Reductions

A function $f : \Sigma^* \to \Sigma^*$ is *polynomial-time computable* (p.-t.c.) if there ist a p.b. deterministic Turing machine comuting f.

Let $L, R \subseteq \Sigma^*$.
L is *polynomial-time reducible* (p.-t. red.) to R if there is a p.-t.c. *reduction* $r : \Sigma^* \to \Sigma^*$ s.t., for any $w \in \Sigma^* : w \in L \iff r(w) \in R$.

NEW (in contrast with Ch.5): polynomial time bounds!

Thm. 1 p.-t. reducibility is a transitive relation.

Applying p.-t. reductions I

Suppose we have to design a p.-t. algorithm for L.
Assume L is p.-t. red. to R.
Assume there is a p.-t. algorithm B for deciding R in p.-t.

Then, $L \in \mathcal{P}$ by the following algorithm A:

1. Given input w, compute $r(w)$.

2. Use B to compute "$r(w) \in R$".

3. Return B's answer.

Hence, $L \in \mathcal{P}$.

More precisely, if p bounds the running time for computing r, then $|r(w)| \leq p(|w|)$. If q bounds the running time of B, then $\approx q(p(|w|)$ bounds the running time of A.

We have used r and B as subroutines / auxiliary procedures for designing A.

Applying p.-t. reductions II

Suppose we *believe/know* that there is no p.-t. algorithm for L.

Assume L is p.-t. red. to R.

Then, we can conclude that there is *no* p.-t. algorithm for R.

Assume the contrary: Assume there is a p.-t. algorithm B for deciding R in p.-t.

Then, the preceding argument shows that there is a p.-t. algorithm for L, contradicting our belief/knowledge.

A simple reduction

Thm. 2 HC is p.-t. red. to TS.

TS $= \{\langle G\rangle \,\$\, \langle W\rangle \,\$\, \langle B\rangle \mid$ there is a tour in G with total weight not exceeding $B\}$

HC $= \{\langle G\rangle \mid G$ has a Hamitonian cycle $\}$

Proof. TASK: Design a reduction function r mapping a HC instance $\langle G\rangle$ onto a TS instance $\langle G'\rangle \,\$\, \langle W\rangle \,\$\, \langle B\rangle$.

Given $G = (V, E)$, compute TS instance as follows:

1. $G' = (V, E')$ is a complete graph.
2. $W = (w_{ij})$ is defined as:

$$w_{ij} = \begin{cases} 0 & \text{if } (v_i, v_j) \in E \\ 1 & \text{if } (v_i, v_j) \notin E \end{cases}$$

3. $B = 0$.

If G has a tour, then this tour has weight 0 in G'. If G' has a tour of weight 0, then this tour contains only edges from G, so that it is a tour in G. □

Cor.: If HC were "difficult," then TS would be so, as well.

The directed Hamilt. Circuit Problem

HCD $= \{\langle G \rangle \mid G$ has a Hamitonian cycle $\}$

Given a *directed* graph G, does G have a directed tour (which is called *Hamiltonian cycle* in this case)?

Lem. 3 HCD $\in \mathcal{NP}$

Thm. 4 HCD is p.-t. red. to HC.

Proof. Given directed graph $G = (V, E)$, compute undirected graph $G' = (V', E')$:

For each vertex $v \in V$, there are vertices v_0, v_1 and v_2 in V'. The following edges are in E':

1. $\{v_0, v_1\}$ for each $v \in V$
2. $\{v_1, v_2\}$ for each $v \in V$
3. $\{v_2, w_0\}$ for each $(v, w) \in E$

If $v^1, \ldots, v^n, v^1$ is a HC in G, then

$$v_0^1, v_1^1, v_2^1, \ldots, v_0^n, v_1^n, v_2^n, v_0^1$$

is a HC in G'. If h is a HC in G', then w.l.o.g. the vertices v_0, v_1, v_2 are traversed in this order for each $v \in V$, since v_1 has degree two. Hence, there is a HC in G "corresponding" to h. □

Satisfiability and Variants

A *Boolean expression* (B.E.) is an expression composed of (Boolean) variables, parentheses, and the logical operators $\wedge$, $\vee$ and $\neg$.

