Chapter 4: Using NP-Completeness to Analyze Subproblems

- Analyzing sub-problems
- Number problems & strong NP-completeness
- Time complexity as a function of natural parameters
Proving a problem NP-complete is not the end, but rather the beginning:
- Sub-problems (special cases) may be solved in polynomial time
- Heuristics with provable performance bounds may exist
- Pseudo-polynomial time algorithms may exist

An actual occurrence of the problem in the real world may, in fact, be a special case.
Sub-Problems

Recall that a problem \( \Pi \) consists of:

- A set \( D \) of instances, and
- A set \( Y_{\Pi} \subseteq D \) of “yes” instances

A sub-problem of a problem \( \Pi = (Y_{\Pi}, D) \) is a problem \( \Pi' = (Y'_{\Pi}, D') \) where:

- \( D' \subseteq D \), and
- \( Y'_{\Pi} = Y_{\Pi} \cap D' \)

Note that a sub-problem is sometimes referred to as a *special case.*
Examples of Restrictions

- **Graph problems:**
  - vertex degree limits
  - planarity, bipartite, acyclic

- **Set problems:**
  - limits on set sizes
  - number of common elements between sets
  - total number of sets

- **Logic problems:**
  - size of clauses
  - number of times a literal occurs

- **Scheduling problems:**
  - limits on task lengths, deadlines, orderings

- **Arbitrary parameters:**
  - number of colors, assignments, bins
Examples of Sub-Problems

- Satisfiability sub-problems:
  - 2-SAT, 3-SAT, 4-SAT, K-SAT for any fixed $K \geq 0$.
  - Each clause contains only “positive” variables, or “negative” variables.
  - Each literal occurs in at most 5 clauses (more generally $K$ clauses, for fixed $K \geq 1$)

- Note that 3SAT is not a sub-problem of 4SAT.

- Similarly, 3DM is not a sub-problem of 4DM.

- Note, however, that AM3SAT is a sub-problem of AM4SAT (what the heck is AM?).
Examples of Sub-Problems

- Thus, a problem $\Pi$ may have many sub-problems, many of which are sub-problems of each other.

- Any of these, including $\Pi$, may be NP-complete or solvable in polynomial time.

- Note, however, that if $\Pi \in P$ then every sub-problem $\Pi' \in P$. 

![Diagram of sub-problems](image)
Examples of Sub-Problems

- **Graph colorability sub-problems:**
  - Graph colorability for planar graphs.
  - Graph 2-colorability, 3-colorability, $K$-colorability for any fixed $K \geq 0$.
  - Graph $K$-colorability for planar graphs, for any fixed $K \geq 0$.
  - Graph $K$-colorability for graphs where no vertex has degree exceeding $J$, for any fixed $J \geq 0$ and any fixed $K \geq 0$.
  - Graph $K$-colorability for planar graphs where no vertex has degree exceeding $J$, for any fixed $J \geq 0$ and any fixed $K \geq 0$.
  - Graph $K$-colorability for acyclic planar graphs where no vertex has degree exceeding $J$, any fixed $J \geq 0$ and any fixed $K \geq 0$.

- Exercise – draw the boundary diagram for the above set of problems.
How does one approach a new problem or sub-problem?

- Obsessively pursue an NP-completeness proof.

- Obsessively pursue a polynomial-time algorithm.

- Alternate between the two, until one approach leads to success.

- Alternate between related sub-problems, pursuing both NP-completeness results, and polynomial-time algorithms.

- Place special emphasis on the border between the unknown and the known.

For NP-completeness proofs:
  - Modify an NP-completeness proof for the more general problem so that it applies to the sub-problem.
  - Transform the more general problem, known to be NP-complete, to the sub-problem.
Dimensional Matching

1. Attempt to constrain the NP-completeness proof for this problem to apply to the sub-problem.

2. Attempt to generalize the poly-time algorithm for this problem to apply to the more general problem.

Figure 4.2: One possible state of knowledge about subproblems of an NP-complete problem Π1. Problems are represented by circles, filled-in if known to be NP-complete, empty if known to be in P, and dotted if “open.” An arrow from Π1 to Π2 signifies that Π1 is a subproblem of Π2.
**Precedence Constrained Scheduling**

**PRECEDENCE CONSTRAINED SCHEDULING**

INSTANCE: A finite set $T$ of “tasks” (each assumed to have “length” 1), a partial order $<$ on $T$, a number $m \in \mathbb{Z}^+$ of “processors,” and an overall “deadline” $D \in \mathbb{Z}^+$.

