i |\partial_G \langle U\rangle_i \backslash \partial_G \langle U\rangle| = c_{\text{bound}} \cdot \alpha |\partial_G \langle U\rangle|$
   (b) $c_{\text{bound}} = O(\log |\partial_G \langle U\rangle|)$ for existential result
   (c) $c_{\text{bound}} = n^{o(1)}$ for fast algorithms

2.1 Motivation: Vertex Sparsifiers

One application of this method of boundary-linked decomposition is for vertex sparsifiers (figure 4). In essence, we can 'compress' $G\{U\}$ into a new graph H by taking each set U_i and contracting it into a single vertex u_i in H.
As it turns out, H preserves cut size between all subsets of boundary vertices.

Theorem 2.1. *For any two set of boundary vertices $A, B \subseteq \partial_G \langle U\rangle$, we have*

$$\text{mincut}_{G\{U\}}(A, B) \leq \text{mincut}_H(A, B) \leq \frac{1}{\alpha} \cdot \text{mincut}_{G\{U\}}(A, B)$$

Proof. The first half of the inequality is easy to prove. For any (A, B) mincut in H, that cut must necessarily exist in $G\{U\}$ because graph H is a 'compression' of graph $G\{U\}$. So we have $\text{mincut}_{G\{U\}}(A, B) \leq \text{mincut}_H(A, B)$.

The second inequality is the harder inequality to prove. Let (X, Y) be an (A, B)-mincut in $G\{U\}$. We want to find another (A, B)-cut (X', Y') in H that is only blown up by a factor of $\frac{1}{\alpha}$. If this can be satisfied, then we have $\text{mincut}_H(A, B) \leq \frac{1}{\alpha} \cdot \text{mincut}_{G\{U\}}(A, B)$ and we're done.

The way we do this is by taking the (X, Y) cut, and for every set U_i that it crosses, redirect the

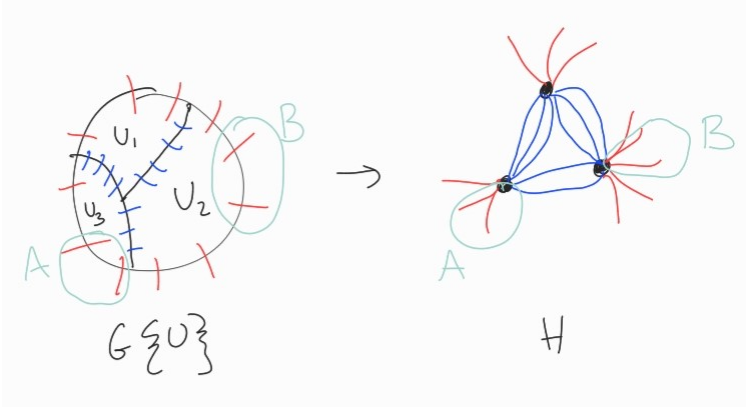

Figure 4: Vertex Sparsifier

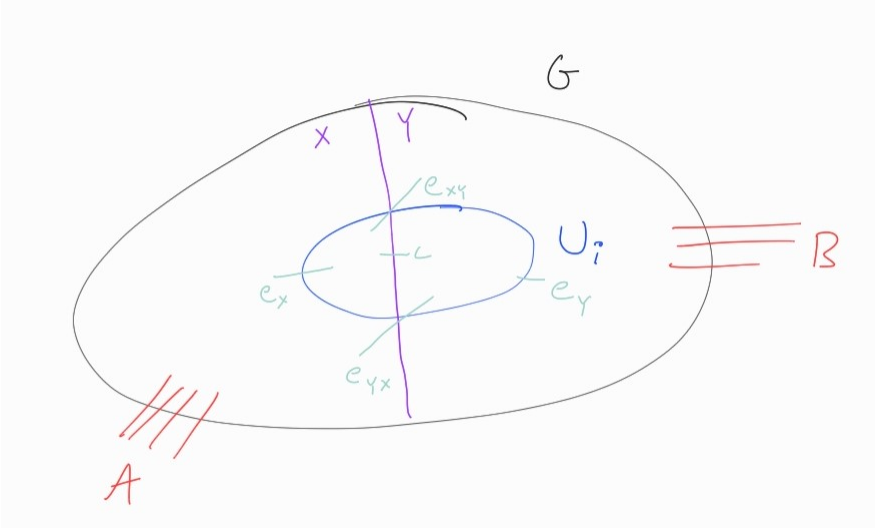

Figure 5: An example of a graph with split edges

cut so that it doesn't cross U_i. Since we can redirect this cut 2 different ways, choose the direction that cuts less edges.

