Approximability & Inapproximability

Though minimization and maximization are the same (by taking inverse ratio) in essense, however we have different levels of (In)approximability for natural maximization and minimization problems.

Approximation Algorithms in General:

→ A popular way to combat NP-hardness

→ Consider any problem $X \in NP$ and the goal of Max or Min

We say an Algorithm ALG is c-approximation if (Assuming Alg has poly-time)

if goal is $\min \frac{cost(Alg(x))}{cost(Opt(x))} \leq c$

if goal is $\max \frac{cost(opt(x))}{cost(Alg(x))} \leq c$

Approximability of Minimization problems

(APX-hard)

- $\Theta(1)$ approximation: Steiner tree, Steiner forest, TSP, vertex cover, Facility location, etc.

e.g. in Steiner tree: An undirected graph G, a weight w(e) associated with each edge e and a set of terminals are given and we want to find a subgraph H with minimum weight that connects these terminals (or equivalently everyone to the root).

In Steiner Forest: different terminals can have different roots.

- $\Theta(\log^* n)$ approximation (define iterated $\log^{(i)} n$ as follows: $\log^{(1)} n = \log_2 n$ and $\log_2^{(i+1)} n = \log_2(\log_2^{(i)} n)$; then $\log^*(n)$ is defined to be the least integer i for which $\log^{(i)} n \leq 1$)

An example is Asymmetric k-Center: given an integer k and a complete digraph over n points together with a distance function obeying the directed triangle inequality, the goal is to chose a set of k points to serve as centers and to assign all points to the centers such that maximum distance of any point to its center is as small as possible [the problem is hard unless $NP \subseteq DTIME(n^{\log\log n})$]

- $\Theta(\log n)$-approximation: Set cover (Dominating set), node-weighted steiner tree (2) Lots of others via FRT & Racke

Set cover: give a collection of subsets of a universe, find the minimum # of subsets which cover all the universe.

Node-weighted steiner tree: is the same as edge-weighted except we have weights on the nodes.

- $\Theta(\log^2 n)$-approximation: Group Steiner tree: Same as steiner tree except we are given several groups of terminals (possibly intersecting) and we want to connect one (terminal) from each group [the problem is hard unless NP $\subseteq$ ZTIME(n^{polylog}) probabilistic algorithms]

- $O(n^{\varepsilon})$ for any constant ε, $\Omega(\log^2 n)$: directed steiner tree: Same as steiner tree except the graph is directed

- $O(n^{\frac{1}{3}})$, $\Omega(2^{\log^{1-\varepsilon} n})$: Label-cover (Min Rep) [to be defined later]

Directed steiner Forest: Same as steiner Forest except the graph is directed

- $\tilde{O}(n)$ $\Omega(n^{1-\varepsilon})$: Coloring: Given an undirected graph G, find the minimum # of colors for vertices such that every two adjacent vertices have different colors.

Approximability of Maximization Problems

$\Theta(1)$-approximation: maximum coverage, max-cut (max matching is already polynomial)

Max. coverage: give several sets where they can share elements and a number K, find at most K of these sets such that the maximum number of elements are covered, i.e., the union of selected sets has maximum size

max-cut: Partition the graph (vertex set) into A & B with maximum # edges between A & B.

$\Theta(\log n)$-approximation: Unique-coverage, domatic number

Unique coverage: given an input the same as set cover find a sub collection to maximize the number of elements uniquely covered, appear in exactly one set. [DFHS'06] [novel hardness result; only $O(\log^{\varepsilon} n)$-hardness under ETH]

Domatic number: maximum k such that graph $G=(V,E)$ has a partition of V into disjoint sets $V_1, V_2, \ldots, V_k$ such that each V_i is a dominating set for G. (not hard to see $k \geq 3$) [FHKS'02]

$O(n^{\frac{1}{3}}), \Omega(2^{\log^{1-\varepsilon} n})$: Label cover (Max Rep) [to be defined]

$\tilde{O}(n)$-$\Omega(n^{1-\varepsilon})$: Independent set (clique)

Independent set: find a maximum independent set S, i.e., a set of vertices such that there is no edge in between.

