

# CMPT 710/407 - Complexity Theory

## Lecture 6: NP

Valentine Kabanets

September 20, 2007

### 1 Nondeterminism

**DTM (deterministic TM):** next step of computation is completely determined by the current configuration of a TM

**NTM (nondeterministic TM):** there may be several possibilities for a next step

Formally, NTM's transition function is  $\delta : Q \times \Sigma \rightarrow 2^{Q \times \Sigma \times \{L,R,-\}}$ , i.e., a pair (state,symbol) is mapped to a set  $\{(state,symbol,movement)\}$ .

Thus, DTM's computation is a path; whereas NTM's computation is a tree.

**Accept/Reject criteria for NTMs:**  $x \in L$  iff there exists some accepting computation.

**Remark 1.** *If  $x \in L$ , then there may be some rejecting computations as well.*

**Time** =  $\max_{\text{paths } p} \{\text{length of } p\}$

**Space** =  $\max_{\text{paths } p} \{\text{number of work-tape cells touched on a path } p\}$

### 2 Nondeterministic Complexity Classes

$\text{NTime}(f(n)) = \{L \mid \text{some multi-tape NTM decides } L \text{ in time at most } f(n)\}$

$\text{NSpace}(f(n)) = \{L \mid \text{some multi-tape NTM decides } L \text{ in space at most } f(n)\}$

- $\text{NP} = \cup_k \text{NTime}(n^k)$
- $\text{NEXP} = \cup_k \text{NTime}(2^{n^k})$
- $\text{NL} = \text{NSpace}(\log n)$
- $\text{NPSPACE} = \cup_k \text{NSpace}(n^k)$

**Remark 2.** *We'll see later that  $\text{NPSPACE} = \text{PSPACE}$ , and so there is really no need to use the name NPSPACE.*

### 3 Alternative definition of NP

$L \in \text{NP}$  if there is a language  $R \in \text{P}$  and a constant  $c$  such that

$$L = \{x \mid \exists y, |y| \leq |x|^c, (x, y) \in R\}$$

**Lemma 3.** *The two given definitions of NP are equivalent.*

*Proof.* Exercise. Hint:  $y$  in Definition 2 = accepting path in Definition 1. □

### 4 Importance of NP

NTMs cannot be efficiently implemented. So they are an abstraction. But, a huge number of real-life problems are in NP because they are of the form: problem description such that a solution to the problem is “small” and the solution is “easy” to test for correctness. (So we can nondeterministically guess a solution, and then test its correctness in polytime.)

### 5 Relations

**Lemma 4.**  $\text{P} \subseteq \text{NP}$  and  $\text{EXP} \subseteq \text{NEXP}$ .

*Proof.* Trivial: a DTM is a special case of an NTM. □

**Theorem 5.**  $\text{NP} \subseteq \text{PSPACE}$

*Proof.* Try all possible  $y$ 's (of polynomial length). □

**Major Open Problem:**  $\text{P} \stackrel{?}{=} \text{NP}$

This is a problem of “Generating a solution vs. Recognizing a solution”. Some examples: student vs. grader; composer vs. listener; writer vs. reader; mathematician vs. computer.

### 6 NP Completeness

$\text{Circuit-SAT} = \{C \mid C \text{ is a satisfiable Boolean circuit}\}$

**Theorem 6 (Cook-Levin).** *Circuit-SAT is NP-complete.*

*Proof.* We need to prove that

1. Circuit-SAT is in NP, and
2. Circuit-SAT is NP-hard (i.e., every language  $L \in \text{NP}$  reduces to Circuit-SAT).

The fact that Circuit-SAT is in NP is easy: Given a circuit  $C$  on variables  $x_1, \dots, x_n$ , nondeterministically guess an assignment to  $x_1, \dots, x_n$  and verify that this assignment is satisfying; this verification can be done in time polynomial in the size of the circuit. In other words,

$$\text{Circuit-SAT} = \{C \mid \exists x, |x| \leq |C|, R'(C, x)\}$$

where  $R'(C, x)$  is True iff  $C(x) = 1$  (i.e.,  $C$  on input  $x$  evaluates to 1).

Now we prove NP-hardness. Take an arbitrary  $L \in \text{NP}$ . Say

$$L = \{x \mid \exists y, |y| \leq |x|^c, R(x, y)\}$$

for some constant  $c$  and  $R \in \text{P}$ . Let's suppose that  $R(x, y)$  is computable in time  $N = (|x| + |y|)^d$ , for some constant  $d$ .

Consider  $N$  steps of computation of the Turing machine deciding  $R$  on input  $x, y$ . This computation can be pictured as a sequence of  $N$  configurations. A configuration at time  $t$  is a sequence of symbols  $y_1 \dots y_m$ , where each  $y_j$  contains the following information: the contents of tape cell  $j$  at time  $t$ , whether or not tape cell is being scanned by the TM at time  $t$ , and if it is, then what is the state of a TM at time  $t$ .

The crucial observation is that the computation of a TM has the following "locality property": the value of symbol  $y_i$  at time  $t + 1$  depends only on the values of symbols  $y_{i-1}, y_i, y_{i+1}$  at time  $t$  (as well as the transition function of the TM).

We can construct a constant-size circuit *Step* that computes the value of  $y_i$  at time  $t + 1$  from the values of  $y_{i-1}, y_i, y_{i+1}$  at time  $t$ . Now, we construct a big circuit  $C(x, y)$  by replacing each symbol  $y_i$  in every configuration at time  $t$  by a copy of the circuit *Step* whose inputs are the outputs of the corresponding three copies of *Step* from the previous configuration. We also modify the circuit so that it outputs 1 on  $x, y$  iff the last configuration is accepting.

The size of the constructed circuit will be at most  $N * N * |Step|$  ( $N$  configurations, at most  $N$  copies of *Step* in each), which is polynomial in  $|x|$ .

Our reduction from  $L$  to Circuit-SAT is the following: Given  $x$ , construct the circuit  $C_x(y) = C(x, y)$  as explained above (with  $x$  hardwired into  $C$ ). It is easy to verify that  $x \in L$  iff there is  $y$  such that  $C_x(y) = 1$ . So this is a correct reduction.  $\square$