Notice figure 5.

- $e_X = |E_G(X \cap U_i, X \setminus U_i)|$
- $e_Y = |E_G(Y \cap U_i, Y \setminus U_i)|$
- $e_{XY} = |E_G(X \cap U_i, Y \setminus U_i)|$
- $e_{YX} = |E_G(Y \cap U_i, X \setminus U_i)|$
- $c = |E_G(X \cap U_i, Y \cap U_i)|$

We have that $e_X + e_{XY}$ is the number of boundary vertices whose endpoint in U_i is on the X side, whereas $e_Y + e_{YX}$ is the number of boundary vertices whose endpoint in U_i is on the Y side. Assume without loss of generality that $e_X + e_{XY} \leq e_Y + e_{YX}$. Then, we want to route the (X', Y')

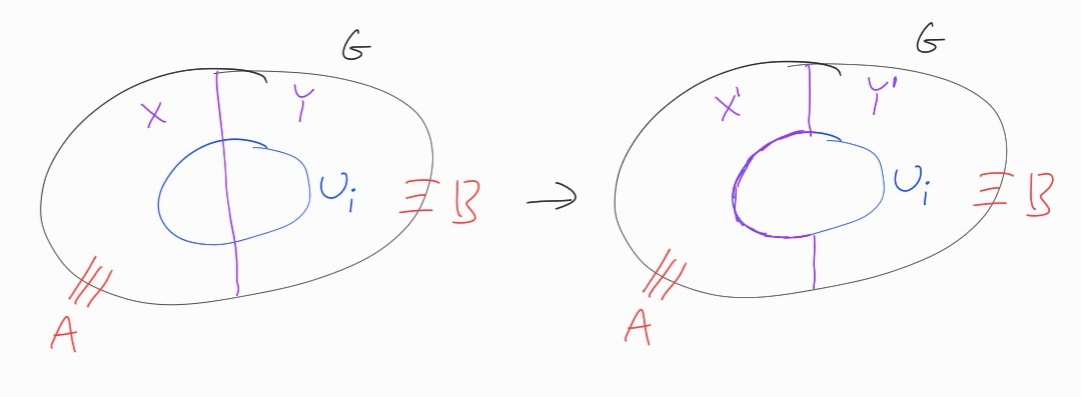

Figure 6: Boundary-linked decomposition algorithm (Algorithm 1)

cut on the X side of U_i.

Note that $e_X \leq e_X + e_{XY} \leq \frac{1}{\alpha}c$, where the second inequality holds by U_i being α-boundary-linked. So we have $E(X,Y) = c + e_{XY} + e_{YX} \geq \alpha \cdot e_X + \alpha \cdot e_{YX} = \alpha \cdot (e_X + e_{YX}) = \alpha \cdot E(X',Y')$. Then, $E(X',Y') \leq \frac{1}{\alpha}E(X,Y)$ and $\text{mincut}_H(A,B) \leq \frac{1}{\alpha} \cdot \text{mincut}_{G\{U\}}(A,B)$. □

How large would the graph H be?

Lemma 2.2. $|E(H)| = \underbrace{|\partial_G \langle U \rangle|}_{\text{old boundary}} + \underbrace{\sum_i |\partial_G \langle U_i \rangle \setminus \partial_G \langle U \rangle|}_{\text{new boundary}} = O(c_{\text{bound}}|\partial_G \langle U \rangle|)$.

From the construction of boundary-linked decompositions, we know that the number of new boundary edges $= \sum_i |\partial_G \langle U_i \rangle \setminus \partial_G \langle U \rangle| \leq c_{\text{bound}} \cdot \alpha |\partial_G \langle U \rangle|$, so the second equality will hold.

As a result, H is a vertex sparsifier of $G\{U\}$. We have reduced the number of vertices and edges to be proportional to $\partial_G \langle U \rangle$ while still approximately preserving all of the cuts for $\partial_G \langle U \rangle$.

2.2 Existence of Boundary-linked Decomposition

The generic algorithm for showing the existence of expander decomposition used the following idea:

- If there is no "sparse" cut, we are done
- Otherwise, find the "sparse" cut, cut the graph into 2 subgraphs, and recurse on both sides.