Note that above hardness often happens for general graphs; For special graphs such as planar graphs (which can be drawn on the plane with no crossing) we do not have so much hardness e.g. for coloring we have $\frac{4}{3}$-hardness, and Prize-Collecting Steiner Forest [see later] is APX-hard: lots of other problems have PTAS, i.e., $1+\varepsilon$ approximation for any constant ε [APX-hard means NO PTAS] (a.k.a Max-SNP-hard)

(see notes for Fixed-Parameter hardness)

Two important problems for hardness of approx. Alg.

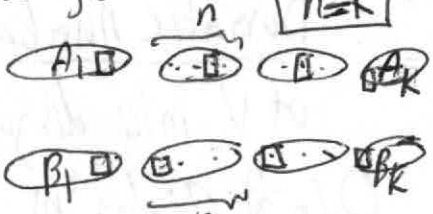
Label Cover: Min Rep and Max Rep

Max-Rep: Given a bipartite graph $G=(A,B,E)$, where $|A|=|B|=N$ and equitable partitions A & B into K-sets of same size $n=\frac{N}{K}$

Namely $A_1 \dots A_k$ and $B_1 \dots B_k$

Goal: choose a subset A' of A and a subset of B' of B such that $|A' \cap A_i| = 1$ and $|B' \cap B_i| = 1$ (which represents A_i & B_i the intersection) and the graph induced on $A' \cup B'$ has maximum number of edges.

sometime $n = k$



□: Representative in Max-Rep

Min-Rep: Input the same as before

Goal: choose a subset A' of A and a subset of B' of B (maybe more than one element from A_i or B_j) such that All super-edges are covered while minimizing $|A'| + |B'|$

important → (super-edges will be represented by functions as well)

We say there is a super-edge (i,j) if there is an edge in $G[A_i \cup B_j]$.

We say a super-edge (i,j) is covered if $a \in A' \cap A_i$, $b \in B' \cap B_j$ & $(a,b) \in E(G)$

→ Sometime it is convenient (and possible) to restrict the MIN REP (or MAX-REP) so that for every super-edge (A_i, B_j), i.e., $G[A_i \cup B_j]$, each vertex in B_j is adjacent to at most one vertex in A_i (star property since $G[A_i \cup B_j]$ is a collection of vertex-disjoint stars). This has applications e.g. in proof of hardness for survivable network design [Kortsarz, Krauthgamer, Lee '03]

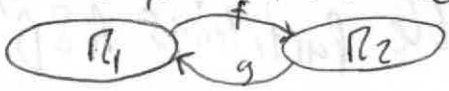
A unique Game instance is an instance in which $G[A_i \cup B_j]$ is further a matching

Approximation-preserving Reduction from A to B (APR-Reduction)

In general it means if we can well approximate B then we can well-approximate A as well. Indeed there are more than eight notions of approximation preserving reductions differing only in some details. Let's see a practical one here.

Def: Let Π_1 & Π_2 be minimization problems. An approximation (factor) preserving reduction from Π_1 to Π_2 consists of two poly-time computable functions f, g such that:

(I) for any instance I_1 of Π_1, $I_2 = f(I_1)$ is an instance of Π_2 such that $opt_{\Pi_2}(I_2) \le opt_{\Pi_1}(I_1)$.



(II) For any feasible solution s_2 of I_2, $s_1 = g(I_1, s_2)$ (g maps s_2 into an instance I_1)
$cost_{\Pi_1}(I_1, s_1) \leq cost_{\Pi_2}(I_2, s_2)$.

Intuitively: $sol_{\Pi_2} \geq sol_{\Pi_1} \geq opt_{\Pi_1} \geq opt_{\Pi_2}$

→ For maximization problems $\leq$ becomes $\geq$. often $\leq$ becomes $=$, esp. for (I) condition.

→ since Π_1 in some sense is sandwiched between Π_2 any approximation factor for Π_2 is an approximation factor for Π_1 as well.

→ more generally if $opt_{\Pi_2}(I_2) \leq \alpha\, opt_{\Pi_1}(I_1)$ and $cost_{\Pi_1}(I_1, s_1) \leq \beta\, cost_{\Pi_2}(I_2, s_2)$ we will have $\alpha\beta$-approximation preserving reduction.

Though as we see via approximation preserving reductions we can get several hardness reductions, another more useful reductions are Gap preserving Reduction especially because of PCP (Probabilistically checkable proof) Theorem.

