21.4.2
Polynomial-time reductions and hardness

Polynomial-time reductions and hardness

1. For decision problems X and Y, if $X \leq_P Y$, and Y has an efficient algorithm, X has an efficient algorithm.
2. If you believe that **Independent Set** does NOT have an efficient algorithm...
3. Showed: **Independent Set** $\leq_P$ **Clique**
4. $\implies$ **Clique** should not be solvable in polynomial time.
5. If **Clique** had an efficient algorithm, so would **Independent Set**!

Proposition 21.2.

If $X \leq_P Y$ and X does not have an efficient algorithm, Y cannot have an efficient algorithm.

Polynomial-time reductions and hardness

1. For decision problems $\boldsymbol{X}$ and $\boldsymbol{Y}$, if $\boldsymbol{X} \leq_P \boldsymbol{Y}$, and $\boldsymbol{Y}$ has an efficient algorithm, $\boldsymbol{X}$ has an efficient algorithm.
2. If you believe that **Independent Set** does NOT have an efficient algorithm...
3. Showed: **Independent Set** $\leq_P$ **Clique**
4. $\implies$ **Clique** should not be solvable in polynomial time.
5. If **Clique** had an efficient algorithm, so would **Independent Set**!

Proposition 21.2.

If $X \leq_P Y$ and X does not have an efficient algorithm, Y cannot have an efficient algorithm.

Polynomial-time reductions and hardness

1. For decision problems X and Y, if $X \leq_P Y$, and Y has an efficient algorithm, X has an efficient algorithm.
2. If you believe that **Independent Set** does NOT have an efficient algorithm...
3. Showed: **Independent Set** $\leq_P$ **Clique**
4. $\implies$ **Clique** should not be solvable in polynomial time.
5. If **Clique** had an efficient algorithm, so would **Independent Set**!

Proposition 21.2.

If $X \leq_P Y$ and X does not have an efficient algorithm, Y cannot have an efficient algorithm.

Polynomial-time reductions and hardness

1. For decision problems X and Y, if $X \leq_P Y$, and Y has an efficient algorithm, X has an efficient algorithm.
2. If you believe that **Independent Set** does NOT have an efficient algorithm...
3. Showed: **Independent Set** $\leq_P$ **Clique**
4. $\Longrightarrow$ **Clique** should not be solvable in polynomial time.
5. If **Clique** had an efficient algorithm, so would **Independent Set**!

Proposition 21.2.

If $X \leq_P Y$ and X does not have an efficient algorithm, Y cannot have an efficient algorithm.

Polynomial-time reductions and hardness

1. For decision problems $\boldsymbol{X}$ and $\boldsymbol{Y}$, if $\boldsymbol{X} \leq_P \boldsymbol{Y}$, and $\boldsymbol{Y}$ has an efficient algorithm, $\boldsymbol{X}$ has an efficient algorithm.
2. If you believe that **Independent Set** does NOT have an efficient algorithm...
3. Showed: **Independent Set** $\leq_P$ **Clique**
4. $\implies$ **Clique** should not be solvable in polynomial time.
5. If **Clique** had an efficient algorithm, so would **Independent Set**!

Proposition 21.2.

If $X \leq_P Y$ and X does not have an efficient algorithm, Y cannot have an efficient algorithm.

Polynomial-time reductions and hardness

1. For decision problems $\boldsymbol{X}$ and $\boldsymbol{Y}$, if $\boldsymbol{X} \leq_P \boldsymbol{Y}$, and $\boldsymbol{Y}$ has an efficient algorithm, $\boldsymbol{X}$ has an efficient algorithm.
2. If you believe that **Independent Set** does NOT have an efficient algorithm...
3. Showed: **Independent Set** $\leq_P$ **Clique**
4. $\implies$ **Clique** should not be solvable in polynomial time.
5. If **Clique** had an efficient algorithm, so would **Independent Set**!

Proposition 21.2.

If $\boldsymbol{X} \leq_P \boldsymbol{Y}$ and $\boldsymbol{X}$ does not have an efficient algorithm, $\boldsymbol{Y}$ cannot have an efficient algorithm.

Polynomial-time reductions and instance sizes

Proposition 21.3.

Let $\mathcal{R}$ be a polynomial-time reduction from X to Y. Then for any instance I_X of X, the size of the instance I_Y of Y produced from I_X by $\mathcal{R}$ is polynomial in the size of I_X.

Proof.

$\mathcal{R}$ is a polynomial-time algorithm and hence on input I_X of size $|I_X|$ it runs in time $p(|I_X|)$ for some polynomial $p()$.
I_Y is the output of $\mathcal{R}$ on input I_X.
$\mathcal{R}$ can write at most $p(|I_X|)$ bits and hence $|I_Y| \leq p(|I_X|)$. □

Note: Converse is not true. A reduction need not be polynomial-time even if output of reduction is of size polynomial in its input.