We can use this same algorithm for existence of boundary-linked decomposition, with "sparsity" defined via boundary-linkedness. (Figure 6, Algorithm 1)

We know that $G\{U_i\}$ will be α-boundary-linked for all i. What we need to do now is to bound the number of new boundary vertices $\sum_i |\partial_G \langle U_i \rangle \setminus \partial_G \langle U \rangle|$.

Goal: Ideally, we want $\sum_i |\partial_G \langle U_i \rangle \setminus \partial_G \langle U \rangle| = O(|\partial_G \langle U \rangle|)$.

Algorithm 1 Boundary-linked Decomposition Algorithm: Decomp($G\{U\}, \alpha$)

Require: $G\{U\}, \alpha$
  if $\partial_G \langle U \rangle$ is α-well-linked **then**
    return $\{U\}$
  else
    there must be a cut $(S, \overline{S}) \in G\{U\}$ where $\delta_G(S) < \alpha \min\{|S \cap \partial_G \langle U \rangle|, |\overline{S} \cap \partial_G \langle U \rangle|\}$
    return Decomp($G\{U \cap S\}, \alpha$) $\cup$ Decomp($G\{U \cap \overline{S}\}, \alpha$)
  end if

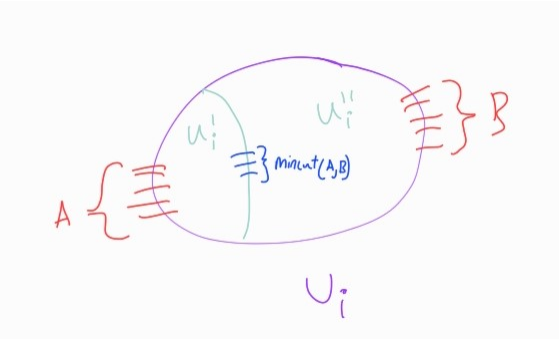

Figure 7: Sparse cut of U_i

Strategy:

- Put "money" into each vertex of the graph
- Every time we create a new boundary vertices, we will pay \$1
- **Invariant**: Each boundary vertex in $\partial_G \langle U_i \rangle$ has at least $4\alpha \log_{3/2}(\partial_G \langle U_i \rangle)$ dollars, for all *current* sets U_i. We will need to reallocate money at each step in order to maintain the invariant
- We will show that the initial money is enough to 'pay' for all of the new boundary vertices created

The point of all of this is to bound the number of new boundary vertices. Since we must spend \$1 for each new boundary vertex created, the total number of created boundary vertices is at most the total initial money, or $O(\alpha \cdot |\partial_G \langle U \rangle| \cdot \log |\partial_G \langle U \rangle|) = O(|\partial_G \langle U \rangle|)$. [Note that this requires $\alpha = O(\frac{1}{\log |\partial_G \langle U \rangle|})$]

Proof. Consider U_i in figure 7. We find a sparce cut in $G\{U_i\}$ and recurse on $G\{U_i'\}$ and $G\{U_i''\}$. We have

- $A = \partial_G \langle U \rangle_i' \cap \partial_G \langle U \rangle_i$
- $B = \partial_G \langle U \rangle_i'' \cap \partial_G \langle U \rangle_i$
- $C = \text{mincut}_{G\{U_i\}}(A, B)$

Assume w.l.o.g that $|A| \leq |B|$. We will use money from A to pay for the new boundary edges created.
First note that $|\partial_G \langle U_i' \rangle| = |A| + |C| \leq |A| + \alpha \cdot |A| = (1 + \alpha) \cdot |A| \leq \frac{4}{3}|A| \leq \frac{2}{3}|\partial_G \langle U_i \rangle|$.

(The first equality holds from how $\partial_G \langle U_i' \rangle$ is defined, the second inequality holds from the fact that U_i is not well-linked and C is a mincut, the third equality holds from algebra, the fourth inequality holds from $\alpha = O(\frac{1}{\log |\partial_G \langle U \rangle|})$, the final inequality holds from $|A| \leq |B|$ and $|A| + |B| = \partial_G \langle U_i \rangle$.)
This gives us $\log_{3/2} |\partial_G \langle U_i' \rangle| \leq \log_{3/2} |\partial_G \langle U_i \rangle| - 1$.