Def: Let P and P' be maximization problems. A gap preserving reduction from P to P' is a polynomial-time algorithm which given an instance I of P with $|I| = n$ produces an instance I' of P' with $|I'| = n'$ such that if

(I) If $OPT(I) \geq h(n)$ then $OPT(I') \geq h'(n')$

(II) If $OPT(I) \leq \frac{1}{g(n)} h(n)$, then $OPT(I') \leq \frac{1}{g'(n')} h'(n')$

for some functions $h(n), g(n), h'(n'), g'(n')$ with $g(n), g'(n') \geq 1$.

Observe that if Gap-$P_{g(n)}$ is hard (e.g. NP-hard) and thus approximating P within factor $g(n)$ is hard, then Gap-$P_{g'(n')}$ is also hard (and thus approximating P within factor $g'(n')$ is hard).

For minimization problems we replace (I) and (II) with ($\tilde{I}$) and ($\tilde{II}$)

(I) If $OPT(I) \leq h(n)$ then $OPT(I') \leq h'(n')$

(II) If $OPT(I) \geq g(n)\, h(n)$ then $OPT(I') \geq g'(n')\, h'(n')$.

Now before seeing some examples, let's see PCP theorem and unique game conjecture.

PCP Theorem (Raz '98): For every $\varepsilon > 0$, there is a MaxRep instance with $n \leq \frac{1}{\varepsilon^{O(1)}}$ (remember $N = nk$) for which it is NP-hard to distinguish two cases below:

Case 1: There is a solution which covers ALL super edges.

Case 2: In every solution we can cover at most ε fraction of super edges.

Unique Game Conjecture (Khot '02): For every $0 < \varepsilon < \frac{1}{2}$, there is a Unique-Game instance for which it is NP-hard to distinguish two cases below:

Case 1: There is a solution which covers at least $1-\varepsilon$ fraction of super edges.

Case 2: In every solution we can cover at most ε fraction of super edges.

Note that if UGC is correct (though there is some doubt about it at least in this very strong form), then every gap-preserving reduction/approximation preserving reduction using that gives us inapproximability results.

Often we do inapproximability as follows:

UGC, PCP $\xrightarrow{\text{Gap-Preserving reductions}}$ Some problems $\xrightarrow{\text{Approximation preserving reductions}}$ other problems

Note that both types of reductions above are transitive.

A fundamental and quite difficult result which is equivalent to PCP thm and in fact derived from it is as follows:

Thm: It is NP-hard to distinguish the following two cases for Max-3SAT (in which we want to maximize # satisfied clauses) for an absolute constant ε_0 (i.e., there exists such an ε_0):

Case 1: The given input instance is satisfiable

Case 2: At most $(1-\varepsilon_0)$ fraction of clauses are satisfiable in every assignment.

Thm above say Max-3-SAT is APX-hard

Using above we can get $\frac{7}{8}$-inapproximability result for $_{MAX}$3-SAT (unless P=NP, but then the gap is NOT between all satisfiability and $\frac{7}{8}$ satisfiability)

Similarly we get below useful consequences from PCP.

Thm: Unless $NP \subseteq DTIME\ O(n^{Polylog(n)})$, it is hard for

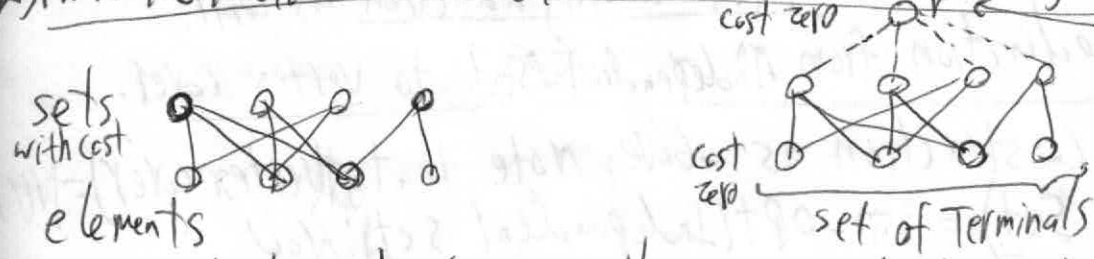
A) Max-Rep to distinguish following two cases:
Case 1: we cover ALL super-edges.
Case 2: we can cover at most $\frac{1}{2^{\log^{1-\varepsilon} n}}$ fraction of super-edges.

