

Exam 26 October 2012, 8:00–13:00, Sparta:A–B

EDAN55 Advanced Algorithms

Comments

1a For instance, $A = \{1,3,4,6\}$, $B = \{2\}$, $C = \{5\}$ of cutsize 6.
(There are other correct answers.)

1b $A = \{1,4,5\}$, $B = \{2,6\}$, $C = \{6\}$. This is the unique correct answer.

1c The simplest answer seems to be the 3-star,

$$G = (\{1,2,3,4\}, \{\{1,4\}, \{2,4\}, \{3,4\}\})$$


with optimal solution $A = \{1\}$, $B = \{2,3\}$, $C = \{4\}$ (cutsize 3) and algorithm's solution $A = \{1,4\}$, $B = \{2\}$, $C = \{3\}$ (cutsize 2). There are many other ways of solving this. Some students managed to find only instances with approximation factor $\frac{7}{9}$ or other nonoptimal answers, these were graded with $\frac{3}{4}$ points.

1d There are many, many ways of doing this correct. Here's one of the neater: "Consider an edge $e = v_j v_k$ not in the cut. We will associate two unique edges that *are* in the cut with e . This shows that at least $\frac{2}{3}|E| \geq \frac{2}{3}\text{OPT}$ edges are in the cut. The association is given as follows. Assume without loss of generality that $k > j$. When v_k was added to one of the three parts, there must have been edges $v_{j'} v_k$ and $v_{j''} v_k$ to each of the two other parts ($j' < k$ and $j'' < k$). (There may have been many such edges, for definiteness pick the ones with j' and j'' minimal. Note that the two edges are never associated with another edge than $v_j v_k$ because the association only happens via the highest-numbered endpoint v_k .) Both $v_{j'} v_k$ and $v_{j''} v_k$ have endpoints in different parts, so they will each contribute to the cut." (This was super-formal. Most student answers will miss small points such as uniqueness, or do the argument only for $v_j v_k \in A$; full marks were still obtained by those.)

What doesn't work is to begin by saying "The worst case is when ..."—this is not how worst-case arguments work. The melody of a worst-case argument is "Consider an arbitrary case. We have ..." or something similar: they make *no* assumption on structural properties of the input.

1e The argument I had in mind is the following: "Set $\epsilon = \frac{1}{2m}$ and run the hypothetical approximation algorithm on an instance of Max Tripartition. By assumption, the algorithm find a partition of

size at most $\frac{1}{2m}$ below the optimum. Since the optimum is at most $|E| = m$, and the solution is integral, the approximation algorithm solves the decision problem, and by assumption runs in polynomial time." For this argument, it's essential that the student tells me what ϵ is; answers of the form "by making ϵ sufficiently small" or similarly imprecise statements received 1 point. A popular alternative argument, was to reduce from 3-colouring, which is known to be hard to approximate. Not what I had in mind, but perfectly correct.

- 1f "Initially, set $A_1 = \{v_1\}, \dots, A_k = \{v_k\}$. Then, for every i with $k < i \leq n$, greedily add v_i to the set A_i to which it has the fewest edges, breaking ties arbitrarily. The resulting approximation factor is $(k-1)/k$." Most students give a full argument for why the approximation factor is what it is. A popular wrong answer gives the approximation factor $\frac{2}{k}$. Getting the algorithm right but the analysis wrong earns $\frac{1}{2}$ points.
- 1g "Initially, set $A = \{v_1\}, B = \{v_2\}$. Then, for every i with $1 < i \leq n$, if there are more edges from A to v_i than from v_i to B , add v_i to A (otherwise to B). The resulting approximation factor is $\frac{1}{4}$." (There are 4 quantities in play: edges from A to v_i , from v_i to A , from B to v_i , and from v_i to B . The algorithm makes the best choice, so it throws away at most 3 edges for each edge added.) A popular wrong answer gives the approximation factor $\frac{1}{2}$.
- 2b "Construct the set L of all lines between 2 points of S ; L as size n^2 . For every subset of k lines from L , check if they cover all points. There are $\binom{|L|}{k} = \binom{n^2}{k}$ such subsets." There are many other ways of doing this.
- 2c "If the $k+1$ colinear points are not covered by a single line, they must each lie on their own line. This requires $k+1 > k$ lines."
- 2d  , but there are many other examples.
- 2e First check if there are $k+1$ colinear points in S . (This can be done by considering every of the n^2 lines l between 2 points in S and checking if l contains $k+1$ points. The time for this part is $O(n^3)$. If such $k+1$ points exist, cover them with a single line (this is correct by 2c) and remove them. So assume now without loss of generality that S contains at most k colinear points. If S has more than k^2 points, answer "no". (Every of the k lines could cover at most k points each.) Otherwise solve the problem by exhaustive

search. The running time for the last step depends on $|S|$, which depends only on k .

