Approximation Algorithms

Yufei Tao

Department of Computer Science and Engineering
Chinese University of Hong Kong

We have already seen that some **decisions problems** (very likely) cannot be solved in polynomial time.

In practice, however, many problems are not decision problems, but rather are so-called **optimization problems**. Nevertheless, in this lecture, we will see that NP-hardness on decision problems implies that many optimization problems (very likely) cannot be solved in polynomial time, either.

So what can do on those optimization problems?

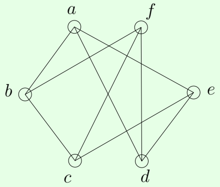
Instead of declaring failure immediately, we can find **near-optimal** solutions in polynomial time, namely, trading away quality for efficiency. Such algorithms are called **approximation algorithms**.

Consider an undirected graph $G = (V, E)$.

Consider a subset $S \subseteq V$. S is a **vertex cover** if every edge $\{u, v\} \in E$ is adjacent to at least one vertex in S, i.e., $u \in S$, $v \in S$, or both.

The vertex cover problem: Find a vertex cover of the smallest size.

Example:



An optimal solution is $\{a, f, c, e\}$.

The vertex cover problem: Find a vertex cover of the smallest size.

Note that this is **not** a decision problem.
Instead this is an **optimization** problem: from all the vertex covers, we want the optimal one.
Earlier, we already introduced the **decision version** of the problem:

The vertex cover decision problem: Given an integer k, decide whether there is a vertex cover with at most k vertices.

We state the next result without proof:

Theorem: The vertex cover decision problem is NP-complete.

Do you think the vertex cover (optimization) problem can be solved in polynomial time? See the next slide for the answer.

Theorem: If we can solve the vertex cover problem in polynomial time, then we can solve every problem in $\mathtt{NP}$ in polynomial time.

Proof: Suppose that we can obtain the smallest size k^* of all vertex covers in polynomial time. Then, we can trivially solve the vertex cover decision problem in polynomial time by comparing k^* with k.

As mentioned earlier, the vertex cover decision problem is NP-complete. Hence, that it can be solved in polynomial time means that all the problems in $\mathtt{NPC}$ can be solved in polynomial time. □

Corollary: Unless $\mathtt{P} = \mathtt{NP}$, the vertex cover problem cannot be solved in polynomial time.

Now what? Should we give up in despair?
Well, not if you are willing to **sacrifice the quality** of your solution a little.

Denote by OPT the smallest size of vertex covers.
Next, we will show that it is possible to obtain in polynomial time a vertex cover that includes at most $2 \cdot OPT$ vertices.

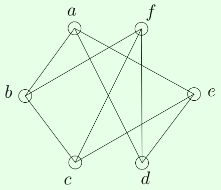
The constant 2 is referred to as the **approximation ratio**.
Our algorithm is referred to as an **approximation algorithm**.

An Approximation Algorithm

```
1. $S = \emptyset$
2. while $E$ is not empty
3.     pick an arbitrary edge $\{u, v\}$ in $E$
4.     add $u, v$ to $S$
5.     remove from $E$ all the edges of $u$
6.     remove from $E$ all the edges of $v$
7. return $S$
```

Example

Consider $G = (V, E)$ is the following graph:



Suppose we pick edge $\{b, c\}$.
Then, $S = \{b, c\}$, and $E = \{\{a, e\}, \{a, d\}, \{d, e\}, \{d, f\}\}$.

Suppose we pick edge $\{a, e\}$ next.
Then, $S = \{a, b, c, e\}$, and $E = \{\{d, f\}\}$.

Finally we pick edge $\{d, f\}$.
The final $S = \{a, b, c, d, e, f\}$.

It is trivial to implement the algorithm in time polynomial to $|V|$ and $|E|$. Next, we will prove

Theorem: $|S| \leq 2 \cdot OPT$.

In other words, our algorithm achieves an approximation ratio of 2.

We denote by M the set of edges picked. For instance, in the previous example, M consists of $\{b, c\}$, $\{a, e\}$, and $\{d, f\}$.

Claim: The edges in M do not share any vertices.

Proof: Suppose that M has two edges e_1, e_2 both adjacent to a vertex v. Without loss of generality, let us assume that e_1 was picked before e_2. However, after e_1 is picked, all the edges of v are deleted. Hence, it is impossible for e_2 to be picked, giving a contradiction. □

Claim: $|M| \leq OPT$.

Proof: Any vertex cover must include at least a vertex of each edge in M. Then, $|M| \leq OPT$ follows from the previous claim. □

Thus, the theorem of the previous slide follows from the obvious fact that $|S| = 2|M|$.

We have seen that even though the vertex cover problem can unlikely be solved in polynomial time **exactly**, we can still find a good **approximate solution** (with a small approximation ratio) in polynomial time.

This **trade-quality-for-efficiency approach** is very common in tackling NP-hardness, and is a major topic in computer science.

We will see another problem next.

We are given n sets $S_1, S_2, ..., S_n$.
Define $U = \bigcup_{i=1}^{n} S_i$, called the **universe**.

The set cover problem: From $\{S_1, S_2, ..., S_n\}$, find the smallest number of sets whose union is U.