Based on our invariant, we know that for each $a \in A$, a has $4\alpha \log_{3/2} |\partial_G \langle U \rangle_i|$. After the partition into U_i', U_i'', a has a little bit of extra money (since the new graph has less boundary edges than before). a only needs $4\alpha \cdot |\log_{3/2}| \cdot (\partial_G \langle U_i' \rangle) \leq 4\alpha \cdot |\log_{3/2}| \cdot (\partial_G \langle U_i \rangle - 1) = 4\alpha \cdot |\log_{3/2}| \cdot (\partial_G \langle U_i \rangle) - 4\alpha$ dollars.So a has 4α extra dollars it can spend.

After cutting U_i, we collect all of the extra dollars from A, which will be $4\alpha|A|$. We will use this money for creating all of the boundary vertices and for maintaining the invariants on both sides of the cut (for both U_i' and U_i'').

The total money needed =

- $2|C| \longrightarrow$ (\$1 for each of the $|C|$ boundary edges in both U_i' and U_i'') +
- $|C| \cdot 4\alpha \log_{3/2}(\partial_G \langle U_i' \rangle) \longrightarrow$ (to maintain the invariant in U_i') +
- $|C| \cdot 4\alpha \log_{3/2}(\partial_G \langle U_i'' \rangle) \longrightarrow$ (to maintain the invariant in U_i'')

We have $2|C| + |C| \cdot 4\alpha \log_{3/2}(\partial_G \langle U_i' \rangle) + |C| \cdot 4\alpha \log_{3/2}(\partial_G \langle U_i'' \rangle) \leq 2|C| + |C| \cdot 8\alpha \log_{3/2}(\partial_G \langle U_i \rangle) \leq 2|C| + 2|C| = 4|C| \leq 4|A|\alpha$.
(The first inequality holds from the fact that the number of boundary edges of U_i is an upper bound on the number of boundary edges of U_i' and U_i'', the second inequality holds from $\alpha \leq \frac{1}{4 \log_{3/2}(\partial_G \langle U_i \rangle)}$, the third equality holds from algebra, the fourth inequality holds from U_i not being α-well-linked and (U_i', U_i'') being a mincut).

Since we have enough extra money to pay for all of the costs of adding boundary edges and maintaining invariants, we are done. In other words, $G\{U_i\}$ is α-boundary-linked for all i, and $\sum_i |\partial_G \langle U_i \rangle \setminus \partial_G \langle U \rangle| = O(\alpha |\partial_G \langle U \rangle| \log |\partial_G \langle U \rangle|)$. □

2.3 How to compute it fast?

Exercise 2.1. Give a polynomial time algorithm for boundary-linked decomposition with slightly worse guarantee.

1. Show that an algorithm that, given $G\{U\}$, either
   - reports that $\partial_G \langle U \rangle$ is $\alpha/\text{polylog}(n)$-well-linked, or.
   - finds a cut certifying that $\partial_G \langle U \rangle$ is not α-well-linked
2. Use this subroutine to get boundary-linked decomposition. The only change in the analysis is that when we stop the recursion at $G\{U_i\}$, we only guarantee that $G\{U_i\}$ is $\alpha/\text{polylog}(n)$-boundary-linked

Exercise 2.2. Use the cut-matching game framework to compute α-boundary-linked decomposition in almost-linear time, where $\alpha \geq 1/n^{o(1)}$ and $\sum_i |\partial_G \langle U_i \rangle \setminus \partial_G \langle U \rangle| = O(\alpha |\partial_G \langle U \rangle| n^{o(1)})$. *Hint*: Find most balanced sparse cut.

3 Boundary-linked Expander Decomposition

We've shown that we can partition graphs based on boundary-linkedness and based on expanders (expander decomposition). Can we do both at the same time?

Terminology: U is (α, ϕ)-linked in G if $G\{U\}$ is α-boundary-linked and $G\{U\}$ is a ϕ-expander.

If U is (α, ϕ)-linked, then

- Edges in $G\{U\}$ are $\frac{1}{\phi}$-well-linked. (The all-to-all demand between edges of $G\{U\}$ is routable in $G[U]$ with congestion $O(\frac{\log n}{\phi})$).
- Boundary edges in $G\{U\}$ are $\frac{1}{\alpha}$-well-linked. (The all-to-all demand between boundary edges of $G\{U\}$ is routable in $G[U]$ with congestion $O(\frac{\log n}{\phi})$).

