Monash University
Faculty of Information Technology

FIT2014 Theory of Computation

Lecture 26
Polynomial-time reductions

slides by Graham Farr

Overview

- Definition of polynomial time reduction
- Examples
- Properties

Polynomial-time reductions

Definition
A **polynomial-time reduction** from K to L is
a *polynomial-time mapping reduction* from K to L.

So, it's a *polynomial-time* computable function

$$f : \Sigma^* \to \Sigma^*$$

such that, for all $x \in \Sigma^*$,

$$x \in K \quad \text{if and only if} \quad f(x) \in L$$

Polynomial-time reductions

Polynomial-time reductions are also called:

- polynomial-time mapping reductions
- polynomial-time many-one reductions
- polynomial transformations
- Karp reductions

If there is a polynomial-time reduction from K to L, then we write $K \leq_P L$.

Examples

One place we can look for examples:

- mapping reductions!

Which of the **mapping reductions** in Lecture 21 are **polynomial-time**?

	Yes	No
EQUAL $\longrightarrow$ HALF-AND-HALF	☐	☐
HALF-AND-HALF $\longrightarrow$ PARENTHESES	☐	☐
FA-Empty $\longrightarrow$ No-Digraph-Path	☐	☐
RegExpEquiv $\longrightarrow$ FA-Empty	☐	☐

Examples

INDEPENDENT SET $\leq_P$ CLIQUE

The **complement** $\overline{G}$ of G: edges $\longleftrightarrow$ non-edges

Independent sets in G correspond to cliques in $\overline{G}$.

G has an independent set of size $\geq k$ if and only if $\overline{G}$ has a clique of size $\geq k$.
So:

$$(G, k) \in \text{INDEPENDENT SET} \quad \text{if and only if} \quad (\overline{G}, k) \in \text{CLIQUE}.$$

Construction of $(\overline{G}, k)$ from (G, k) is polynomial time.
So the function

$$(G, k) \mapsto (\overline{G}, k)$$

is a polynomial-time reduction from INDEPENDENT SET to CLIQUE.

Examples

VERTEX COVER $\leq_P$ INDEPENDENT SET

If G is a graph and $X \subseteq V(G)$, then:

X is a vertex cover of G if and only if $V(G) \setminus X$ is an independent set of G.

So:

$(G, k) \in$ VERTEX COVER if and only if $(G, n-k) \in$ INDEPENDENT SET.

The construction is polynomial time.

So the function

$$(G, k) \mapsto (G, n-k)$$

is a polynomial-time reduction from VERTEX COVER to INDEPENDENT SET.

Examples

2-SAT $\leq_P$ 3-SAT

Given a Boolean formula φ in CNF with 2 literals per clause,
we want to transform it to another Boolean formula φ' in CNF with 3 literals/clause,
such that

$$\varphi \text{ is satisfiable} \quad \text{if and only if} \quad \varphi' \text{ is satisfiable.}$$

For each i:

Suppose i-th clause in φ is $x \vee y$.

Create a new variable w_i which appears nowhere else.

Replace clause $x \vee y$ by two clauses:

$$(x \vee y \vee w_i) \wedge (x \vee y \vee \neg w_i)$$

Examples

Then show that:

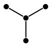
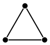
- this construction takes polynomial time
- φ is satisfiable if and only if φ' is satisfiable.

Examples

SUBGRAPH ISOMORPHISM $:= \{(G, H) : G$ is isomorphic to a *subgraph* of $H\}$.

GRAPH ISOMORPHISM $\leq_P$ SUBGRAPH ISOMORPHISM

$$(G, H) \mapsto \begin{cases} (G, H) & \text{if } |V(G)| \geq |V(H)|, \\ \left(\quad , \quad \right) & \text{if } |V(G)| < |V(H)|. \end{cases}$$

Polynomial time!

Does it work the other way round?

PARTITION

$$\left\{ (s_1, s_2, \ldots, s_n) \ : \ \text{for some } J \subseteq \{1, 2, \ldots, n\}, \ \sum_{i \in J} s_i = \sum_{i \in \{1,\ldots,n\} \setminus J} s_i \right\}$$

SUBSET SUM

$$\left\{ (s_1, s_2, \ldots, s_n, t) \ : \ \text{for some } J \subseteq \{1, 2, \ldots, n\}, \ \sum_{i \in J} s_i = t \right\}$$

$$\text{PARTITION} \quad \leq_P \quad \text{SUBSET SUM}$$

$$(s_1, s_2, \cdots, s_n) \quad \mapsto \quad (\, s_1, s_2, \ldots, s_n, (s_1 + s_2 + \cdots + s_n)/2 \,)$$

Can you show SUBSET SUM $\leq_P$ PARTITION ?

Others to try:

$$\text{3-COLOURABILITY} \leq_P \text{GRAPH COLOURING}$$

where $\quad$ GRAPH COLOURING $:= \{ (G, k) : G \text{ is } k\text{-colourable} \}$

$$\text{2-COLOURABILITY} \leq_P \text{3-COLOURABILITY}$$

$$\text{HAMILTONIAN CIRCUIT} \leq_P \text{HAMILTONIAN PATH}$$

$$\text{2-COLOURABILITY} \leq_P \text{2-SAT}$$

$$\text{SATISFIABILITY} \leq_P \text{3-SAT}$$

$$\text{3-COLOURABILITY} \leq_P \text{SATISFIABILITY}$$

Properties

Reflexive: For any L, $L \leq_P L$.