B) For Min-Rep: Case 1: There is a solution of $2k$ (i.e., one representative from each A_i and each B_j).
Case 2: To cover all super-edges we need at least $2k\,2^{\log^{1-\varepsilon} n}$ elements.

Note that by the above theorem both MIN-REP and MAX-REP cannot be approximated within ration $2^{\log^{1-\varepsilon} n}$, for any fixed $\varepsilon > 0$, unless $NP \subseteq DTIME(n^{Polylog(n)})$.

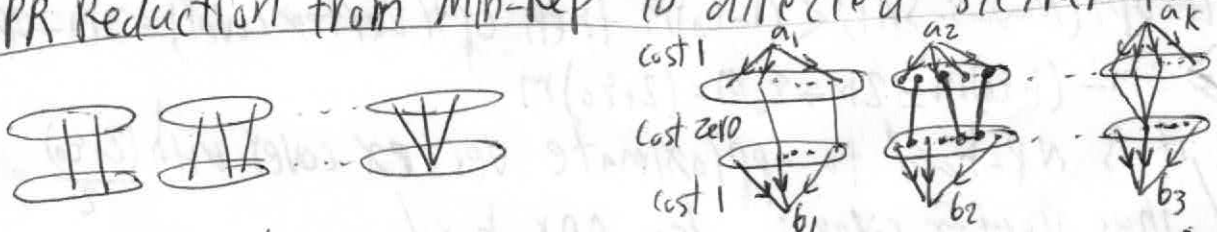
Now let's see some examples (via aforementioned reductions):

☆) APR Reduction from set cover to Node-weighted Steiner tree:



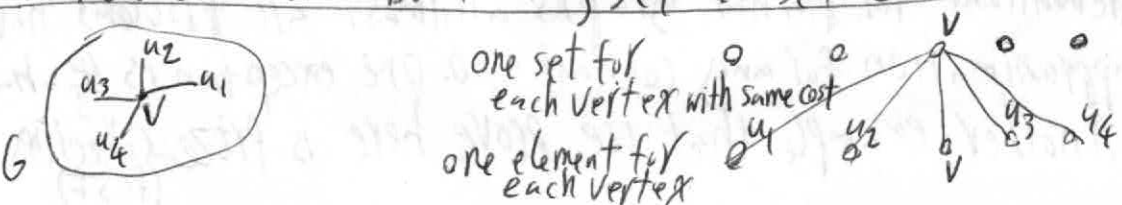
Note that opt remains the same in both instances.

(☆) APR Reduction from Min-Rep to directed steiner forest

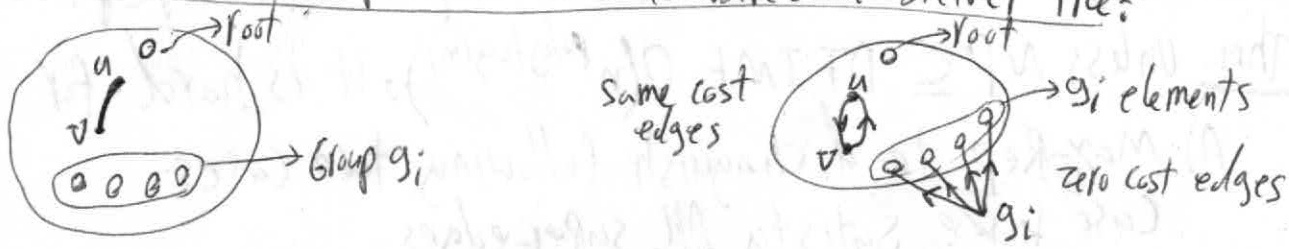


Again opt has the same cost in both instances. connecting (a_i, b_j) is required if (A_i, B_j) is a super-edge

(☆) APR Reduction from Dominating set to set cover:



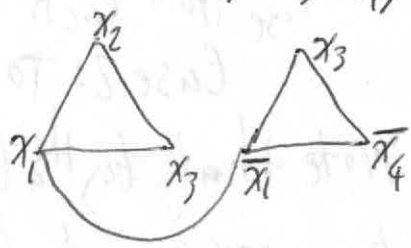
*) APR Reduction from Group steiner Tree to Directed steiner Tree:



(*) APR Reduction from Max-3-SAT to independent set: $\emptyset = (x_1 \vee x_2 \vee x_3) \wedge (\bar{x}_1 \vee x_3 \vee \bar{x}_4)$

let $\emptyset$ has clauses $C_1, \dots C_m$ and n variables $x_1, \dots, x_n$

graph $G_\emptyset$ has 3 nodes per clause and thus $3m$ nodes (corresponding to 3 literals)



For each clause, we put all edges between its nodes

we add edges between x_i & $\bar{x}_i$ if x_i and $\bar{x}_i$ belong to two different clauses.

It is easy to check opt(Max-3-SAT) = opt(independent set)

Also since according to PCP, Max-3-SAT is APX-hard, so is independent set.

Also it implies there is a gap between opt = m and opt $< (1-\varepsilon_0)m$ for independent set. and thus it is a gap-preserving reduction as well.

* Gap-Preserving Reduction from Max-3-SAT to vertex cover.

we use the same construction as above: Note that opt(vertex cover) = $|V(G)|$ - opt(Independent Set) = $3m$ - opt(Independent set). Now

Case1 if opt(Max-3-SAT) = m then opt(vertex cover) = $3m - m = 2m$

Case2 if opt(Max-3-SAT) $< (1-\varepsilon_0)m$ then opt(vertex cover) = $3m$ - opt(Max-3-SAT)

$> 3m - (1-\varepsilon_0)m = 2m + \varepsilon_0 m = (2+\varepsilon_0)m$

Thus, it is NP-hard to approximate vertex cover with $\frac{(2+\varepsilon_0)}{2} = (1+\frac{\varepsilon_0}{2})$ and thus vertex cover is also APX-hard.

Now let's see a more complicated example of Gap-Preserving Reduction. As we aforementioned for planar graphs, almost all problems have PTAS, i.e., $1+\varepsilon$ approximation for any constant $\varepsilon > 0$. One exception is $\frac{4}{3}$-hardness for Coloring. Another example that we prove here is Prize-Collecting Steiner forest (PCSF)

In PCSF, an instance of steiner forest is given and in addition we are allowed to not connect pair (s_i, t_i) and instead pay penalty π_i (in addition of the cost of forest.)

(X) There is a gap-preserving reduction from vertex cover on 3-regular graphs to PCSF on series-parallel graphs (which are special planar graphs).
[Bateni, H., Marx, D.]

Pf: First note that there is a gap-preserving reduction from vertex cover to vertex cover on 3-regular graphs. since a 3-regular graph with n vertex has a vertex-cover of size at least $\frac{n}{3}$, thus there is a hard-instance of this problem for which we cannot distinguish two cases below unless P=NP:

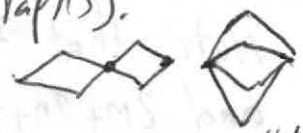


Fig. Series-parallel graphs can be obtained by two operations above.

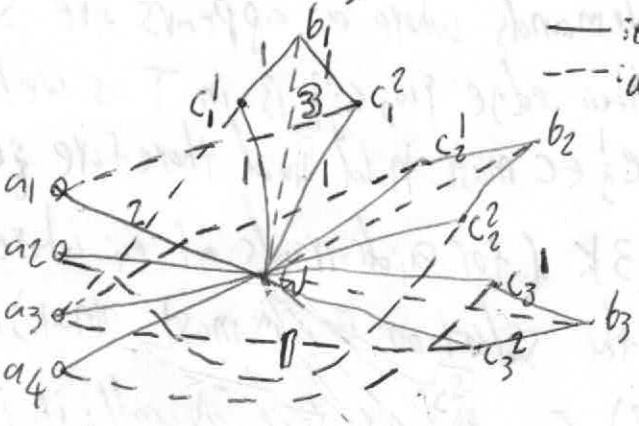
Case 1: The graph has a vertex cover of size cn, for some constant $c \geq 0$

Case 2: Every vertex cover of the graph has size greater than $c(1+\varepsilon_0)n$, for some constant $\varepsilon_0 > 0$.