Polynomial-time reductions and instance sizes

Proposition 21.3.

Let $\mathcal{R}$ be a polynomial-time reduction from X to Y. Then for any instance I_X of X, the size of the instance I_Y of Y produced from I_X by $\mathcal{R}$ is polynomial in the size of I_X.

Proof.

$\mathcal{R}$ is a polynomial-time algorithm and hence on input I_X of size $|I_X|$ it runs in time $p(|I_X|)$ for some polynomial $p()$.
I_Y is the output of $\mathcal{R}$ on input I_X.
$\mathcal{R}$ can write at most $p(|I_X|)$ bits and hence $|I_Y| \leq p(|I_X|)$. □

Note: Converse is not true. A reduction need not be polynomial-time even if output of reduction is of size polynomial in its input.

Polynomial-time reductions and instance sizes

Proposition 21.3.

Let $\mathcal{R}$ be a polynomial-time reduction from X to Y. Then for any instance I_X of X, the size of the instance I_Y of Y produced from I_X by $\mathcal{R}$ is polynomial in the size of I_X.

Proof.

$\mathcal{R}$ is a polynomial-time algorithm and hence on input I_X of size $|I_X|$ it runs in time $p(|I_X|)$ for some polynomial $p()$.
I_Y is the output of $\mathcal{R}$ on input I_X.
$\mathcal{R}$ can write at most $p(|I_X|)$ bits and hence $|I_Y| \leq p(|I_X|)$. □

Note: Converse is not true. A reduction need not be polynomial-time even if output of reduction is of size polynomial in its input.

Polynomial-time Reduction

Definition 21.4.

A polynomial time reduction from a decision problem X to a decision problem Y is an algorithm $\mathcal{A}$ that has the following properties:

1. Given an instance I_X of X, $\mathcal{A}$ produces an instance I_Y of Y.
2. $\mathcal{A}$ runs in time polynomial in $|I_X|$. This implies that $|I_Y|$ (size of I_Y) is polynomial in $|I_X|$.
3. Answer to I_X YES $\iff$ answer to I_Y is YES.

Proposition 21.5.

If $X \leq_P Y$ then a polynomial time algorithm for Y implies a polynomial time algorithm for X.

Polynomial-time Reduction

Definition 21.4.

A polynomial time reduction from a decision problem X to a decision problem Y is an algorithm $\mathcal{A}$ that has the following properties:

1. Given an instance I_X of X, $\mathcal{A}$ produces an instance I_Y of Y.
2. $\mathcal{A}$ runs in time polynomial in $|I_X|$. This implies that $|I_Y|$ (size of I_Y) is polynomial in $|I_X|$.
3. Answer to I_X YES $\iff$ answer to I_Y is YES.

Proposition 21.5.

If $X \leq_P Y$ then a polynomial time algorithm for Y implies a polynomial time algorithm for X.

Transitivity of Reductions

Proposition 21.6.

$X \leq_P Y$ and $Y \leq_P Z$ implies that $X \leq_P Z$.

Proof.

1. $\mathcal{R}_{X \to Y}$: Polynomial reduction that works in polynomial time $f(x)$.
2. $w \in L_X \iff w' = \mathcal{R}_{X \to Y}(w) \in L_Y$.
3. $\mathcal{R}_{Y \to Z}$: Polynomial reduction that works in polynomial time $g(x)$.
4. $w' \in L_Y \iff w'' = \mathcal{R}_{Y \to Z}(w') \in L_Z$.
5. $w \in L_X \iff w' = \mathcal{R}_{X \to Y}(w) \in L_Y \iff w'' = \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(w)) \in L_Z$.
6. $w \in L_X \iff \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(w)) \in L_Z$.
7. $\mathcal{R}'(x) = \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(x))$ is a reduction from X to Z.
8. Running time of $\mathcal{R}'(x)$ is $h(x) = g(f(x))$, which is a polynomial.

□

Transitivity of Reductions

Proposition 21.6.

$X \leq_P Y$ and $Y \leq_P Z$ implies that $X \leq_P Z$.

Proof.

1. $\mathcal{R}_{X \to Y}$: Polynomial reduction that works in polynomial time $f(x)$.
2. $w \in L_X \iff w' = \mathcal{R}_{X \to Y}(w) \in L_Y$.
3. $\mathcal{R}_{Y \to Z}$: Polynomial reduction that works in polynomial time $g(x)$.
4. $w' \in L_Y \iff w'' = \mathcal{R}_{Y \to Z}(w') \in L_Z$.
5. $w \in L_X \iff w' = \mathcal{R}_{X \to Y}(w) \in L_Y \iff w'' = \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(w)) \in L_Z$.
6. $w \in L_X \iff \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(w)) \in L_Z$.
7. $\mathcal{R}'(x) = \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(x))$ is a reduction from X to Z.
8. Running time of $\mathcal{R}'(x)$ is $h(x) = g(f(x))$, which is a polynomial.