Given an assignment of 0 (FALSE) and 1 (TRUE) to all the variables, a B.E. can be recursively evaluated as follows:

1. The value of a variable is according to the assignment.

2. If E_1 and E_2 are B.E.s, then $E_1 \wedge E_2$ is a B.E. evaluating to 1 iff both E_1 and E_2 evaluate to 1.

3. If E_1 and E_2 are B.E., then $E_1 \vee E_2$ is a B.E. evaluating to 1 iff E_1 or E_2 (or both) evaluate to 1.

4. If E is a B.E., then $\neg E$ is a B.E. evaluating to 1 iff E evaluates to 0.

Rem.: The syntactically correct B.E.s over a given set of Variables can be easily described by a CFG.

The language SAT

A B.E. E is *satisfiable* if there is an assignment to its variables such that (then) E evaluates to 1.

W.l.o.g., let all Boolean variables be from the (infinite) set $\{x_1, x_2, x_3, \ldots\}$

Write x_j (in the codification of some B.E. E) as $x\beta(r_j)$, where $\beta(m)$ denotes some binary codification of the number m, and r_j indicates that x_j is the r_j's different variable appearing in E (from left to right).

Then,
SAT $= \{\langle E\rangle \mid E$ is satisfiable $\}$
is a language over $\{\wedge, \vee, \neg, (,), x, 0, 1\}$.

Lem. 5 If E is a B.E. of "length" n, then $|\langle E\rangle| = O(n \log n)$.

For p.-t. computations, it is hence insignificant whether referring to the length of E or of $\langle E\rangle$.

By "guessing" a satisfying assignment:

Lem. 6 SAT $\in \mathcal{NP}$.

LIT-SAT

Let us call a B.E. to be in *literal normal form* (literal NF) if any negation may only directly refer to a variable occurrence.

Ex.: $\neg(\neg(x_1 \vee x_2) \wedge x_2)$ is not in literal NF.

LIT-SAT $= \{\langle E \rangle \mid E$ is a B.E. in literal NF which is satisfiable $\}$

Thm. 7 LIT-SAT is p.-t. red. to SAT.

Proof. Apply de Morgan's laws "exhaustively". □

Ex.: $\neg(\neg(x_1 \vee x_2) \wedge x_2)$
$= \neg\neg(x_1 \vee x_2) \vee \neg x_2$
$= x_1 \vee x_2 \vee \neg x_2$.

Rem: What about p.-t.??

A B.E. is said to be in **conjunctive normal form** (CNF) if it is of the form

$$C_1 \wedge C_2 \wedge \ldots \wedge C_k$$

and each C_j (called a *clause*) is of the form

$$C_j = \alpha_{j1} \vee \alpha_{j2} \vee \ldots \vee \alpha_{jr_j},$$

where each α_{ij} is a *literal*, i.e., either x or $\neg x$ for some variable x.
We also write $\bar{x}$ instead of $\neg x$.

Ex.: $(x_1 \vee x_2 \vee x_3) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee x_3)$ is in CNF.

CSAT $= \{\langle E \rangle \mid E$ is a B.E. in CNF which is satisfiable $\}$

Thm. 8 CSAT is p.-t. red. to LIT-SAT.

Rem.: Kinber/Smith only consider CSAT.

CSAT $\rightarrow$ LIT-SAT

Let V_1 and V_2 be sets of variables with $V_1 \subseteq V_2$. An assignment α_2 of V_2 *extends* an assignment α_1 of V_1 if $\alpha_1(x) = \alpha_2(x)$ for all $x \in V_1$.

By induction on the number r of occurrences of $\wedge$ and $\vee$, we shall prove:

CLAIM: Any B.E. E in literal NF can be transformed into a B.E. E' in CNF consisting of at most n clauses over a set of variables V' that include the variables V of E and at most n other variables s.t. E is satisfiable iff E' is satisfiable.
Moreover, if an assignment α of V satisfies E, then there is an extension α' of α that evaluates to 1 on E'.
Conversely, if α' is an assignment to V' that evaluates to 1, then its restriction to V satisfies E.

Proof of CLAIM:

Anchor: $r = 0$ ✓

Induction step: We distinguish two cases:

1. $E = E_1 \wedge E_2$. By I.H., E_1 can be transformed into E_1' and E_2 into E_2'. W.l.o.g., assume that no variable that is not present in E appears both in E_1' and in E_2'. Then, $E' := E_1' \wedge E_2'$ satisfies the requirements.