Now consider this hard instance graph G. Let $m = \frac{3n}{2}$ be the number of edges of G, v_i be the ith vertex, e_j be the jth vertex with endpoints e_j^1 & e_j^2. Now we construct a graph H and a set D of demand-pairs (with penalties) as follows:

The vertices of H consists of

1) a_i for $1 \leq i \leq n$
2) b_j, c_j^1, c_j^2 for $1 \leq j \leq m$
3) central vertex w



The edges are as follows:

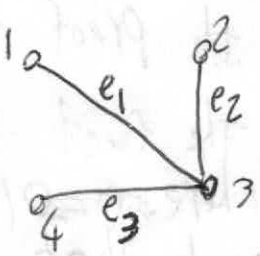
I) $\{w, a_i\}$ of cost 2 for $1 \leq i \leq n$

II) $\{w, c_j^1\}, \{w, c_j^2\}, \{c_j^1, b_j\}, \{c_j^2, b_j\}$ of cost 1 ... for $1 \leq j \leq m$

Finally the demand-pair set D contains.

I) $\{w, b_j\}$ with penalty 3 for $1 \leq j \leq m$

II) if $v_i = e_j^l$ for $1 \leq i \leq n, 1 \leq j \leq m, l \in \{1, 2\}$ then $\{a_i, c_j^l\}$ is a demand with penalty 1



Next we prove that

(1) Given a vertex cover C of size K for G, a solution of PCSF of cost $2m+2n+K$ can be constructed.

(2) Given a solution of PCSF of cost at most $2m+2n+K$, a vertex cover of size at most K can be constructed.

Note that then there should be a gap between $2m+2n+cn=(5+c)n$ and $2m+2n+c(1+\varepsilon_0)n=(5+c(1+\varepsilon_0))n$, i.e, PCSF is hard to approximate within a factor $1+\varepsilon_0=\frac{5+c(1+\varepsilon_0)}{5+c}$ unless $P=NP$.

Let's prove (1). Let T be a tree of H that contains

- edge $\{w,a_i\}$ iff $v_i \notin C$,
- edges $\{w,c_j^1\}, \{c_j^1,b_j\}$ iff $e_j^1 \notin C$
- edges $\{w,c_j^2\}, \{c_j^2,b_j\}$ iff $e_j^1 \in C$.

Since $\{w,a_i\}$ has cost 2 and cost of all other edges are 1, the total cost of T is $2(n-K)+2m$. Note that all the demands $\{w,b_j\}$ are connected either via c_j^1 or c_j^2 since C is a vertex cover. Also if $v_i \notin C$, then all three (since 3-reg.) demands where a_i appears are satisfied: edge $\{w,a_i\}$ is in T and if $v_i=e_j^1$ then edge $\{w,c_j^1\}$ is in T as well (note that if $v_i=e_j^2$ and $v_i \notin C$, then $e_j^1 \in C$ must hold and therefore $\{w,c_j^2\}$ is in T). Thus the total penalty is at most $3K$ (for 3 demands of a_i where $v_i \in C$) and thus the total cost of the solution is at most $2(n-K)+2m+3K=2n+2m+K$, as desired.

(2) Can be proved as well: it is not hard and remains as an exercise. ∎

Note that

I) the proof above is approximation-preserving in some sense as well (by some other definitions)

II) The fact that vertex cover was APX-hard on 3-regular graphs where $m=O(n)$ was very important. Indeed lots of other problems such as Maximum independent Set, minimum dominating set and max cut are APX-hard on 3-regular graphs [Alimonti'00] and thus very good for reductions.

Dense k-subgraph: Find a subgraph on k vertices with maximum # of edges.

→ The best upper bound is $O(n^{\frac{1}{4}+\varepsilon})$ in time $O(n^{\frac{1}{\varepsilon}})$ for any constant $\varepsilon > 0$. [Bhaskara, Charikar, Chlamtac, Feige, Vijayaraghavan '10]

→ However in terms of lower bounds, we know only there is no PTAS, i.e., APX-hard, if $NP \subseteq BPTIME(2^{n^\varepsilon})$, where BP is bounded-error probabilistic (randomized) class. [Khot '06]

→ Though there is such a big gap between upper and lower bounds, the people believe the correct bound should be $\Omega(n^c)$ for some constant c, even $c = \frac{1}{4}$.