We want to show an algorithm with the following characteristics:

- **Input:** $G\{U\}, \phi$
- **Output:** a partition $U_1, U_2, ..., U_k$ of U such that:
  - U_i is (α, ϕ)-linked in G (with $\alpha = \frac{1}{\Theta(\log n)}$)
  - $\sum_i |\partial_G \langle U_i \rangle| = O(|\partial_G \langle U \rangle| + \phi \text{vol}_G(U) \log m)$.

3.1 Induced subgraph with k-boundary-self-loops

To conclude that U is (α, ϕ)-linked, we introduce the following notion:

Let $G[U]^k$ denote an **induced subgraph with k-boundary-self-loops** (figure 8). What this means is that we start with the induced subgraph $G[U]$ and add k self-loops (or a self-loop with capacity k) on each boundary edge $e = (u, v) \in E(U, V \setminus U)$ with endpoint $u \in U$. (Figure TODO)

Lemma 3.1. If $G[U]^{\alpha/\phi}$ is a ϕ-expander, then U is (α, ϕ)-linked.

Proof. We can prove the cases separately.

$G\{U\}$ is a ϕ-expander because

- For any cut $(S, \overline{S})$ in $G\{U\}$, let $(S', \overline{S}') = (S \cap U, \overline{S} \cap U)$ be a cut in $G[U]^{\alpha/\phi}$.
- We have $\delta_{G\{U\}}(S) \geq \delta_{G[U]^{\alpha/\phi}}(S')$ but $\text{vol}_{G[U]}(S) \leq \text{vol}_{G[U]^{\alpha/\phi}}(S')$.
- So $\Phi_{G[U]}(S) \geq \Phi_{G[U]^{\alpha/\phi}}(S') \geq \phi$

$G\{U\}$ is α-boundary-linked.

- Let A, B be a partition of $\partial_G \langle U \rangle$.

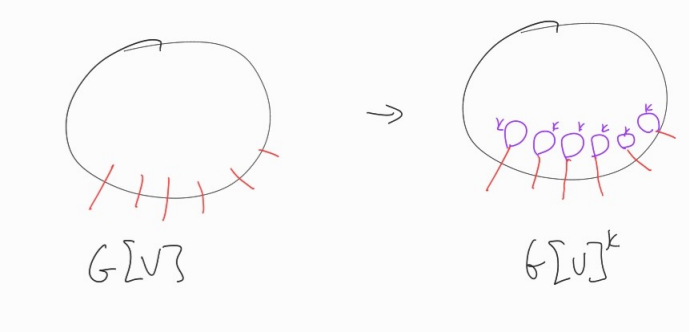

Figure 8: Induced subgraph with k-boundary-self-loops

- For any (A, B)-mincut $(S, \overline{S})$ in $G\{U\}$, let $(S', \overline{S}') = (S \cap U, \overline{S}' \cap U)$ be a cut in $G[U]^{\alpha/\phi}$.
- We have

$$\begin{aligned}
\text{mincut}_{G\{U\}}(A, B) &= \delta_{G\{U\}}(S) \\
&= \delta_{G[U]^{\alpha/\phi}}(S') \\
&\geq \phi \cdot \min\{\text{vol}_{G[U]^{\alpha/\phi}}(S'), \text{vol}_{G[U]^{\alpha/\phi}}(\overline{S}')\} \\
&= \phi \cdot \min\{\frac{\alpha}{\phi} \cdot |A|, \frac{\alpha}{\phi} \cdot |B|\} \\
&= \alpha \cdot \min\{|A|, |B|\}
\end{aligned}$$

□

3.2 Existence of Boundary-linked Expander Decomposition

The algorithm and analysis for boundary-linked expander decomposition will be very similar to that of the general boundary-linked decomposition.

Input:

- an induced subgraph $G\{U\}$ with boundary vertices
- a parameter ϕ.

Output: a partition of $U_1, \ldots, U_k$ of U such that

- $G[U_i]^{\alpha/\phi}$ is a ϕ-expander
  - $\alpha = 1/4 \log_{3/2}(\text{vol}(G[U]^{\alpha/\phi})) = \Theta(1/\log m)$
- $\sum_i |\partial_G \langle U_i \rangle \setminus \partial_G \langle U \rangle| = O(|\partial_G \langle U \rangle| + \phi \text{vol}_G(U) \log m)$

Analysis strategy:

- We'll put money on "edges" of graphs, and whenever we create a new boundary vertex, we pay \$1.
- **Invariant**: Each vertex u in $G[U_i]^{\alpha/\phi}$ has at least $\deg_{G[U_i]^{\alpha/\phi}}(u) \times 4\phi \log_{3/2}(\text{vol}(G[U_i]^{\alpha/\phi}))$ dollars.

