Algorithm Theory

Georg Moser

Institute of Computer Science @ UIBK

Summer 2007

Content

~~W 1~~ ~~Introduction, Problems and Algorithms~~
~~W 2~~ ~~Turing machines as algorithms, multiple-string TMs~~
~~W 3~~ ~~Random access machines, nondeterministic machines~~
~~W 4~~ ~~Complexity classes~~
~~W 5~~ ~~The Hierarchy Theorems~~
~~W 6~~ ~~Reachability Method~~
~~W 7~~ ~~Savitch's Theorem~~
~~W 8~~ ~~Reductions, completeness, Cook's Theorem~~
~~W 9~~ ~~NP-complete problems, Variants of SAT~~
~~W 10~~ ~~Graph-theoretic Problems~~
~~W 11~~ ~~Hamilton Path~~
~~W 12~~ ~~Sets and Numbers~~
~~W 13~~ ~~**coNP** & Primality~~
W 14 Function Problems

Function Problems: FSAT

Definition — FSAT

FSAT is the problem:

1. given a formula φ in conjunctive normal form (CNF)
2. find a satisfying assignment for φ

Theorem

given a polynomial algorithm for SAT, we can define a polynomial algorithm for FSAT

Proof

call the hypothetical program that decides SAT H
assume φ contains the variables $\{x_1, \dots, x_n\}$

Idea

- apply H repeatedly
- test suitable variable assignments at each stage

Algorithm

1. call H on φ
   if H fails: stop
2. divide φ into $\varphi_1 = \varphi[x_1 = \mathbf{true}]$ and $\varphi_2 = \varphi[x_1 = \mathbf{false}]$
3. use H to decide whether φ_1 or φ_2 is satisfiable
4. fix the partial assignment T accordingly and repeat
5. if all variables are exhausted: stop

□

Definition — polynomially balanced

- a relation R is called polynomially decidable
  if $\exists$ DTM deciding $\{x; y \colon (x, y) \in R\}$
- R is called polynomially balanced
  if $(x, y) \in R$ implies $|y| \leqslant |x|^k$ for some $k \geqslant 1$

Characterising **NP**

let $\mathrm{L} \subseteq \Sigma^*$ be a language

Theorem

$\mathrm{L} \in$ **NP** if and only if
$\exists$ polynomially decidable and polynomially balanced relation R
with $\mathrm{L} = \{x \colon (x, y) \in R\}$

Proof Sketch

$\Rightarrow$ any relation decided by a polynomial verifier is balanced, as the verifier can read at most polynomially many letters (in $|x|$) of the certificate

$\Leftarrow$ employ the DTM as polynomial verifier $\square$

Function Problems: In relation to **NP**

given

- let $\mathrm{L} \in$ **NP**
- assume R_{L} is a polynomially decidable and balanced relation

Definition — function problems

- the function problem $F\,\mathrm{L}$ associated with L is the problem:
  1. given x
  2. if $\exists\, y$ with $R_{\mathrm{L}}(x, y)$ find y otherwise return *no*
- note, the function problem is described by the relation

Example

FSAT is the function problem associated with SAT
use the polynomially balanced relation for SAT

Definition

- **FNP** $= \{F\,\mathrm{L} \mid \mathrm{L} \in$ **NP**$\}$
- **FP** $\subseteq$ **FNP** such that we only consider problems in **FNP** solvable in polytime

Example

FSAT $\in$ **FNP**

Definition — logspace reduction

A reduces to B, if

- $\exists$ functions $R \colon \Sigma^* \to \Sigma^*$ and $S \colon \Sigma^* \to \Sigma^*$
  R, S logspace computable
- if x an instance of A
  then $R(x)$ an instance of B
- if z is a correct output of B on $R(x)$
  then $S(z)$ is a correct output of A on x

The **FP** $\neq$ **FNP** problem

Definition — **FNP**-completeness

a function problem A is complete for **FNP**
if $A \in$ **FNP**, and all problems in **FNP** reduce (in logspace) to A

Theorem

FSAT is **FNP**-complete

Theorem

FP $=$ **FNP** iff **P** $=$ **NP**.

Proof Sketch

$\Rightarrow$ by definition, i.e., the function problem is "stronger"

$\Leftarrow$ assume **NP** $=$ **P**, in particular SAT $\in$ **P**
as FSAT is **FNP**-complete, we only need to show FSAT $\in$ **FP**
however SAT $\in$ **P** implies FSAT $\in$ **FP** $\square$

Total Functions

Definition

total function

- a problem R in **FNP** is total if
  $\forall$ strings x
  $\exists$ at least one string y such that $R(x, y)$
- this subclass is denoted as **TFNP**

Remark

total function problems correspond to the language $\mathrm{L} = \Sigma^*$
hence the corresponding decision problem is meaningless

Example

FACTORING

1. given an integer n
2. find its prime decomposition $n = {p_1}^{k_1} \cdots {p_m}^{k_m}$
   together with primality certificates for $p_1, \dots, p_m$