

Computational Complexity Theory – Course Notes

CCT Team

20. september 2006

1 Lecture 15/9-2006

1.1 Complete problems

One reason for caring about complexity classes is complete problems. Below are examples of complete problems in different classes:

L	All problems except the empty and the full.
NL=coNL	GAP, 2SAT.
P	Circuit Value.
NP	SAT, Clique, Circuit SAT.
NEXP	Succint Circuit SAT, Succint SAT, Succint Clique.

Circuit Value is the problem given a circuit with only constants as sources/input and only one output gate what is the output?

1.2 The tableau method

We have several versions of the tableau method:

Theorem 1 *Let a deterministic TM, M , running in time $p(n)$ deciding L be given. Then $\exists C_1, C_2, \dots : C_i : \{0, 1\}^n \rightarrow \{0, 1\}$.*

- $|c_i| \leq q(i)$
- $\forall x \in \{0, 1\}^i : x \in L \Leftrightarrow C_i(x) = 1$
- \exists poly time TM M' that given n produces C_n .

Theorem 2 *Let a deterministic TM, M , running in time $p(n)$ deciding L be given. Then $\exists C_1, C_2, \dots : C_i : \{0, 1\}^n \rightarrow \{0, 1\}$.*

- $|c_i| \leq q(i)$
- $\forall x \in \{0, 1\}^i : x \in L \Leftrightarrow C_i(x) = 1$

- \exists log space TM M' that given n produces C_n .

Theorem 3 Let a deterministic TM, M , running in time $2^{p(n)}$ deciding L be given. Then $\exists C_1, C_2, \dots : C_i : \{0, 1\}^n \rightarrow \{0, 1\}$.

- $|c_i| \leq q(i)$
- $\forall x \in \{0, 1\}^i : x \in L \Leftrightarrow C_i(x) = 1$
- \exists log space TM M' that given n produces C'_n , where C'_n is a succinct (to be defined) description of C_n .

Definition 1 Given a circuit C (defining a bit) we say that C' (defining a multi output function) is a succinct description of C if

$$c'(i) = (j, k, t)$$

where i is a binary index of a gate in C , j and k are the indices of the input of gate i (we let the constants 0 and 1 be gates at index 0 and 1), and t is taken from the set $\{\wedge, \vee, \neg, \overset{\wedge}{out}, \overset{\vee}{out}, \overset{\neg}{out}\}$.

1.3 Alternating Turing Machine

An ATM starts out as a single TM. At any point in time this TM can split in $O(1)$ copies of itself. It dies but leaves a boolean gate describing how to combine the subresults. Formally:

Definition 2 An ATM is a TM with a transition relation instead of a function. Each state is marked with a gate from the set $\{\wedge, \vee, \neg\}$.

A configuration, γ , of the machine is eventually accepting if

1. γ is accepting; or
2. γ is in an \wedge -state and all next configurations are eventually accepting; or
3. γ is in an \vee -state and some following configurations are eventually accepting; or
4. γ is in a \neg -state and the unique next configuration of γ is eventually rejecting.

We define eventually rejecting as not eventually accepting.

The resources used by an ATM are:

- Time: Total number of steps taken by any computation.
- Space: $\log(\text{number of configurations})$, $s(n) \geq \log(n)$. Measures the number of processes if we assume that processes that do the same magically merge into one process.

This gives rise to $\mathbf{ATIME}(t(n))$, $\mathbf{ASPACE}(s(n))$, and $\mathbf{ATS}(t(n), s(n)) = \{L \mid \text{decided by an ATM using time } t(n) \text{ and space } s(n)\}$.

Lemma 1 *An ATM with \neg -states can be simulated by an ATM without \neg -states with constant blowup in time and space.*

Proof: Instead of killing itself the ATM puts a bit on the worktape reminding itself it should have been killed.

1.4 Games

A one player game, a puzzle, can be described formally by a nondeterministic TM. We define the following formalism for describing two player games:

Definition 3 *A TM with a transition relation rather than a function.*

- *State of TM are divided into two sets. One set for the states that player one controls and one set that player two controls.*
- *Inputtape describes board and the transition relation describes the rules.*
- *There are two distinguished terminating states: Player one wins and player two wins.*
- *A configuration is an eventual win for player one/two if that player has a winning strategy from that point. Player one has a winning strategy if the configuraion is in a player one state and at least one next configuration is an eventual win. Likewise for player two.*

This definition is actually equivalent to the definition of an ATM and we can define $\mathbf{ATIME} = \{L \mid \text{A two player game can be played in time (=playing time) } t : \forall x \in L \Leftrightarrow \text{player one has a winning strategy on initial board, } x.x \notin L \Leftrightarrow \text{player two has a winning strategy on initial board, } x\}$.

Definition 4 $\mathbf{ATS}(a(\cdot), t(\cdot), s(\cdot)) = \text{Languages decided by time } t \text{ (playing time), space } s, \text{ and } a \text{ alternations = parse of control from one player to the other.}$