University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn
M. Ahmadi, P. Schneider

Algorithms Theory
Sample Solution Exercise Sheet 7

Due: Monday, 4th of February, 2019, 14:15 pm

Exercise 1: The Modified Contraction Algorithm *(3+6 Points)*

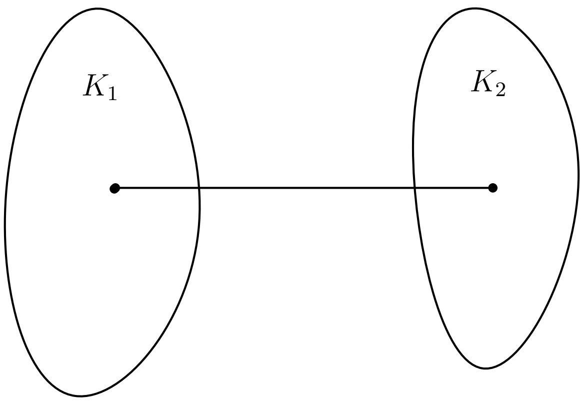
Let us consider the following modified version of the contraction algorithm presented in the lecture. Instead of choosing an edge uniformly at random and merging its endpoints, in each step the modified algorithm chooses a pair of nodes in graph G uniformly at random and merges the two nodes into a single node.

(a) Give an example graph of size at least n where the above algorithm does not work well, that is, where the probability of finding a minimum cut is exponentially small in n.

(b) Prove that property for the example you gave in (a), i.e., show that the modified contraction algorithm has probability of finding a minimum cut at most c^n for some constant $c < 1$.

Sample Solution

(a) We give an unfavorable example where the above modification of the contraction algorithm is doing as bad as claimed. Without restriction and for convenience we assume n is divisible by 4 (otherwise we do the following with n', defined as the smallest multiple of 4 bigger than n). Let G be a graph with n nodes that consists of two cliques K_1, K_2 of size $n/2$ each and has exactly one edge connecting those two cliques (c.f. Figure below).



As soon as we contract a pair of nodes from different cliques, the minimum cut in the resulting multigraph becomes large and we will not be able to find the minimum cut (of size 1). The probability to always choose pairs within one clique becomes very low as we show in part (b).

(b) Recall that the contraction algorithm always produces a multi-graph on "classes" of nodes where cuts are at least as large as the corresponding cuts in the original graph. Obviously the (unique) min-cut is (K_1, K_2), which has size 1. Moreover, all other cuts have size at least $\frac{n}{2} - 1$. We can never obtain the min-cut if we contract a pair of nodes u, v with $u \in K_1$ and $v \in K_2$, since this destroys the small cut between the cliques and any cut after such a contraction has size at least $\frac{n}{2}-1$. In order to obtain the min-cut we need to be lucky enough that we *always* randomly choose a pair u, v that are in the same clique.

We analyze the chance that this happens in the first $T = n/4$ contractions of the modified algorithm. Let n_1, n_2 be the number of remaining nodes in K_1, K_2 after T contractions. Since we contract at most $n/4$ pairs, we have $n_1, n_2 \geq n/4$. The probability to draw a remaining node uniformly at random from a *fixed* clique in any of the first T contractions is therefore at least $\frac{n/4}{n} = 1/4$. Let $\mathcal{E}_i$ be the event that the algorithm randomly chooses *two* nodes u, v from the *same* clique in step $i \leq T$. We have

$$\mathbb{P}(\mathcal{E}_i) = 1 - \mathbb{P}(\overline{\mathcal{E}_i}) = 1 - \mathbb{P}(\text{“}u, v \text{ from different cliques”}) \leq 1 - 2 \cdot \frac{1}{4} \cdot \frac{1}{4} = \frac{7}{8}.$$

Let $\mathcal{E} = \bigcap_{i=1}^{n/4} \mathcal{E}_i$ be the event that we draw pairs of nodes from the same cliques in all of the first $T = n/4$ contractions, which is a necessary condition for finding the min-cut. Since our bounds $\mathbb{P}(\mathcal{E}_i) \leq 7/8$ for the probabilities of the events $\mathcal{E}_i$ are independent from one another, we have

$$\mathbb{P}(\mathcal{E}) = \mathbb{P}\Big(\bigcap_{i=1}^{n/4} \mathcal{E}_i\Big) \leq (7/8)^{n/4} = \big(\sqrt[4]{7/8}\big)^n = c^n, \text{ where } c := \sqrt[4]{7/8} < 1.$$

Exercise 2: Metric TSP with Small Edge Weights *(3+6 Points)*

Consider the family of complete, weighted, undirected graphs $G = (V, E, w)$ in which all edges have weight either 1 or 2.

Remark: TSP is the Traveling Salesperson Problem. The goal is to find a tour, i.e., a permutation $v_1, \ldots, v_n$ of nodes, that minimizes the total weight of edges on that tour $w(v_1, v_n) + \sum_{1=1}^{n-1} w(v_i, v_{i+1})$.