→ Nowadays there are lots of problems which are Dense k-subgraph hard.

→ Indeed its minimization version, namely, minimum k-edge coverage, in which we want the minimum number of vertices in G whose induced subgraph has at least k edges is also very useful for reductions as well.

Thm: If there is a polynomial time f-approximation algorithm for minimum-k edge coverage, then there is a polynomial time $2f^2$-approximation algorithm for the dense k-subgraph problem.

Pf: left as an exercise

Note that dense subgraph (with no k) is polynomial-time solvable.

Cor: if dense k subgraph is $\Omega(n^c)$ hard, minimum k-edge coverage is $\Omega(n^{\frac{c}{2}})$ hard.

Now let's see a dense k subgraph problem:

k-forest: given a graph G with edge weights, a number k, and a set of pairs (s_i, t_i), $1 \le i \le h$, buy a set of edges to connect at least k pairs.

Thm: If there is an approximation preserving reduction from minimum k-edge coverage to k-forest. [note that k-Steiner tree has constant-approximation factor though]

Pf: For an instance of minimum k-edge coverage, we construct an instance star of k-forest as follows. Say the center of star is c and there is a unit-cost edge $[c, v]$ for each $v \in V(G)$. For each edge $[u, v] \in E(G)$, we put a demand pair (u, v). Now, it is easy to see any subgraph of the star connecting k pairs corresponds to a set of vertices whose induced subgraph has at least k edges and vice versa.

→ See references to [Feige, Kortsarz, Peleg '93] for many more reductions [in Google Scholar]

The Unique Coverage Problem

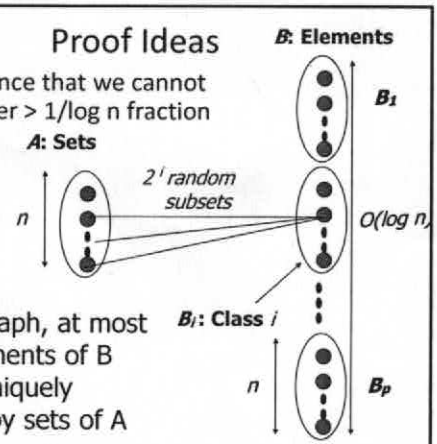
- **Simple *O(log n)* approximation algorithm**
  - Partition the elements into *log n* classes according to their degrees, i.e., the number of sets that cover an element
  - Let i be the class of maximum cardinality
  - Choose a set in S to be in S' with prob. $1/2^i$
- **Proof Sketch:** we uniquely cover $1/e^2$ fraction of elements of class i in expectation
- Can be de-randomized by the standard method of conditional expectation

Hardness Result

- The algorithm seems naïve
- Several other problems have the same solution
- Can we do better?
- It seems combination sometimes can be hard
- Theorem: This problem is hard to approximate within a factor better than $O(\log^c n)$, $0<c<1$, unless NP has a sub-exponential algorithm.
- $O(\log^{1/3} n)$ hard or even $O(\log n)$ hard under stronger but plausible complexity assumptions

Proof Ideas

- A bad instance that we cannot uniquely cover > 1/log n fraction



In this graph, at most $O(n)$ elements of B can be uniquely covered by sets of A

Proof Ideas

- **Bipartite Independent Set (BIS) problem:**
  - Given a bipartite graph $G(A \cup B, E)$ where $|A|=|B|=n$
  - Find a bipartite sub-graph $G'(A' \cup B', E')$ where $|A'|=a$, $|B'|=b$ and E' is an empty set.
- **Theorem:** Unless NP has sub-exponential algorithm, it is hard to decide between $(n^c, n/\log^d n)$-BIS and $(n^{c'}, n/\log^{d'} n)$-BIS where $0<c'<c \leq 1$ and $0 \leq d<d' \leq 1$
- Now between A and B_i we put a random matching and an instance of BIS where the edges remain with probability $1/2^{i-1}$
- For **Yes** instance, we have a unique cover of size $\Omega((c-c')n\log^{1-d} n)$
- For **No** instance, we have a unique cover of size at most $O((c-c')n\log^{1-d'} n)$ (***idea:*** the matching with independent set makes unique cover)
- The inapproximability threshold can be improved under other stronger but still plausible complexity assumptions

Boosting/Amplifying Gaps:

We have seen the size of maximum clique (complement of independent set) is APX-hard and thus hard to approximate within a $(1-\varepsilon_0)$ factor for some fixed $\varepsilon_0 > 0$. Now let's boost the hardness via graph product.