QUESTION: Is there a “schedule” $\sigma: T \rightarrow \{0,1,\ldots,D\}$ such that for each $i \in \{0,1,\ldots,D\}$, $|\{t \in T: \sigma(t) = i\}| \leq m$, and such that, whenever $t < t'$, then $\sigma(t) < \sigma(t')$?

Other scheduling problems we have considered:

**MULTIPROCESSOR SCHEDULING**

INSTANCE: A finite set $A$ of “tasks,” a “length” $l(a) \in \mathbb{Z}^+$ for each $a \in A$, a number $m \in \mathbb{Z}^+$ of “processors,” and a “deadline” $D \in \mathbb{Z}^+$.

QUESTION: Is there a partition $A = A_1 \cup A_2 \cup \ldots \cup A_m$ of $A$ into $m$ disjoint sets such that:

$$\max\{ \sum_{a \in A_i} l(a) : 1 \leq i \leq m \} \leq D$$
**Precedence Constrained Scheduling**

**Precedence Constrained Scheduling**

**INSTANCE:** A finite set $T$ of “tasks” (each assumed to have “length” 1), a partial order $<$ on $T$, a number $m \in \mathbb{Z}^+$ of “processors,” and an overall “deadline” $D \in \mathbb{Z}^+$.

**QUESTION:** Is there a “schedule” $\sigma: T \rightarrow \{0, 1, \ldots, D\}$ such that for each $i \in \{0, 1, \ldots, D\}$, $|\{t \in T: \sigma(t) = i\}| \leq m$, and such that, whenever $t < t'$, then $\sigma(t) < \sigma(t')$?

Other scheduling problems we have considered:

**Minimum Tardiness Sequencing**

**INSTANCE:** A finite set $T$ of “tasks,” each $t \in T$ having “length” 1 and a “deadline” $d(t) \in \mathbb{Z}^+$, a partial order $<$ on $T$, and a non-negative integer $K \leq |T| - 1$.

**QUESTION:** Is there a “schedule” $\sigma: T \rightarrow \{0, 1, \ldots, |T| - 1\}$ such that $\sigma(t) \neq \sigma(t')$ whenever $t \neq t'$, such $\sigma(t) < \sigma(t')$ whenever $t < t'$, and such that $|\{t \in T: \sigma(t) + 1 > d(t)\}| \leq K$?
Precedence Constrained Scheduling

PRECEDECE CONSTRANINED SCHEDULING

INSTANCE: A finite set $T$ of “tasks” (each assumed to have “length” 1), a partial order $\prec$ on $T$, a number $m \in \mathbb{Z}^+$ of “processors,” and an overall “deadline” $D \in \mathbb{Z}^+$.

QUESTION: Is there a “schedule” $\sigma: T \rightarrow \{0,1,\ldots,D\}$ such that for each $i \in \{0,1,\ldots,D\}$, $|\{t \in T: \sigma(t) = i\}| \leq m$, and such that, whenever $t \preceq t'$, then $\sigma(t) < \sigma(t')$?

Current state of knowledge concerning PCS sub-problems:

“$m$ arbitrary” means part of the problem instance

*Figure is incorrect in book where $m \leq 3$
For graph problems a common restriction is on the degree of a vertex, i.e., a maximum bound on the number of adjacent vertices for a given vertex.

Many problem become solvable in polynomial time when the maximum degree of a vertex is bounded.

<table>
<thead>
<tr>
<th></th>
<th>In P for $d(v)\leq$</th>
<th>NP-complete for $d(v)\geq$</th>
</tr>
</thead>
<tbody>
<tr>
<td>VERTEX COVER</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>HAMILTONIAN CIRCUIT</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>GRAPH 3-COLORABILITY</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>FEEDBACK VERTEX SET</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>
How about the CLIQUE problem?

For any fixed upper bound on $d(v)$, the CLIQUE problem is solvable in polynomial time.