(a) Assume you have a subroutine that computes a *minimum* 2-matching for the above family of graphs in polynomial time. Describe an *efficient* algorithm that computes a 4/3-approximation for the TSP problem for graphs of this family.

Remark: A 2-matching is a subset $M \subseteq E$, so that every $v \in V$ is incident to exactly 2 edges in M.

(b) Prove that your algorithm computes a 4/3-approximation of TSP on the this family of graphs.

Sample Solution

(a) Our algorithm basically consists of two steps. First we run the given subroutine which gives us a division of G into cycles (each node is part of exactly one cycle). Then we join these cycles by removing one edge from each cycle and then carefully join the loose endpoints with additional edges such that we obtain a single cycle (i.e. a tour). (While we consider the above sufficient) we formalize this with the following (efficient) algorithm:

Algorithm 1 `TSP`

run the subroutine to compute a 2-matching M
$C_1, \ldots, C_\ell \leftarrow$ set of cycles on G induced by M ▷ *the C_i consist of edges*
$T \leftarrow \bigcup_{i=1}^{\ell} C_i$
$v_1 \leftarrow$ arbitrary node on C_1
for $i = 1, \ldots, \ell - 1$ **do**
  Select an edge $\{u_i, v_i\} \in C_i$ incident to v_i ▷ *v_i was selected in the previous step*
  $v_{i+1} \leftarrow$ arbitrary node on C_{i+1}
  $T \leftarrow (T \setminus \{u_i, v_i\}) \cup \{u_i, v_{i+1}\}$ ▷ *connect two cycles C_i, C_{i+1} in T*
Select an edge $\{u_\ell, v_\ell\} \in C_\ell$ incident to v_ℓ
$T \leftarrow (T \setminus \{u_\ell, v_\ell\}) \cup \{u_\ell, v_1\}$ ▷ *connect last cycle with the first cycle to obtain a tour*
return T ▷ *or a permutation of nodes on T as they occur on the cycle T*

(b) Since on a minimal TSP tour T_{min} every node has exactly two incident edges, the edges of T_{min} are also a 2-matching. Therefore the total edge weight of the edges on the tour T_{min} (let us abbreviate this with $w(T_{min})$) is at least that of a minimal 2-matching M, i.e., $w(T_{min}) \geq w(M)$. Obviously $w(T_{min}) \geq n$ since each edge has weight at least 1.

The min. 2-matching M disassembles G into at most $n/3$ cycles (the smallest possible cycle with the required property is a triangle). In our algorithm we remove an edge from each cycle and add another edge instead. In the worst case we always remove an edge with weight 1 and add one with weight 2 instead. Therefore we add at most $n/3$ weight to $w(M)$ when we construct T. Thus

$$w(T) \leq w(M) + \frac{n}{3} \leq w(T_{min}) + \frac{w(T_{min})}{3} = \frac{4w(T_{min})}{3}.$$

Exercise 3: Covering Paths *(4+4+4 Points)*

Consider an undirected, unweighted Graph $G = (V, E)$ with n nodes. We are given a set $\mathcal{P}$ of p *simple* paths in G, where each path has exactly ℓ nodes. We say a path $P \in \mathcal{P}$ is *covered* by a set Q of nodes if P has a node in Q. Then, the goal is to find a set $Q \subseteq V$ of nodes with minimum cardinality such that *every* path in $\mathcal{P}$ is covered by Q.

Now consider the following simple greedy algorithm: It starts with $Q = \emptyset$. As long as there is a path not covered by Q, the node that covers the most uncovered paths is added to Q.

(a) Argue why the above greedy algorithm provides a $(1+\ln p)$-approximation of an minimal solution.

(b) Show that after selecting i nodes, there are at most $p(1-\frac{\ell}{n})^i$ uncovered paths left.

(c) Show that for $i > \frac{n}{\ell} \ln p$ all paths are covered. *Hint:* $1-x < e^{-x}$ for all $x > 0$.

Sample Solution

(a) We can reduce the problem at hand to the minimum set cover problem given in the lecture. Let the set of paths $\mathcal{P}$ be the base set, and let $\mathcal{S} = \{S_1, \ldots, S_n\}$ where S_j contains all paths that node j covers. In the lecture we showed that the greedy algorithm has an approximation ratio of $1+\ln s$ where s is the maximum cardinality of any S_j. Since $s \leq p$ the claim follows.

(b) Let $i = 1$, i.e., we selected the first path. The sum of all nodes of all paths is $p\ell$ if we count each node as many times as it appears on a path. Since we have only n nodes in total, on average a node covers $p\ell/n$ paths. There must be one node that covers at least as many paths as the average, thus there is a node that covers at least $p\ell/n$ paths. Since the greedy algorithm selects the node that covers most paths, we "loose" at least $p\ell/n$ many paths in that step, hence we have

$$p - \frac{p\ell}{n} = p\Big(1-\frac{\ell}{n}\Big)$$

many paths left. The rest can be done via induction. Presume that in the i-th iteration we have $p' := p\big(1-\frac{\ell}{n}\big)^i$ paths left. With the same argument as before, after we select the $(i+1)$-th node, we have at most

$$p'\Big(1-\frac{\ell}{n}\Big) \overset{\text{hypothesis}}{=} p\Big(1-\frac{\ell}{n}\Big)^i\Big(1-\frac{\ell}{n}\Big) = p\Big(1-\frac{\ell}{n}\Big)^{i+1}$$

many uncovered paths left.

