Fall 2024, CS 598: Topics in Graph Algorithms
Homework 2
Due: 10/03/2024

Instructions and Policy: You can work in groups of up to two. Each group needs to submit only one solution. You need to indicate the names of the people you discussed a problem with. Solutions to most of these problems can be found from one source or the other. Try to solve on your own first, and cite your sources if you do use them.

Please write clearly and concisely. Refer to known facts. You should try to convince me that you know the solution, as quickly as possible.

For this home work submit solutions to Problems 1- 4. The other two are for you to read and try if you are interested.

Problem 1 Hypergraphs generalize graphs. A hypergraph $G = (V, E)$ consists of a finite set of vertices and set of hyper-edges E. Each hyper-edge e is a subset of V; in graphs each edge is a subset of size 2 while in a hypergraph the hyper-edges can have different sizes. See examples here `https://en.wikipedia.org/wiki/Hypergraph`. The parameters of a hypergraph are n the number of vertices, m the number of hyper-edges and $p = \sum_{e \in E} |e|$. Note that in graphs $p = 2m$ and hence we do not need to keep track of p. Each hypergraph can also be represented via a bipartite incidence graph $H = (V \cup E, F)$ where one side is V and the other side is E and a vertex v and a hyper-edge e are connected by an edge in H iff $v \in e$. We say that s and t are connected in G iff there is a path from s to t in H. We write $\delta_G(S) = \{e \mid e \cap S \neq \emptyset, e \cap (V - S) \neq \emptyset\}$ as the set of edges crossing a set S.

- **Not for submission:** Given a hypergraph and $s, t \in V$ an s-t cut is a set of hyper-edges $F \subseteq E$ such that s and t are not connected in $G - F$. Prove that if F is an s-t cut then there is some $S \subset V$ with $s \in S$ and $t \in V - S$ such that $\delta_G(S) \subseteq F$. This is mainly for your understanding.
- Suppose we have non-negative costs/capacities $c : E \to \mathbb{R}_+$ on the hyper-edges. Describe a way to compute the minimum cost s-t cut in a hypergraph via a reduction to a-b maxflow in a directed graph. Suppose the running time of the directed maxflow routine is $T(m', n')$ on a graph with m' edges and n' vertices, what is the run-time of your algorithm as a function of n, m, p of the given hypergraph?
- A global mincut in a hypergraph $G = (V, E)$ is a the minimum capacity cut that separates some two vertices. Suppose you had an algorithm for a-b-maxflow algorithm in a directed graph that runs in $O(m' + n')\text{polylog}(n)$-time where n' and m' are the number of nodes and edges. Describe a randomized algorithm to compute the global mincut in a hypergraph based on the isolating cut approach. What is the running time of your algorithm in terms of n, m, p. Explain why the isolating cut approach applies and explain the running time carefully.

Problem 2 Spanners. See Prob 2 from CMU's homework `https://www.cs.cmu.edu/~15850/hws/h2.pdf`.

Problem 3 This problem is on sparsest cut and embedding based proofs for flow-cut gap.

- We saw a proof that the integrality gap of the sparsest cut LP is 1 on trees. Argue that the integrality gap of the sparsest cut LP in a graph on n vertices is $O(\log n)$ via the probabilistic tree embedding result that we saw in class.
- Suppose we want to solve the sparsest cut problem in a cycle graph. That is we have $G = (V, E)$ where the edge set forms a cycle and there are non-negative edge capacities $c : E \to \mathbb{R}_+$. The demand graph $H = (V, F)$ and $d : F \to \mathbb{R}_+$ is arbitrary. Describe a simple combinatorial algorithm to find the sparsest cut in a cycle.
- Prove that the integrality gap of the sparsest cut LP is 1 on a cycle graph.

Problem 4 One of the motivations for expanders is their good routing ability. We will explore this aspect in this problem. Given a supply graph $G = (V, E)$ with edge capacities $c : E \to \mathbb{R}_+$ and a demand graph $H = (V, F)$ with demands $d : F \to \mathbb{R}_+$, we say that H is routed in G with congestion γ if there is a multicommodity flow that routes all the demands in F when the edge capacities of G are scaled up by γ. Equivalently, the maximum concurrent flow for H in G is at least $1/\gamma$.

- Suppose G is an expander. Let M be *any* matching on V. Prove that the demand graph (V, M) can be routed in G with congestion $O(\log n)$.
- Suppose G has conductance at least 1. Let H be any demand graph such that $\deg_H(v) \leq \deg_G(v)$. Prove that H can be routed in G with congestion $O(\log n)$.
- Suppose G is an expander. Let M be any matching of cardinality at most $n/\log n$. Prove that $H = (V, M)$ can be routed in G with congestion $O(1)$. ***Hint:*** Use a spanning tree T in G and use it to distribute flow from each terminal (an end point of M) to $\Theta(\log n)$ non-terminals and use uniform multicommodity flow.

Problem 5 Not for submission: This problem is regarding the notion of *element connectivity* which is useful in bridging edge and vertex connectivity. Let $G = (V, E)$ be a graph and let V be partitioned into two sets B and W where B is the set of terminals and W is the set of non-terminals. The elements are $E \cup W$. For any two distinct terminals s, t the element-connectivity between s and t is the maximum number of elment-disjoint paths between s and t. Note that the paths need not be disjoint in terminals. We denote the element connectivity between s and t by $\kappa'_G(s, t)$. An alternative definition is via cuts: $\kappa'_G(s, t)$ is the minimum number of elements whose removal disconnects s from t. See figure. Note that κ' is defined only between the terminals.

