2-Approximation Algorithm for Vertex Cover

VertexCover(G)

```
1: C ← ∅
2: while E ≠ ∅ do
3:     select an edge (u, v) ∈ E, C ← C ∪ {u, v}
4:     Remove from E every edge incident on either u or v
5: return C
```

- Let the set C and C^* be the sets output by above algorithm and an optimal alg, respectively. Let S be the set of edges selected.
- Since no two edge in S are covered by the same vertex (Once an edge is picked in line 3, all other edges that are incident on its endpoints are removed from E in line 4), we have $|C^*| \geq |S|$;
- As we have added both vertices of edge (u, v), we get $|C| = 2|S|$ but C^* have to add one of the two, thus, $|C|/|C^*| \leq 2$.

Outline

Summary

- We consider decision problems
- Inputs are encoded as $\{0, 1\}$-strings

Def. The complexity class P is the set of decision problems X that can be solved in polynomial time.

- Alice has a supercomputer, fast enough to run an exponential time algorithm
- Bob has a slow computer, which can only run a polynomial-time algorithm

Def. (Informal) The complexity class NP is the set of problems for which Alice can convince Bob a yes instance is a yes instance

Summary

Def. B is an efficient certifier for a problem X if

- B is a polynomial-time algorithm that takes two input strings s and t
- there is a polynomial function p such that, $X(s) = 1$ if and only if there is string t such that $|t| \leq p(|s|)$ and $B(s,t) = 1$.

The string t such that $B(s,t) = 1$ is called a certificate.

Def. The complexity class NP is the set of all problems for which there exists an efficient certifier.

Summary

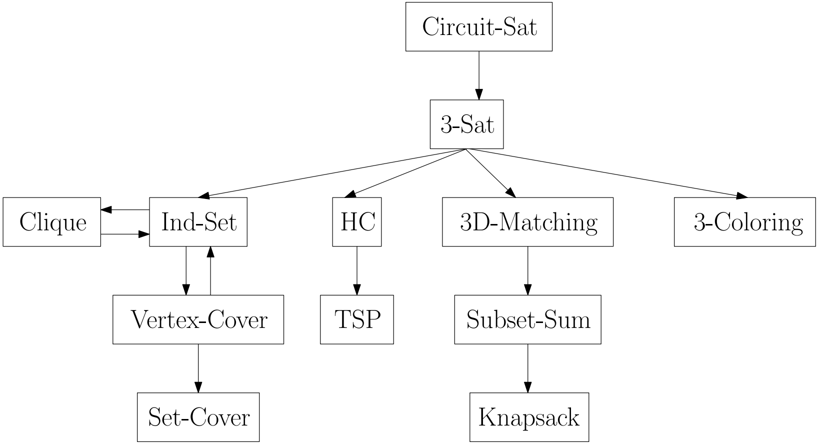
Def. Given a black box algorithm A that solves a problem X, if any instance of a problem Y can be solved using a polynomial number of standard computational steps, plus a polynomial number of calls to A, then we say Y is polynomial-time reducible to X, denoted as $Y \leq_P X$.

Def. A problem X is called NP-complete if

1. $X \in$ NP, and
2. $Y \leq_\text{P} X$ for every $Y \in$ NP.

- If any NP-complete problem can be solved in polynomial time, then $P = NP$
- Unless $P = NP$, a NP-complete problem can not be solved in polynomial time

Summary



Circuit-Sat
3-Sat
Clique
Ind-Set
HC
3D-Matching
3-Coloring
Vertex-Cover
TSP
Subset-Sum
Set-Cover
Knapsack

Summary

Proof of NP-Completeness for Circuit-Sat

- Fact 1: a polynomial-time algorithm can be converted to a polynomial-size circuit
- Fact 2: for a problem in NP, there is a efficient certifier.
- Given a problem $X \in$ NP, let $B(s, t)$ be the certifier
- Convert $B(s, t)$ to a circuit and hard-wire s to the input gates
- s is a yes-instance if and only if the resulting circuit is satisfiable

- Proof of NP-Completeness for other problems by reductions