2. $E = E_1 \vee E_2$. By I.H., E_1 can be transformed into E_1' and E_2 into E_2'. Let

   $$E_j' = C_{j,1} \wedge \ldots \wedge C_{j,r_j}$$

   Let y be a new variable and set $E' := \tilde{E}_1 \wedge \tilde{E}_2$, where

   $$\tilde{E}_j = \tilde{C}_{j,1} \wedge \ldots \wedge \tilde{C}_{j,r_j}$$

   with $\tilde{C}_{1,i} := y \vee C_{1,i}$ and $\tilde{C}_{2,i} := \bar{y} \vee C_{2,i}$.

Transitivity: SAT $\rightarrow$ CSAT

Ex.: Consider

$$\neg(\neg(x_1 \vee x_2) \wedge (\neg x_1 \vee x_2))$$

De Morgan into LIT-SAT:

$$E = \underbrace{(x_1 \vee x_2)}_{=:E_1} \vee \underbrace{(x_1 \wedge \bar{x}_3)}_{=:E_2}$$

E_1 is then transformed into

$$E_1' = (x_1 \vee y_1) \wedge (x_2 \vee \bar{y}_1),$$

while $E_2' = E_2$.

Hence, $E' = (x_1 \vee y_1 \vee y_2) \wedge (x_2 \vee \bar{y}_1 \vee y_2) \wedge (x_1 \vee \bar{y}_2) \wedge (\bar{x}_3 \vee \bar{y}_2)$

$x_1 = 0$, $x_2 = 1$ and $x_3 = 1$ is an assignment α satisfying E.
By setting $y_1 = 1$ and $y_2 = 0$, we can extend α as to satisfy E'.

3-SAT as defined in K/S.

A B.E. in CNF is in 3NF if each clause contains at most three variables or negated variables. $\rightsquigarrow$ 3-SAT

Thm. 9 3-SAT is p.-t. red. to CSAT.

Proof. Consider the following procedure:
WHILE E contains a "large" clause with more than 3 variables DO
Pick some "large" clause $C = \ell_1 \vee \ldots \vee \ell_k$.
Replace C in E by

1. $C_{short} = \ell_1 \vee \ell_2 \vee y$ and

2. $C_{tail} = \bar{y} \vee \ell_3 \vee \ldots \vee \ell_k$

(where y is a "new" variable)
OD
Observe: C_{tail} is shorter than C!
Formally, again "extension argument". □

Ex-3-SAT as meant by K/S.

A B.E. in CNF is in exact 3NF if each clause contains exactly three variables or negated variables. $\rightsquigarrow$ Ex-3-SAT

Thm. 10 Ex-3-SAT is p.-t. red. to 3-SAT.

Proof. Consider the following procedure:
WHILE E contains a "small" clause with less than 3 variables DO
Pick some "small" clause C.
Replace C in E by

1. $C_1 = C \vee y$ and

2. $C_2 = C \vee \bar{y}$

(where y is a "new" variable)
OD
Observe: C_1, C_2 are longer than C!
Formally, again "extension argument". $\square$

CLIQUE revisited

$C = \{\langle G \rangle \, \$ \, \langle k \rangle \mid G \text{ has a clique of size } k\}$

Thm. 11 C is p.-t. red. to Ex-3-SAT.

Proof. Let $E = C_1 \wedge \ldots \wedge C_n$ be a B.E. in exact 3NF. Construct a graph G s.t. E is satisfiable iff G has a clique of size n.

For each clause $C_r = (\ell_1^r \vee \ell_2^r \vee \ell_3^r)$, create three vertices v_1^r, v_2^r and v_3^r in G. Connect v_i^r and v_j^s by an edge iff

1. $r \neq s$ and

2. ℓ_i^r is not the negation of ℓ_j^s.

The reduction can be computed in p.t. $\square$

The correctness of the construction:

Let α be a satisfying assignment of E. Each clause must contain some literal ℓ_i^r evaluating to 1. Pick the corresponding vertex v_i^r. The vertices picked this way form a clique of size n.

Let G' be a clique in G of size n. By the first condition, no two vertices of G' "belong" to the same clause. Making the literals "true" which correspond to clique-vertices (this yields no contradiction by the second condition) gives a (partial) assignment which can be extended arbitrarily to give a satisfying assignment of E.