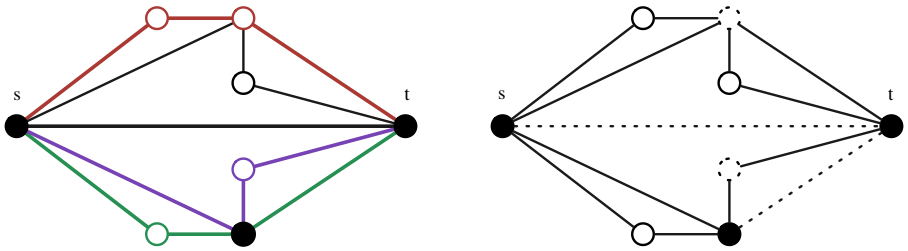

Figure 1: The black vertices are the terminals. The left image shows 4 element-disjoint st-paths. The right image shows removing 4 elements disconnects s and t. $\kappa'(s,t) = 4$.

- Given $G = (V, E)$ and $s, t \in B$ describe an algorithm to compute the $\kappa'_G(s,t)$ via a reduction to maxflow in an associated graph. Note that $\kappa'(s,t) = \kappa'(t,s)$.
- Given three terminals a, b, c prove that $\kappa'(a,b) \geq \min\{\kappa'(a,c), \kappa'(b,c)\}$.
- Let $f : 2^B \to \mathbb{Z}_+$ be a symmetric function where $f(S)$ is defined as the minimum element cut between S and $B - S$ in G. Prove that f is submodular.
- Suppose we have an s-t-maxflow algorithm that runs in $O(m' + n')\text{polylog}(n)$-time in a directed graph with n' nodes and m' edges. Given an instance of element connectivity, let the global mincut of the given instance be defined as the $\min_{x,y \in B, x \neq y} \kappa'_G(x,y)$. Describe an efficient randomized algorithm to compute this global-mincut using the isolating cut approach and the fast maxflow algorithm.

Problem 6 Not for submission: As we discussed in lecture, tree packings are best understood as special cases of matroid base packing. Some nice properties emerge by considering the submodularity of the associated rank function. We do not have time to develop that machinery but we can nevertheless figure out some structure from first principles. Let $G = (V, E)$ be a connected multi-graph. Recall that we defined the upper bound for tree packing via vertex partitions. Suppose P is a partition of V, then the maximum number of edge disjoint spanning trees is at most $\frac{|E_P|}{|P|-1}$. We can view this from a purely edge set point of view which will abstract out the vertices. For any subset $A \subseteq E$ of edges define $\text{rank}(A)$ as $n - \text{comp}(A)$ where $\text{comp}(A)$ is the number of connected components induced by A. Note that $\text{rank}(E) = n - 1$ since G is connected. Note that if k is the maximum number of edge disjoint trees then for any $A \subseteq E$, each spanning tree must contain at least $n - 1 - \text{rank}(A)$ edges from $E - A$ to connect the connected components. Thus if k is the maximum number of edge disjoint spanning trees than $k \leq \frac{|E-A|}{\text{rank}(E)-\text{rank}(A)}$. Thus, $\min_{A \subseteq E} \frac{|E-A|}{\text{rank}(E)-\text{rank}(A)-1}$. For a given A we define $\alpha(A) = \frac{|E-A|}{\text{rank}(E)-\text{rank}(A)}$.

- Argue that if A is an edge set then there is a partition P corresponding to A such that $\alpha(A) = \frac{|E_P|}{|P|-1}$.
- Suppose A, B are two edge sets that $\alpha(A) \leq \lambda$ and $\alpha(B) \leq \lambda$. Then argue that $\alpha(A \cap B) \leq \lambda$. Use this to argue that there is a unique minimal edge set A^* such that $\alpha(A) = \tau_{\text{frac}}(G)$, the fractional tree packing number of G. This corresponds to a partition P^*.

- $\tau_{\text{frac}}(G)$ is also called the network strength of strength of G (to distinguish it from the mincut). For each edge $e \in E - A^*$ we assign its strength λ_e as $\tau_{\text{frac}}(G)$. Consider any component of P^*, and let the corresponding graph be $H = (V_i, E_i)$. Show that $\tau_{\text{frac}}(H) > \tau_{\text{frac}}(G)$. We can find the network strength of H and assign it to all the edges crossing its unique maximal partition P_i^*. By using the implicit recursive procedure, we obtain a strength value for each edge $e \in E$. These strength values turn out to be very useful in graph sparsification.

- Suppose you want to solve the k-cut problem which is defined as follows. Given $G = (V, E)$ and an integer $k > 1$, we wish to partition V into k non-empty parts $V_1, V_2, \ldots, V_k$ such that we minimize the number of edges crossing the parts. Let $P^* = (V_1, V_2, \ldots, V_h)$ be the optimum Tutte-NW partition of G and without loss of generality assume that $|\delta(V_1)| \leq |\delta(V_2)| \leq \ldots \leq |\delta(V_h)|$. Show that if $k = h$ then P^* is an optimum k-cut. Suppose $k < h$. Argue that the partition given by $V_1, V_2, \ldots, V_{k-1}, \cup_{i \geq k} V_i$ is a $2(1 - 1/k)$-approximation for the optimum k-cut.