Complexity

Definitions

We say $Y \in$ **P** if
1) Y has a polynomial time **solution**.

We say $Y \in$ **NP** if
1) Y has a polynomial time **checker**.

We say $Y \in$ **NP-Complete** if it **as hard** as any other problem in NP.
1) $Y \in$ NP
2) $\forall\ X \in$ NP, $X \leq_p Y$

Theorems

P $\subseteq$ NP.

Let X be a problem.
If Y $\in$ NP-Complete and Y $\leq_p$ X
then X $\in$ NP-Complete.

Let X, Y, Z be problems.
If X $\leq_p$ Y, and Y $\leq_p$ Z,
then X $\leq_p$ Z.

NP Complete Problems

3-SAT - $F = (x_1 \vee x_2 \vee \overline{x_3}) \wedge ... \wedge (x_{n-2} \vee \overline{x_{n-1}} \vee x_n)$
Given a formula, F, of and's of k clauses, $c_1, /..., c_k$, or's of 3 literals, is the formula satisfiable?
Vertex Cover - $S \subseteq V$ if $\forall\ (u,\ v) \in E$, $u \in S$ or $v \in S$.
Given a graph, what is the minimal subset of nodes such that every edge is encaptured?
Independent Set - $S \subseteq V$ if $\forall u,\ v \in V$, $u,\ v$ are not connected.
Given a graph, what is the max subset of nodes such that each component is disjoint?
Hitting Set - $H \subseteq A$ such that $H \cap b_i \neq \emptyset$.
Given a set A and B, a set of subsets of A, what is the minimum set H such that H and $b_i \in\ B$ have a similar element?
Clique - $S \subseteq V$ such that $\forall u,\ v \in S$, $u,\ v$ are connected.
Given a graph, what is the max size subgraph such that all nodes are connected in the subgraph.
3-Color - Given a graph, G, is it possible to color the nodes in a way such that no incident edge has nodes of the same color?

NP Complete Reductions

Clique $\iff$ Independent Set

Given a graph G, if we find the maximum independent set, S, of $G*$, the graph with the complement of edges, then S is the max clique of G, and vice-versa.
By definition of independent set, S has no similar edges, so in G, S will have edges going to every other node in S.

Vertex Cover $\iff$ Independent Set

Given a graph G, if we find the maximum independent set, S, of G, the minimum vertex cover of G is $V - S$, and vice-versa.

3-SAT $\leq_p$ Independent Set

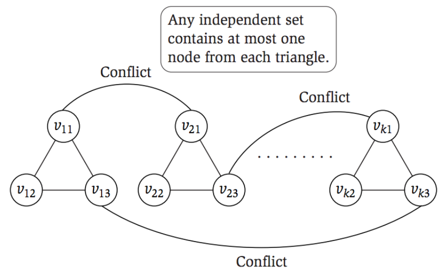
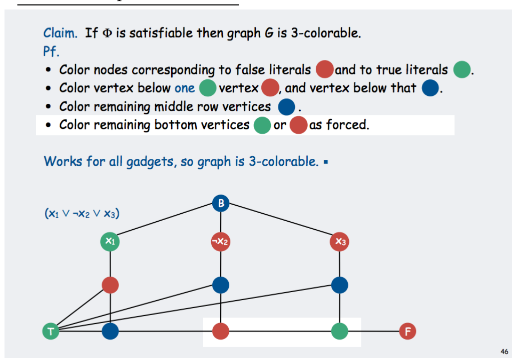


Figure 8.3 The reduction from 3-SAT to Independent Set.

3-Sat $\leq_p$ 3-Color



PSPACE

Definitions

We say $Y \in$ **PSPACE** if
1) Y has a polynomial space solution.

Theorems

3-SAT $\in$ PSPACE, if we use a binary counter of length n, where n is the number of literals.

NP $\subseteq$ PSPACE, since we can reduce all problems in NP to 3-SAT, and 3-SAT $\in$ PSPACE.

Q-SAT $\in$ PSPACE, since we only have to keep one bit of information per quantifier as we recur and backtrack through the quantifiers.