□

Transitivity of Reductions

Proposition 21.6.

$X \leq_P Y$ and $Y \leq_P Z$ implies that $X \leq_P Z$.

Proof.

1. $\mathcal{R}_{X \to Y}$: Polynomial reduction that works in polynomial time $f(x)$.
2. $w \in L_X \iff w' = \mathcal{R}_{X \to Y}(w) \in L_Y$.
3. $\mathcal{R}_{Y \to Z}$: Polynomial reduction that works in polynomial time $g(x)$.
4. $w' \in L_Y \iff w'' = \mathcal{R}_{Y \to Z}(w') \in L_Z$.
5. $w \in L_X \iff w' = \mathcal{R}_{X \to Y}(w) \in L_Y \iff w'' = \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(w)) \in L_Z$.
6. $w \in L_X \iff \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(w)) \in L_Z$.
7. $\mathcal{R}'(x) = \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(x))$ is a reduction from X to Z.
8. Running time of $\mathcal{R}'(x)$ is $h(x) = g(f(x))$, which is a polynomial.

□

Transitivity of Reductions

Proposition 21.6.

$X \leq_P Y$ *and* $Y \leq_P Z$ *implies that* $X \leq_P Z$.

Proof.

1. $\mathcal{R}_{X \to Y}$: Polynomial reduction that works in polynomial time $f(x)$.
2. $w \in L_X \iff w' = \mathcal{R}_{X \to Y}(w) \in L_Y$.
3. $\mathcal{R}_{Y \to Z}$: Polynomial reduction that works in polynomial time $g(x)$.
4. $w' \in L_Y \iff w'' = \mathcal{R}_{Y \to Z}(w') \in L_Z$.
5. $w \in L_X \iff w' = \mathcal{R}_{X \to Y}(w) \in L_Y \iff w'' = \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(w)) \in L_Z$.
6. $w \in L_X \iff \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(w)) \in L_Z$.
7. $\mathcal{R}'(x) = \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(x))$ is a reduction from X to Z.
8. Running time of $\mathcal{R}'(x)$ is $h(x) = g(f(x))$, which is a polynomial.

□

Transitivity of Reductions

Proposition 21.6.

$X \leq_P Y$ and $Y \leq_P Z$ implies that $X \leq_P Z$.

Proof.

1. $\mathcal{R}_{X \to Y}$: Polynomial reduction that works in polynomial time $f(x)$.
2. $w \in L_X \iff w' = \mathcal{R}_{X \to Y}(w) \in L_Y$.
3. $\mathcal{R}_{Y \to Z}$: Polynomial reduction that works in polynomial time $g(x)$.
4. $w' \in L_Y \iff w'' = \mathcal{R}_{Y \to Z}(w') \in L_Z$.
5. $w \in L_X \iff w' = \mathcal{R}_{X \to Y}(w) \in L_Y \iff w'' = \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(w)) \in L_Z$.
6. $w \in L_X \iff \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(w)) \in L_Z$.
7. $\mathcal{R}'(x) = \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(x))$ is a reduction from X to Z.
8. Running time of $\mathcal{R}'(x)$ is $h(x) = g(f(x))$, which is a polynomial.

□

Transitivity of Reductions

Proposition 21.6.

$X \leq_P Y$ and $Y \leq_P Z$ implies that $X \leq_P Z$.

Proof.

1. $\mathcal{R}_{X \to Y}$: Polynomial reduction that works in polynomial time $f(x)$.
2. $w \in L_X \iff w' = \mathcal{R}_{X \to Y}(w) \in L_Y$.
3. $\mathcal{R}_{Y \to Z}$: Polynomial reduction that works in polynomial time $g(x)$.
4. $w' \in L_Y \iff w'' = \mathcal{R}_{Y \to Z}(w') \in L_Z$.
5. $w \in L_X \iff w' = \mathcal{R}_{X \to Y}(w) \in L_Y \iff w'' = \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(w)) \in L_Z$.
6. $w \in L_X \iff \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(w)) \in L_Z$.
7. $\mathcal{R}'(x) = \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(x))$ is a reduction from X to Z.
8. Running time of $\mathcal{R}'(x)$ is $h(x) = g(f(x))$, which is a polynomial.

□

Transitivity of Reductions

Proposition 21.6.

$X \leq_P Y$ and $Y \leq_P Z$ implies that $X \leq_P Z$.