3b Check all subsets of C . Running time $O^*(2^n)$.

3c View U as the vertex set of a graph and C has the edge set. Then (U, C) contains a perfect matching of size r if and only if the U can be covered using $n - r$ sets (namely, the r 2-sets from the perfect matching, each disjointly covering 2 points, and $n - 2r$ other sets, one per point). Perfect matching is known to be in P.

3d By 3c, we can assume that C contains at least one set S with $|S| \geq 3$. Branch on whether S belongs to the solution or not. In the first case, at least 3 elements are removed from U , and one from C . In the second case, one element is removed from C . The running time in terms of $n + m$ is $T(n + m) = T(n + m - 1) + T(n + m - 4)$, with solution $O^*(1.39^{n+m})$. Initially, we have $n + m = 2n$, so the total running time grows as $1.39^{2n} < 1.41^{2n} < \sqrt{2}^{2n} = 2^n$.

4a $S = \{1, 2, 3, 5, 6\}$.

4b " $\Pr(v \in S) = 1 - 2^{-d(v)}$: the probability that a neighbour of v has lower priority is $\frac{1}{2}$. By independence, all neighbours of v have lower priority with probability $(\frac{1}{2})^{d(v)}$, which is the complementary event $v \notin S$." Note the phrase "by independence". Popular wrong answer $(\frac{1}{2})^{d(v)}$ receives $\frac{3}{4}$ points. Several students have difficulties assessing $\Pr(p(u) < p(v))$ (which is $\frac{1}{2}$). Other popular wrong answers are $1 - 1/(d(v) + 1)$ and $d(v)/(d(v) + 1)$ and the puzzling $1 - p(v)^{d(v)}$ and $\sum_i p(v_i)d(v_i)$ (which is not even a probability).

4c "No. If $u \in S$ then $v \notin S$." There are many ways of arguing for this, including very proper ones like " $\Pr(u \in S \wedge v \in S) = 0$, but $\Pr(u \in S) = \frac{1}{2}$ and $\Pr(v \in S) = \frac{1}{2}$." This is a good place to remind students the *read the bloody question*, several students start with "Yes." (which is wrong) and then proceed to give a perfectly correct answer for *dependence* (which is right). I have been very charitable in re-interpreting these answers, but it requires that a *careful argument is included*. Also, many students fail to say either "yes" or "no", again leaving me to guess what the answer is supposed to be (and trust me, not all arguments are clear enough to determine that.)

4d "1." A different way to ask the question would have been "Prove that the algorithm always finds a vertex cover."

4e "Introduce the indicator random variable X_i , with $X_i = 1$ if $v_i \in S$ and 0 otherwise. From 4b we have $E[X_i] = \Pr(X_i = 1) = 1 - \frac{1}{2}^{d(v)}$."

Then, by linearity of expectation, $E[|S|] = E[\sum X_i] = \sum E[X_i] = n(1 - \frac{1}{2}^d(v))$. Not solving 4b (or inheriting a wrong answer) loses you no points. You do lose points for not including an argument, at the very least write “by linearity of expectation” or introduce the indicator random variables. Failure to do so, loses you $\frac{1}{2}$ to 1 points. Some students compute the expectation directly from the definition (instead of using the indicator random variable trick and linearity); this earns full points and my respect, but is an immensely complicated way of solving this exercise.

4f “The optimum solution has size at most $\frac{1}{2}n$, by taking every other node. From 4b, the algorithm finds a solution of size at least $\frac{3}{4}n$. The resulting approximation factor is $\frac{3}{2}$.” There is some confusion about what to divide by what (including in the literature), I’ve given full marks to any interpretation that made sense, as long as the upper bound on OPT and the lower bound on the algorithm were correct.

4g “The minimum vertex cover is given by $S = \{1\}$. The algorithm finds this S if and only if $p(1) < p(i)$ for all $i = 2, \dots, n$. This happens with probability $\frac{1}{n}$.” If you want, you can expand on the reason for the probability (“In any choice of priorities, exactly 1 vertex has the minimum priority. By a symmetry argument, every vertex is equally likely.” A *very* popular wrong answer is to observe that $\Pr(p(1) < p(i)) = \frac{1}{2}$ (which is correct) and then wrongly continue “so $p(1)$ is smaller than $p(i)$ for all $i = 2, \dots, n$ with probability $\frac{1}{2}^{n-1}$.” This argument assumes independence of the events $p(1) < p(i)$ (otherwise you couldn’t multiply their probabilities), but these events are *not* independent.

4h “The success probability is $q = \frac{1}{n}$. Thus, the probability that t independent repetitions all fail is $(1 - q)^t$. For $t = q^{-1} = n$ this value is at most $\frac{1}{e}$ according to 13.1.”