Turing Machines

Definitions

A **turing machine** is a 7-tuple $(Q,\ \Sigma,\ \Gamma,\ \delta,\ q_0,\ q_{accept},\ q_{reject})$
A **configuration** is a string. For example, $0q_01$, shows that the head is on 1, has written 0, and is in the state q_0.
Q - the set of **states**.
Σ - the **input alphabet**.
Γ - the **tape alphabet**, where $u \in \Gamma$ and $\Sigma \subset \Gamma$.
δ - $Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$, the **transition function**.

The set of strings a machine, M, accepts is called the **language** of M. We say a language is **recognizable** if some Turing machine accepts, rejects, or loops. We say a language is **decidable** if some Turing machine halts.

Theorems

Decidable $\implies$ recognizable, but recognizable does not necessarily imply decidable.

The **Halting Problem** is undecidable.
Assume for the sake of contradiction we have a machine, H, that is a decider on the inputs of any arbitrary Turing machine, and any arbitrary string. Let D be a machine that takes what H outputs and negates it. If we feed D into itself, we reach a contradiction.

Approximations

Load Balancing

Given m machines with n jobs, $t_1, ..., t_n$, what is the minimum makespan?
We can use a greedy solution where we find the machine with the minimum load and add the job to it.
This solution is bounded by $T \leq 2T^*$.
1) $T^* \geq max\{t_i\}$
2) $T^* \geq \frac{1}{m} \sum\limits_{i=1}^{n} t_i = \frac{1}{m} W$
Consider the machine with the makespan before the last job. The machine has a load of $T - t_j$, and since it was the minimum of the rest of the m machines, we know that the total load was $m(T - t_j) \leq W$, the total time of all jobs. So $T - t_j \leq \frac{W}{m}$ and by 1) and 2), $T \leq 2T^*$.

If we sort the jobs by descending times, the solution is bounded by $T \leq 1.5T^*$.
If there are more jobs than machines, then one machine will have at least 2 jobs. Since the jobs are sorted descending, then $T^* \geq 2t_{m+1}$, and then we have the inequality $\frac{T^*}{2} \geq t_{m+1}$. Now that we have this, we can do a similar proof to the first and we see that $T \leq 1.5T^*$.

Homework Problems

Truck Scheduling

Given a set of weights, $w_1,\ ..., w_n$, and a value K, what is the minimum number of trucks needed? If we use a greedy algorithm adds trucks as needed, we will use $T \leq 2T^*$ trucks. Let W be the sum of the weights. $T^* \geq \frac{W}{K}$ since each truck holds at most K. Consider the result from greedy, in the form $2q + 1$ trucks. If we split this into consecutive groups of two, we see that each pair has a weight of at least K. Since we have $q+1$ pairs, $K(q+1) \leq W$ and we see $q + 1 \leq \frac{W}{K}$, so $T^* \geq q + 1$, and by our assumption $T = 2q + 1$ which is less than a factor of 2 greater.

Odd Length Undecidable Language

Find a language that decides if a string is of odd length, but also is undecidable itself. Consider the language A_{TModd}. Since the input for A_{TM} is ¡M, w¿, a Turing machine and input string, we can let $A_{TModd} = \{< M, w > c < M, w > | < M, w > \in A_{TM}\}$. We can see the length of the string is odd, but since A_{TM} is undecidable, A_{TModd} is undecidable.

3-SAT $\leq_p$ 3-Color

Suppose there is a satisfying assignment for 3-SAT. We color $\bar{G}$ with base, true, false. If $x_i = 1$, color v_i with true, and $|v_i$ with false. The opposite for $x_i = 0$. When we extend this coloring into the subgraph, we see we have a three coloring.
Suppose there is a 3-color of $\bar{G}$. Each node v_i is assigned with a true color or false color. We set the corresponding x_i appropriately. We claim that in each 3-SAT clause, at least one term has a truth value = 1. If this was not true, then all the three of the corresponding nodes has the false color and therefore no but no clause subgraph is such. Contradiction.