Def: Given two graphs $G_1=(V_1,E_1)$ and $G_2=(V_2,E_2)$ we define graph product $G = G_1 \times G_2$ as follows:

Let $E_1' = E_1 \cup \{(u,u) \mid u \in V_1\}$ (we add loops to all vertices)

$E_2' = E_2 \cup \{(a,a) \mid a \in V_2\}$

Now $V(G) = V_1 \times V_2$ and $E(G) = \{((u,a),(v,b)) \mid (u,v) \in E_1' \text{ and } (a,b) \in E_2'\}$

Note that $G_1 \times G_2$ is not necessarily the same as $G_2 \times G_1$.

First a lemma whose proof is simple and leaves as an **exercise**.

Lm: $\text{max-clique}(G) = \text{max-clique}(G_1) \cdot \text{max-clique}(G_2)$

Now, if we define G^k recursively as follow: $G^1 = G$ and $G^k = G^{k-1} \times G$.

→ By above lemma $\text{max-clique}(G^k) = (\text{max-clique}(G))^k$.

As we proved earlier, in max-clique (and independent set) there is a gap between $\text{opt}(G) = m = \frac{|V(G)|}{3}$ and $\text{OPT}(G) < (1-\varepsilon_0)m = (1-\varepsilon_0)\frac{|V(G)|}{3}$. Thus it is NP-hard to distinguish between $\text{max-clique}(G^k) = m^k$ and $\text{max-clique}(G^k) < (1-\varepsilon_0)^k m^k$. Thus max-clique is NP-hard to approximate within $(1-\varepsilon_0)^k$ but how large k can be? The reduction from G to G^k needs to be poly-time and thus k should be a constant since $m = \frac{|V(G)|}{3}$ and m^k should be polynomial in $|V(G)|$. However by choosing k large enough, we can obtain a hardness c for max-clique for any $c > 0$.

→ Indeed as aforementioned clique is $\Omega(n^{1-\varepsilon})$ hard, though the proof is more involved, it still uses boosting. Indeed the technique of boosting is very important for PCP theorem and hardness of Max-Rep & Min-Rep

Unique-game Conjecture Consequences:

→ As we mentioned the truth of UCG would imply the optimality of many known approximation algorithms (assuming $P \neq NP$). To just give an example Max2-sat current best approximation 0.940, Max-cut current best approximation 0.878 and Vertex-cover current best approximation 2 all would be tight. [KKMO'07, KR'03]

→ Currently there is no consensus regarding the truth of UCG. Some stronger forms are defined and some proved to be wrong. A famous one is Small Set Expansion (SSE) Conjecture which assumes unique game conjecture is true for somewhat expanding graphs (which is NOT proved to be wrong).

→ Assuming SSE, treewidth, pathwidth, minimum cut linear arrangement and interval graph are hard to approximate within any constant factor. [Austrin, Pitassi, Wu'12]

→ The proofs of above is involved and can be seen in the References.

→ In 2010, Arora, Barak and Steurer found a subexponential-time approx. alg. for unque games but of course this does not refute the conjectur.

Integrality Gaps & Unique Game Conjecture:

Integrality Gap of an LP (and its generalization SDP): The ratio between optimum fractional and optimum Integral.

→ Linear Programming (LP) and Semi-Definite-programming (SDP) are common approaches to model a relaxed version of an NP-hard problem.

→ It is known that approximation factors cannot be better than Integrality gap of an LP (SDP) used in the approximation algorithms unless we use side-bounds in the approx. alg.

→ However Integrality Gaps of many natural LP are indeed the hardness of approximations for famous problems such as Vertex cover, set cover, Group steiner tree, node-weighted steiner tree, etc.

→ Roughly speaking UCG says SDP Gaps (and LP Gaps in case they are equal to SDP Gaps for some problems such as vertex cover) are hardness of Approximations as well and thus in a sense SDP is the most powerful technique for Approximation. [Raghavendra'08]