Approximation Algorithms

Scribe: Muhan Li

Introduction

(Quoted from wiki)

Approximation algorithms are efficient algorithms that find approximate solutions to optimization problems (in particular NP-hard problems) with **provable guarantees** on the distance of the returned solution to the optimal one.

We usually use the approximation factor as a constraint.

For a minimization problem:

$$ALG(I) \leq A * OPT(I)$$

Where:

1. A - approximation factor, $A \geq 1$
2. $ALG(I)$ - cost of some problem instance returned by our algorithm
3. $OPT(I)$ - cost returned by the optimal solution.

For a maximization problem:

$$ALG(I) \geq A * OPT(I)$$

Where $A \leq 1$

The edge cover problem

We have talked about this problem in "Parameterized Algorithms", the exact mathematical definition for this problem is:

> For graph $G = (V, E)$, find set $S \subseteq V$ with min $|S|$, such that $\forall e = (u, v) \in E, u \in S \lor e \in S$.

An algorithm

> While ($\exists$ uncovered edge (u, v)):
>
> - pick an uncovered (u, v).
> - Add both u and v to S.
>
> return S.

(Why choose two nodes? Eg: in the star topology, if choose one node, the worst case is choosing all leaf nodes and not choosing the center node, while choosing two nodes will definitely cover the center node.)

Approximation analysis

Theorem: approximation factor of this algorithm is 2.

Proof:

Let $e_1, \ldots, e_k$ be the k edges ALG have chosen:

Lemma 1:

Edges $e_1, \ldots, e_k$ selected by ALG must be disjoint and not share any end nodes.

Lemma 1 proof:

If edge e_i and e_j share one end node u, suppose e_i is added to S before e_j, then before e_j is added to S, since u of e_i is in the cover set S, then e_j must be covered, which contradicts our assumption that e_j is not covered after adding e_i and will be added to S.

Now we have all the tools we need to prove the validity of algorithm and its approximation factor.

Validity proof:

Suppose there exists an edge e_i not covered by ALG, since none of its end nodes are be covered by ALG, both of its end nodes must be added to S by definition of our algorithm, which contradicts our assumption.

Approximation factor proof:

Since $ALG(I) = 2k$, and because OPT has to choose half of the nodes of $e_1, \ldots, e_k$ to cover all k disjoint edges, $OPT(I) \geq k$, and $ALG(I) \leq 2OPT(I)$.

The set cover problem

Find the minimum set of indices I, s.t.

$$\bigcup_{i \in I} s_i = u$$

(The collection of subset family $\{s_i\}$ is predetermined)

An algorithm

let V_k be the set of uncovered elements, and $V_0 = u$.

> while ($V_t \neq \emptyset$)
>
> - Find s_j that covers most elements in V_t
> - Add s_j to solution I
> - $V_{t+1} = V_t \setminus S_j$
>
> return solution I

Approximation analysis

Claim: Approximation factor of the algorithm is $O(\log n)$, where $n = |u|$.

Proof:

Since $V_0 = V$, the size of initial uncovered set is $|V_0| = n$.

We can prove a key observation of V_1 size $|V_1|$:

> Suppose $OPT(I) = k$, then there exists some set s_j which satisfies $|s_j| \geq \frac{n}{k}$ (prove by contradiction).
>
> Therefore our algorithm chooses $s_1, |s_1| \geq \frac{n}{k}$, since it always choose the set with the most uncovered elements.
>
> Therefore an upper bound on the size of $|V_1|$ is:

> $$V_1 = V - s_j \Rightarrow |V_1| \leq n - \frac{n}{k} = (1 - 1/k) * n$$

We would like to prove $|V_t| \leq (1 - 1/k)^t * n$:

Let the remaining uncovered set be covered by the union of s_j: $V_t = \cup_{j, j \leq t} s_j$, then:

$$\exists j, s.t. |V_t \cap s_j| \geq \frac{|V_t|}{k}$$

Since $OPT(I) = k$ (prove by contradiction).

Finally by induction:

$$\begin{aligned}
|V_{t+1}| &\leq |V_T| - |s_j| \\
&= |V_t| - \frac{|V_t|}{k} \\
&= (1 - \frac{1}{k})|V_t| \\
&\leq (1 - \frac{1}{k}) * (1 - \frac{1}{k})^t * n \\
&= (1 - \frac{1}{k})^{t+1} * n
\end{aligned}$$

let $t = k * logn$, gets $|V_t| \leq 1$.

Weighted set cover problem

Assign weight w_i to set s_i, find set cover which $min \sum_{i \in I} w_i$.

Let binary random variable x_i indicate if set s_i is chosen in the solution (when $x_i = 1$, s_i is chosen)

Need to solve (which means every element is at least covered by one set):

$$min \sum_{i=1}^{m} w_i * x_i \qquad s.t. \forall u \in U, \sum_{i: u \in s_i} x_i \geq 1$$

This is a NP-hard integer programming problem. We can do a relaxation and convert it to a polynomial complexity linear programming problem: let $x_i \in [0, 1]$ instead of $x_i \in \{0, 1\}$.

Solve the new linear programming problem, obtain $x_1, \ldots, x_m$, then choose each s_i with probability:

$$min\{2x_i \log n, 1\}$$

It is clear that $LP \leq OPT$ (which means LP is better), because LP includes the set of IP (integer programming) solutions since it works in a relaxed environment.

Need to prove $ALG \leq O(\log n) * LP$:

$$
\begin{aligned}
&P\{u \text{ is not covered}\}\\
&= P\{s_i \text{ is not in solution } \forall i, u \in s_i\}\\
&= \prod_{i:u\in s_i} P\{s_i \text{ is not in solution}\}\\
&= \prod_{i:u\in s_i} (1 - x_i 2\log n)^+ \ (+\text{ means replace with zero if prob} < 0)\\
&(\text{because } e^{-t} \geq 1-t)\\
&\leq \prod_{i:u\in s_i} e^{-x_i 2\log n}\\
&= e^{\sum_{i:u\in s_i} x_i 2\log n}\\
&(\text{because } \sum x_i \geq 1)\\
&\leq e^{-2logn}\\
&= \frac{1}{n^2}
\end{aligned}
$$

therefore:

$$
P\{\text{at least one element is not covered}\} \leq \frac{1}{n}
$$

$$
\begin{aligned}
&\mathbb{E}[\text{cost of solution}]\\
&= \mathbb{E}[\sum_i \mathbb{I}\{s_i \text{ is chosen in solution}\} * w_i]\\
&= \sum_i P\{s_i \text{ is chosen in solution}\} * w_i\\
&\leq \sum 2(\log n) x_i w_i\\
&= 2(\log n) \sum x_i w_i
\end{aligned}
$$

reference essay