Reductions

Reduction: Problem A reduces to Problem B if, given a “black box” (subroutine) for B, one can solve A using a (polynomial) number of calls to the subroutine.

Trivial Example:

- B is addition – \( B(x, y) = x + y \)
- A multiplication by 3.
- A reduces to B because we can multiply by 3: \( A(z) = B(z, B(z, z)) \).
Reductions

Reduction: Problem A reduces to Problem B if, given a “black box” (subroutine) for B, one can solve A using a (polynomial) number of calls to the subroutine.

Trivial Example:

• B is addition – \( B(x, y) = x + y \)

• A multiplication by 3.

• A reduces to B because we can multiply by 3: \( A(z) = B(z, B(z, z)) \).

\[
\text{Multiply by 3 (x)}
\]

\[
y = x + x
\]

\[
\frac{z}{2} = y + x
\]

\[
\text{Return z}
\]
More Reduction Examples

- A is max flow, B is linear programming
- A is $1||\sum C_j$, B is $1||\sum w_j C_j$
- A is $P||C_{\text{max}}$, B is $P|\text{prec}|\sum w_j C_j$

Max Flow ($F, \text{cap} u, s, t$)

Write LP

\[
\begin{align*}
\max \sum_{s,t} f_{st} \\
\text{s.t.} \\
f_{ij} &\leq u_{ij} \\
\sum_{j \in v} f_{ij} &\leq f_{ji} \\
f_{ij} &\leq 0
\end{align*}
\]

Call an LP solver

Return the answer
Have code for $\sum_{j=1}^{\infty} w_j c_j$

\[
\begin{array}{c|c}
1 & 2 \\
2 & 6 \\
3 & 1 \\
\end{array}
\min \{ \eta \} \\
\text{If } \forall j \geq 3 \\
C_j = \sum_{j=3}^{\infty} w_j c_j
\]

\[
\text{Solve } l w_j c_j \left( n, w_1, \ldots, w_n \right) \left( p_1, \ldots, p_n \right) \text{ output schedule}
\]

\[
\text{Solve } l c_j \left( n, p_1, \ldots, p_j \right) \\
\text{Return } \text{Solve } l w_j c_j \left( n, 1, \ldots, p_j \right) \left( p_1, \ldots, p_n \right)
\]
P11Cmax reduces to P1|prec|\text{E}_{\text{max}}(\gamma)

\begin{align*}
\begin{array}{l}
J_1 \quad J_2 \\
J_3 \\
\end{array}
\begin{array}{l}
1 \\
2 \\
3 \\
4 \\
5 \\
6 \\
7 \\
8 \\
9 \\
10 \\
\end{array}
\end{align*}

\begin{align*}
m = 2
\begin{array}{c}
1 \\
3 \\
8 \\
16 \\
\end{array}
\begin{array}{l}
J_1 \quad J_2 \quad J_3 \\
J_3 \quad J_4 \\
\end{array}
\end{align*}
\[
P_{\text{llcratch reduces to } P_{\text{prec}}/E_{\text{wj}}(C)}
\]

<table>
<thead>
<tr>
<th>Job</th>
<th>( p_j )</th>
<th>( w_j )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

If there is a job that has to come at end at it has all the wt., then obj are the same.

\[
\sum w_j C_j = w_1 C_1 + tw_2 C_2 + tw_3 C_3 + tw_4 C_4 + tw_5 C_5 + tw_6 C_6
\]

\[
tw_0 C_0 = C_0
\]
schedule minimizing \( P \) will put job 0 as early as possible (objective = \( C_0 \))

\( C_{\text{max}} \) for jobs 1-5 as early as possible

Diagram showing job activities with \( C_{\text{max}} \) = 10.
Summary

- Form input for P|prec|E_{wj} by setting \( w_j = 0 \) and job \( j \) to a new job. 0 and \( p_0 = g_j \).

- Prec constr

- Solve the P|prec|E_{wj} instance

- Return \( C_0 = \text{Cmax of instance} \)
For technical reasons, we will only consider decision versions of problems.

For example, $P||C_{\text{max}}$; given $m$ machines, $n$ jobs and a number $B$, does the optimal schedule have makespan less than $B$.

For example, Shortest Paths: Given a graph $G$ with weights on the edges, two distinguished vertices $s$ and $t$ and a number $B$, is the shortest path from $s$ to $t$ of length less than $B$.

The decision version and the optimization version of a problem are “equivalent,” that is they each reduce to each other.
**Reduction Example**

**Vertex Cover** A vertex cover of a graph $G=(V,E)$ is a set of vertices $V'$, such that for every edge $(x,y)$, at least one of $x$ and $y$ is in $V'$. The vertex cover problem is given a graph $G$ and a number $k$ and asks whether $G$ has a vertex of size at most $k$.

**Clique** A clique is a set of vertices such that each pair of vertices has an edge between them. The clique problem is given a graph and a number $\ell$ and asks when a graph has a clique of size at least $\ell$.

**Question:** Show that vertex cover reduces to clique.
red X form a V.C.

Clique
Given a graph \((G, \mathcal{L})\)
- Form some new graph \(G'\)
- Find some \(\ell\)
- Solve \(\text{Chile}\) \((G', \ell)\)
- Use solution to \(\text{Chile}\) to find a U.C.
- set of vertices not in VC.
  - have no edges between them.

- set of vertices in a clique
  - have all edges between them.
Reduce VC to clique

Input \((G, k)\)

Compute \(G' = \text{complement of } G\).

Set \( l = |V(G)| - k \)

Solve \(\text{Clique}(G', l)\)

output yes/no from \(\text{Clique}(G', l)\)
Is there a V.C. of size 3?

$G$ is not complete.

$G$ is not complete.

No V.C. of size 3.

VC of size 4.

No clique of size 3.

Clique of size 2.
Claim VC reduces to clique