Transitive: If $K \leq_P L$ and $L \leq_P M$ then $K \leq_P M$.

Properties

$$\begin{tikzcd} x \arrow[d, "f"] \\ y = f(x) \arrow[d, "g"] \\ g(f(x)) \end{tikzcd}$$

Theorem.
If $K \leq_P L$ and $L \leq_P M$ then $K \leq_P M$.

Proof.
Let f be a polynomial-time reduction from K to L.
Let g be a polynomial-time reduction from L to M.

We've already seen (Lecture 21) that $g \circ f$ is a *mapping reduction* from K to M.
We just need to show that it's a *polynomial-time* mapping reduction.

Since f and g are both polynomial-time, we know that:
(i) $f(x)$ is computable in time $\leq c\,|x|^k$, for some constants c, k and all sufficiently large x.
(ii) $g(y)$ is computable in time $\leq d\,|y|^\ell$, for some constants d, ℓ and all sufficiently large y.

It follows from (i) that $|f(x)| \leq c\,|x|^k$ too, for sufficently large x,
since at most one letter of output can be computed in each time-step.

It follows from (ii) that

$$\begin{aligned} \text{time to compute } g(f(x)) \text{ from } f(x) \text{ is} \quad & \leq \; d\,|f(x)|^{\ell} \\ & = \; d\,(c|x|^k)^{\ell} \qquad \text{by above bound on } |f(x)| \\ & = \; d\,c^{\ell}\,|x|^{k\ell} \qquad \text{for large enough } x. \end{aligned}$$

Therefore,

$$\begin{aligned} & \text{time to compute } g(f(x)) \\ & \quad = \; \text{time to compute } f(x) \text{ from } x \; + \; \text{time to compute } g(f(x)) \text{ from } f(x) \\ & \quad \leq \; c\,|x|^k \; + \; d\,c^{\ell}\,|x|^{k\ell} \qquad \text{for sufficiently large } x\text{, using what we did above} \\ & \quad \leq \; c'\,|x|^m \qquad \text{for some constants } c', m \text{ and all sufficiently large } x. \end{aligned}$$

So $g \circ f$ is polynomial-time. $\square$

Properties

Theorem. If $K \leq_P L$ and L is in P, then K is in P.

Proof.
Let f be a polynomial-time reduction from K to L,
and let D be a poly-time decider for L.

Decider for K: *(same as in Lecture 21)*

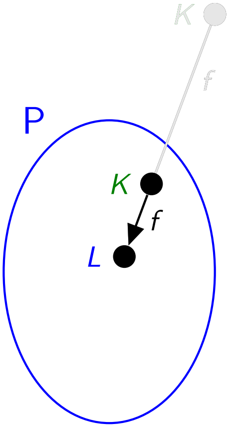
Input: x.

Compute $f(x)$.

Run the Decider for L on $f(x)$.

// *This L-Decider accepts $f(x)$ if and only if $x \in K$, since f is a mapping reduction from K to L.*

We also need to show it's polynomial time.



Properties

If f has time complexity $O(n^k)$, then the length of its output string $f(x)$
must also be $O(n^k)$,
since a TM can, in t steps, output no more than t symbols.

The decider D runs in polynomial time, so suppose it has time complexity $O(n^{k'})$,
where n is the size of the input to D.

If D is given $f(x)$ as input, then the time D takes on it is $O(|f(x)|^{k'})$,
where $|f(x)| =$ length of string $f(x)$.

Since $|f(x)| = O(n^k)$, we find that D takes time $O(n^{kk'})$, where $n = |x|$.

Properties

Total time taken by our decider for K is:

$$\begin{aligned}\text{time taken by } f \text{ on } x \ + \ \text{time taken by } D \text{ on } f(x) &= O(n^k) + O(n^{kk'}) \\ &= O(n^{kk'}),\end{aligned}$$

which is polynomial time. $\square$

Properties

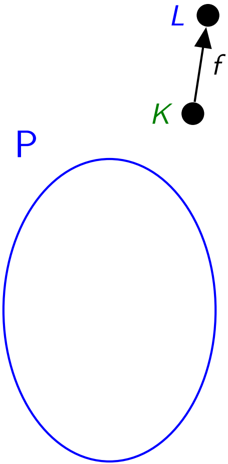
Corollary
If there is a polynomial-time reduction f from K to L, then:

If K is <u>not</u> in P, then L is <u>not</u> in P.

Symbolically:

$$(K \leq_P L) \wedge (K \notin \mathrm{P}) \implies (L \notin \mathrm{P})$$

Proof.
Contrapositive of previous Theorem. $\square$



Exercises

Prove:

If K is in P and L is any language, then $K \leq_P L$.

The fine print: some caveats regarding trivial cases are needed here. What are they?

Prove:

Theorem.
If $K \leq_P L$ and L is in NP, then K is in NP.

Revision

Things to think about:

- You will have seen transformations from one problem to another before, and probably not just in this unit.
  Are any of them polynomial-time reductions?
- Find some of the polynomial-time reductions mentioned on Slide 12.

Reading:

- Sipser, Section 7.4, pp. 299–303.
- M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman & Co., San Francisco, 1979: §2.5.