Proof.

1. $\mathcal{R}_{X \to Y}$: Polynomial reduction that works in polynomial time $f(x)$.
2. $w \in L_X \iff w' = \mathcal{R}_{X \to Y}(w) \in L_Y$.
3. $\mathcal{R}_{Y \to Z}$: Polynomial reduction that works in polynomial time $g(x)$.
4. $w' \in L_Y \iff w'' = \mathcal{R}_{Y \to Z}(w') \in L_Z$.
5. $w \in L_X \iff w' = \mathcal{R}_{X \to Y}(w) \in L_Y \iff w'' = \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(w)) \in L_Z$.
6. $w \in L_X \iff \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(w)) \in L_Z$.
7. $\mathcal{R}'(x) = \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(x))$ is a reduction from X to Z.
8. Running time of $\mathcal{R}'(x)$ is $h(x) = g(f(x))$, which is a polynomial.

□

Transitivity of Reductions

Proposition 21.6.

$X \leq_P Y$ and $Y \leq_P Z$ implies that $X \leq_P Z$.

Proof.

1. $\mathcal{R}_{X \to Y}$: Polynomial reduction that works in polynomial time $f(x)$.
2. $w \in L_X \iff w' = \mathcal{R}_{X \to Y}(w) \in L_Y$.
3. $\mathcal{R}_{Y \to Z}$: Polynomial reduction that works in polynomial time $g(x)$.
4. $w' \in L_Y \iff w'' = \mathcal{R}_{Y \to Z}(w') \in L_Z$.
5. $w \in L_X \iff w' = \mathcal{R}_{X \to Y}(w) \in L_Y \iff w'' = \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(w)) \in L_Z$.
6. $w \in L_X \iff \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(w)) \in L_Z$.
7. $\mathcal{R}'(x) = \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(x))$ is a reduction from X to Z.
8. Running time of $\mathcal{R}'(x)$ is $h(x) = g(f(x))$, which is a polynomial.

Transitivity of Reductions

Proposition 21.6.

$X \leq_P Y$ *and* $Y \leq_P Z$ *implies that* $X \leq_P Z$.

Proof.

1. $\mathcal{R}_{X \to Y}$: Polynomial reduction that works in polynomial time $f(x)$.
2. $w \in L_X \iff w' = \mathcal{R}_{X \to Y}(w) \in L_Y$.
3. $\mathcal{R}_{Y \to Z}$: Polynomial reduction that works in polynomial time $g(x)$.
4. $w' \in L_Y \iff w'' = \mathcal{R}_{Y \to Z}(w') \in L_Z$.
5. $w \in L_X \iff w' = \mathcal{R}_{X \to Y}(w) \in L_Y \iff w'' = \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(w)) \in L_Z$.
6. $w \in L_X \iff \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(w)) \in L_Z$.
7. $\mathcal{R}'(x) = \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(x))$ is a reduction from X to Z.
8. Running time of $\mathcal{R}'(x)$ is $h(x) = g(f(x))$, which is a polynomial.

□

Transitivity of Reductions

Proposition 21.6.

$X \le_P Y$ and $Y \le_P Z$ implies that $X \le_P Z$.

Proof.

1. $\mathcal{R}_{X \to Y}$: Polynomial reduction that works in polynomial time $f(x)$.
2. $w \in L_X \iff w' = \mathcal{R}_{X \to Y}(w) \in L_Y$.
3. $\mathcal{R}_{Y \to Z}$: Polynomial reduction that works in polynomial time $g(x)$.
4. $w' \in L_Y \iff w'' = \mathcal{R}_{Y \to Z}(w') \in L_Z$.
5. $w \in L_X \iff w' = \mathcal{R}_{X \to Y}(w) \in L_Y \iff w'' = \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(w)) \in L_Z$.
6. $w \in L_X \iff \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(w)) \in L_Z$.
7. $\mathcal{R}'(x) = \mathcal{R}_{Y \to Z}(\mathcal{R}_{X \to Y}(x))$ is a reduction from X to Z.
8. Running time of $\mathcal{R}'(x)$ is $h(x) = g(f(x))$, which is a polynomial.

□

Be careful about reduction direction

Note: $\boldsymbol{X} \leq_P \boldsymbol{Y}$ does not imply that $\boldsymbol{Y} \leq_P \boldsymbol{X}$ and hence it is very important to know the FROM and TO in a reduction.

To prove $\boldsymbol{X} \leq_P \boldsymbol{Y}$ you need to show a reduction FROM $\boldsymbol{X}$ TO $\boldsymbol{Y}$
That is, show that an algorithm for $\boldsymbol{Y}$ implies an algorithm for $\boldsymbol{X}$.

THE END

…

(for now)