(c) We show that $i > \frac{n}{\ell} \ln p$ implies $p(1-\frac{\ell}{n})^i < 1$ which means that all paths are covered by greedily selecting i nodes.

$$\begin{aligned} & i > \tfrac{n}{\ell} \ln p \\ \iff\quad & -\tfrac{i\ell}{n} < -\ln p \\ \iff\quad & e^{-\frac{i\ell}{n}} < \tfrac{1}{p} \\ \iff\quad & \big(e^{-\frac{\ell}{n}}\big)^i < \tfrac{1}{p} \\ \overset{\text{Hint}}{\Longrightarrow}\quad & \big(1 - \tfrac{\ell}{n}\big)^i < \tfrac{1}{p} \\ \iff\quad & p\big(1 - \tfrac{\ell}{n}\big)^i < 1 \end{aligned}$$

Exercise 4: Resilience to Edge Failures *(10 Points)*

In the lecture, we showed that every (undirected) graph with edge connectivity λ has at most $n^{2\alpha}$ cuts of size at most $\alpha \cdot \lambda$. Use this fact to prove the following statement:

Let $G = (V, E)$ be an undirected graph with constant edge connectivity $\lambda \geq 1$ and let $p := \min\big\{1, \frac{c \ln n}{\lambda}\big\}$, where $c > 0$ is a constant. Assume that edge $e \in E$ is sampled independently with probability p. Let E_p be the set of sampled edges and let $G_p = (V, E_p)$ be the graph induced by the sampled edges. Show that if the constant c is chosen sufficiently large, the graph G_p is connected with high probability.

Hint: A graph is connected if and only if there is an edge across each of the $2^{n-1} - 2$ possible cuts. Analyze the probability that for a cut of a given size k in G, at least one edge is sampled in E_p. Then, use the upper bound from the lecture on the number of cuts of a given size and a union bound over all cuts of a given size in G. Finally, one can do a union bound over all possible cut sizes.

Sample Solution

We assume $p < 1$, otherwise $G_p = G$ is obviously connected. the graph G_p becomes disconnected if all edges of any cut are removed. The probability that we remove all edges of a cut of size $k := \alpha\lambda$ is $(1-p)^k$. We obtain

$$(1-p)^k = \Big(1 - \frac{c \ln n}{\lambda}\Big)^k \overset{(*)}{<} \exp\Big(-\frac{kc\ln n}{\lambda}\Big) = \exp\Big(-\alpha c \ln n\Big) = n^{-c\alpha} \qquad (*) : 1 - x < e^{-x}$$

(3 Points)

In the lecture we saw that we have at most $n^{2\alpha}$ cuts of size $k = \alpha\lambda$. Let us enumerate these cuts with $C_1, \ldots, C_\ell$ with $\ell \leq n^{2\alpha}$. Let $\mathcal{E}_i^k$ be the event that all edges of cut C_i are removed. Let $\mathcal{E}^k := \bigcup_{i=1}^{\ell} \mathcal{E}_i^k$ be the event that all edges of at least one of the cuts C_i out of the cuts $C_1, \ldots, C_\ell$ of size k are removed. We get

$$\mathbb{P}(\mathcal{E}^k) = \mathbb{P}\Big(\bigcup_{i=1}^{\ell} \mathcal{E}_i^k\Big) \leq \sum_{i=1}^{\ell} \mathbb{P}(\mathcal{E}_i^k) \leq \ell n^{-c\alpha} \leq n^{2\alpha} \cdot n^{-c\alpha} = n^{-(c-2)\alpha} = n^{-(c-2)k/\lambda}$$

(4 Points)

Finally we bound the probability that one cut of *any* cut of *any* size k gets all its edges removed. Clearly we have $k \leq n$. Let $\mathcal{E} := \bigcup_{k=1}^{n} \mathcal{E}^k$. We obtain

$$\mathbb{P}(\mathcal{E}) = \mathbb{P}\Big(\bigcup_{k=1}^{n} \mathcal{E}^k\Big) \leq \sum_{k=1}^{n} \mathbb{P}(\mathcal{E}^k) \leq \sum_{k=1}^{n} n^{-(c-2)k/\lambda} \leq \sum_{k=1}^{n} n^{-(c-2)/\lambda} = n \cdot n^{-(c-2)/\lambda} = n^{-(c-2-\lambda)/\lambda}.$$

(3 Points)

If we desire that $\mathcal{E}$ occurs with probability at most $n^{-c'}$ for some constant $c' > 0$ we have to choose c such that $(c - 2 - \lambda)/\lambda > c'$. This is the case for $c > c'\lambda + \lambda + 2$ which is constant in n.