Observation - If a graph $G$ has $d(v) \leq D$ then the largest clique it can contain is at most $D+1$.

Algorithm:

```plaintext
input: Graph G with $d(v) \leq D$, for all v, and an integer $J \geq 0$;
if ($J \geq D+1$) then
    print("no");
else {
    for (each subset S of $D+1$ vertices or less) {
        if (S forms a clique) {
            print("yes");
            return;
        }
    }
    return "no";
}
```

Since $D$ is fixed, it follows that the above runs in polynomial time.
Recall Graph K-Colorability (for arbitrary K):

**GRAPH COLORABILITY**

INSTANCE: A Graph $G = (V, E)$, positive integer $K \leq |V|$.  
QUESTION: Is $G$ K-colorable, that is, does there exist a function $f: V \rightarrow \{1,2,\ldots,K\}$ such that $f(u) \neq f(v)$ whenever $\{u, v\} \in E$?

Fact: GKC is NP-complete.
Also, Graph $K$-colorability is NP-complete for any fixed $K \geq 3$.

For example – G3C is NP-complete:

**GRAPH 3-COLORABILITY**

**INSTANCE:** A Graph $G = (V, E)$.

**QUESTION:** Is $G$ 3-colorable, that is, does there exist a function $f : V \rightarrow \{1, 2, 3\}$ such that $f(u) \neq f(v)$ whenever $\{u, v\} \in E$?

Fact: G3C is NP-complete.
How about G3C, where \( d(v) \leq 4 \), for all \( v \).

**GRAPH 3-COLORABILITY WITH RESTRICTED DEGREE**

**INSTANCE:** A Graph \( G = (V, E) \), where \( d(v) \leq 4 \), for all \( v \in V \).

**QUESTION:** Is \( G \) 3-colorable, that is, does there exist a function \( f : V \rightarrow \{0, 1, 3\} \) such that \( f(u) \neq f(v) \) whenever \( \{u, v\} \in E \)?

**Theorem 4.1:** GRAPH 3-COLORABILITY with no vertex degree exceeding 4 is NP-complete.

**Proof:**

1) Membership in NP follows immediately from that for the general problem.

2) \( G3C \propto G3CRD \)
Consider the following graph:

Observations:
- $d(v) \leq 4$
- The graph is 3-colorable
- In any 3-coloring of the graph, all “outer” vertices are the same color
Now consider the following graph, which has the same properties:

![Graph Image]

- Observations:
  - \(d(v) \leq 4\)
  - The graph is 3-colorable
  - In any 3-coloring of the graph, all “outer” vertices are the same color

Given a graph \(G\), replace each vertex with \(d(v) \geq 5\) by this subgraph (of an appropriate size).

It follows that the resulting graph \(G'\):
- has \(d(v) \leq 4\)
- is 3-colorable if and only if \(G\) is 3-colorable.
Another restriction for graph problems that frequently results in polynomial time complexity, is by restricting a problem to planar graphs.

A graph is *planar* if it can be embedded in the plane by identifying each vertex with a unique point and each edge with a line (not necessarily straight) connecting its endpoints, so that no two lines meet except at a common endpoint.
CLIQUE is solvable in polynomial time when restricted to planar graphs.

Fact - A planar graph cannot contain a complete subgraph of more than 4 vertices.

Algorithm: (for detecting a clique of size K in a planar graph)

```plaintext
input: Planar graph G and integer K≥0;
if (K ≥ 4) then
    print(“no”);
else {
    for (each subset S of K vertices) {
        if (S forms a K-clique) {
            print(“yes”);
            return;
        }
    }
    return “no”;
}
```

Since $K$ is at most 4, it follows that the above algorithm operates in polynomial $O(n^k)$ time.
Planar Graph 3-Colorability

- In contrast, GRAPH 3-COLORABILITY is NP-complete for planar graphs:

**PLANAR GRAPH 3-COLORABILITY**

INSTANCE: A planar graph $G = (V, E)$.

QUESTION: Is $G$ 3-colorable, that is, does there exist a function $f : V \rightarrow \{1,2,3\}$ such that $f(u) \neq f(v)$ whenever $\{u, v\} \in E$?

- Fact: PG3C is NP-complete.