Example:
$S_1 = \{1, 2, 3, 4\}$
$S_2 = \{2, 5, 7\}$
$S_3 = \{6, 7\}$
$S_4 = \{1, 8\}$
$S_5 = \{1, 2, 3, 8\}$
$U = \{1, 2, 3, 4, 5, 6, 7, 8\}$
An optimal solution is S_1, S_2, S_3, S_4.

The set cover problem: From $\{S_1, S_2, ..., S_n\}$, find the smallest number of sets whose union is U.

This is an **optimization** problem. Earlier, we introduced the **decision version** of the problem:

The set cover decision problem: Given an integer k, decide whether $\{S_1, S_2, ..., S_n\}$ contains k sets whose union is U.

We state the next result without proof:

Theorem: The set cover decision problem is NP-complete.

You can then prove easily:

Corollary: Unless $\mathtt{P} = \mathtt{NP}$, the set cover problem cannot be solved in polynomial time.

Denote by OPT the smallest number of sets whose union is U.

Next, we will show that it is possible to obtain in polynomial time at most $(1 + \ln |U|) \cdot OPT$ sets whose union covers U.

That is, we will achieve an **approximation ratio** of $1 + \ln |U|$.

Approximation Algorithm

```
1. $\mathcal{S} = \emptyset$
2. **while** $U$ still has elements not covered by the sets in $\mathcal{S}$
3.     $S \leftarrow$ the set in $\{S_1, ..., S_n\}$ that includes the largest
       number of **newly-covered elements**, namely, elements in $S$
       that are **not yet covered** by the sets in $\mathcal{S}$
4.     add $S$ to $\mathcal{S}$
5. **return** $\mathcal{S}$
```

Example

$S_1 = \{1, 2, 3, 4\}$, $S_2 = \{2, 5, 7\}$, $S_3 = \{6, 7\}$. $S_4 = \{1, 8\}$
$S_5 = \{1, 2, 3, 8\}$

- At the beginning, $\mathcal{S} = \emptyset$.
- Then, the algorithm adds S_1 or S_5 to $\mathcal{S}$; since the choice is arbitrary, suppose that S_1 is added.
- The second set added to $\mathcal{S}$ can be S_2 or S_3, because each of them has **two** newly-covered elements (e.g., $5, 7$ for S_2); suppose that S_2 is added.
- The next set added can be S_3, S_4, or S_5 (each of them has **one** newly-covered element); suppose that S_3 is added.
- Still the next set added can be S_4 or S_5 (each having **one** newly-covered element).

Now the algorithm terminates with $\mathcal{S} = \{S_1, S_2, S_3, S_4\}$.

It is trivial to implement the algorithm in time polynomial to $\sum_{i=1}^{n} |S_i|$. Next, we will prove

Theorem: $|\mathcal{S}| \le 1 + (\ln |U|) \cdot OPT$.

Denote by $t = |\mathcal{S}|$.
Without loss of generality, suppose that $S_1, S_2, ..., S_t$ are the sets in $\mathcal{S}$, chosen in the order shown.
Denote by z_i $(1 \le i \le t)$ the number of elements **yet to be covered** after S_i has been chosen.
Specially, define $z_0 = |U|$.

Note: $z_t = 0$ and $z_{t-1} \ge 1$. **Think:** why?

Denote by $\mathcal{S}^*$ an optimal solution, namely, $OPT = |\mathcal{S}^*|$.

Lemma: For $i \in [1, t]$, it holds that

$$z_i \le z_{i-1} \cdot \left(1 - \frac{1}{OPT}\right).$$

Proof: Consider the moment after $S_1, ..., S_{i-1}$ have been chosen by the algorithm. By definition, there are still z_{i-1} elements in U that have not been covered by $S_1, ..., S_{i-1}$.

Claim: At this moment, there is at least a set that covers at least z_{i-1}/OPT newly-covered elements.

Proof: Follows immediately from the fact that the z_{i-1} elements must be covered by the OPT sets in $\mathcal{S}^*$. $\square$

Hence, S_i covers at least z_{i-1}/OPT newly-covered elements. The number of elements that have not been covered by $S_1, ..., S_{i-1}, S_i$ is therefore at most $z_{i-1} \cdot (1 - \frac{1}{OPT})$. $\square$

It then follows from the lemma that

$$\begin{aligned}
z_{t-1} &\le z_{t-2} \cdot \left(1 - \frac{1}{OPT}\right) \\
&\le z_{t-3} \cdot \left(1 - \frac{1}{OPT}\right)^2 \\
&\ldots \\
&\le z_0 \cdot \left(1 - \frac{1}{OPT}\right)^{t-1} = |U| \cdot \left(1 - \frac{1}{OPT}\right)^{t-1} \\
&\le |U| \cdot e^{-\frac{t-1}{OPT}}
\end{aligned}$$

where the last inequality used the fact that $1 + x \le e^x$ for any real value x.

As $z_{t-1} \ge 1$, we have:

$$1 \le |U| \cdot e^{-\frac{t-1}{OPT}}$$

which resolves to $t \le 1 + (\ln |U|) \cdot OPT$. □