Algorithm 2 Boundary-linked Expander Decomposition Algorithm: Decomp($G\{U\}^{\alpha/\phi}, \phi$)

Require: $G\{U\}^{\alpha/\phi}, \phi$
  if $G\{U\}^{\alpha/\phi}$ is ϕ-expander **then**
    return $\{U\}$
  else
    there must be a cut $(S, \overline{S}) \in G\{U\}$ that is a ϕ-sparse cut
    return Decomp($G\{U \cap S\}^{\alpha/\phi}, \phi$) $\cup$ Decomp($G\{U \cap \overline{S}\}^{\alpha/\phi}, \phi$)
  end if

- The total number of new boundary vertices is bounded by the total initial money, $O(|\partial_G \langle U \rangle| + \phi \mathrm{vol}_G(U) \log m)$.

Proof. Suppose we find a ϕ-sparse cut (U_i', U_i'') in $G[U_i]^{\alpha/\phi}$ and we recurse on $G[U_i']^{\alpha/\phi}$ and $G[U_i'']^{\alpha/\phi}$.

We have $c < \phi \min\{a, b\}$.

Define

- $a = \mathrm{vol}_{G[U_i]^{\alpha/\phi}}(U_i')$
- $b = \mathrm{vol}_{G[U_i]^{\alpha/\phi}}(U_i'')$
- $c = \delta_{G[U_i]^{\alpha/\phi}}(U_i')$

Assume w.l.o.g. that $a \le b$.
The volume of the bigger side does not increase: $\mathrm{vol}(G[U_i'']^{\alpha/\phi}) \le b + \frac{\alpha}{\phi} c \le b + a = \mathrm{vol}(G[U_i]^{\alpha/\phi})$
The volume of the smaller side decreases by a constant factor: $\mathrm{vol}(G[U_i']^{\alpha/\phi}) \le \frac{2}{3}\mathrm{vol}(G[U_i]^{\alpha/\phi})$ because

$$\begin{aligned} \mathrm{vol}(G[U_i']^{\alpha/\phi}) &\le a + \frac{\alpha}{\phi} c \\ &\le \frac{4}{3} a && \text{as } c \le \phi a \text{ and } \alpha \ll 1/3 \\ &\le \frac{2}{3} \mathrm{vol}(G[U_i]^{\alpha/\phi}). \end{aligned}$$

- So $\log_{3/2}(\mathrm{vol}(G[U_i']^{\alpha/\phi})) \le \log_{3/2}(\mathrm{vol}(G[U_i]^{\alpha/\phi}) - 1$.

After cutting U_i,

- we can collect $a \cdot 4\phi$ dollars (from endpoints of edges in $G[U_i]^{\alpha/\phi}$ incident to U_i').
- money needed for the new boundary vertices is at most
  - $2c \longrightarrow$ [\$1 per new boundary]
  - $\frac{\alpha}{\phi} c \cdot 4\phi \log_{3/2}(\mathrm{vol}(G[U_i']^{a/\phi})) \longrightarrow$ [invariant on $G[U_i']^{a/\phi}$]
  - $\frac{\alpha}{\phi} c \cdot 4\phi \log_{3/2}(\mathrm{vol}(G[U_i'']^{a/\phi})) \longrightarrow$ [invariant on $G[U_i'']^{a/\phi}$]
- So we have enough money to maintain the invariant.

This completes the proof.

- $G[U_i]^{\alpha/\phi}$ is a ϕ-expander for all i.
- $\sum_i |\partial_G \langle U_i \rangle \setminus \partial_G \langle U \rangle| = O(|\partial_G \langle U \rangle| + \phi \mathrm{vol}_G(U) \log m)$

□

Some additional resources here: [And10], [CC13], [GRST21].

References

[And10] Matthew Andrews. Approximation algorithms for the edge-disjoint paths problem via raecke decompositions. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 277–286. IEEE, 2010.

[CC13] Chandra Chekuri and Julia Chuzhoy. Large-treewidth graph decompositions and applications. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 291–300, 2013.

[GRST21] Gramoz Goranci, Harald Räcke, Thatchaphol Saranurak, and Zihan Tan. The expander hierarchy and its applications to dynamic graph algorithms. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2212–2228. SIAM, 2021.