Efficient Algorithms and Problem Complexity
– Nondeterminism and NP –

Frank Drewes
Department of Computing Science
Umeå University
Outline

Today’s Menu

1. Nondeterminism

2. Some Examples of Problems in NP

3. Polynomially Bounded Witnesses
A non-deterministic RAM (nRAM) $N$ is a RAM with one additional type of instruction:

$$\text{goto } k \mid l$$

where $k, l \in \{1, \ldots, m + 1\}$ for a program consisting of $m$ instructions. This instruction sets the program counter to $k$ or $l$.

$\Rightarrow$ for every input, $N$ has a set of valid computations (i.e., not only one). This set forms a binary tree that branches whenever an instruction $\text{goto } k \mid l$ is reached.
Definition: Acceptance and Deciding a Problem

An nRAM $N$ accepts an input if there exists at least one computation that outputs 1; it rejects the input if all computations output 0.

$N$ decides a decision problem $A$ if it accepts all positive instances and rejects all other inputs.

Notes:

- The important part of the definition is the upper part (acceptance and rejection). Notice the asymmetry!
- The definition of “decides” is the usual one.
Nondeterministic (Polynomial) Time

Definition: Running Time of an nRAM

Let $N$ be an nRAM. For a given input, the running time of $N$ is the maximum of the lengths of all computations of $N$ with this input.

As before, we are mainly interested in the worst-case running time with inputs of length $n$ (taking the maximum over all inputs of length $n$).

Definition: NP

NP is the set of all decision problems that can be decided in polynomial time by an nRAM.
Remarks about Nondeterminism

- Nondeterminism will perhaps never admit a faithful **physical realization**.
- The reason is the **unrealistic** definition of acceptance and running time.
- Intuitively, it assumes that we “magically” find the computation that accepts the input (if it exists).

**If it is unrealistic, why should we be interested in nondeterminism?**

- It isolates an **aspect of computation** whose role we do not understand.
- Basically, this aspect is **searching for a solution** (or a **witness**).
- In many cases, nondeterministic algorithms become **extremely simple**.
- **NP** is just across the border of efficient solvability. Or maybe not?
- In other words, if there is a chance to expand the area of efficient solvability, it is probably here you can find it!
NP is in Between P and EXP

Theorem

P ⊆ NP ⊆ EXP, where EXP (also called EXPTIME) is the set of all decision problems that can be decided by a RAM in time $O(2^{p(n)})$ for some polynomial $p$.

Proof sketch

P ⊆ NP because every RAM, by definition, is an nRAM.

For NP ⊆ EXP, suppose $A$ is decided by an nRAM $N$ in time $p(n)$. For an input of length $n$, the tree of all computations of $N$ has less than $2^{2p(n)}$ nodes. Thus, a RAM that simulates $N$ by working like $N$, but eliminates the nondeterminism by backtracking, decides $A$ in time $O(2^{2p(n)})$.

(The details require some care, though.)
Example: Independent Set

Input: An undirected graph $G = (V, E)$ and a number $k \in \mathbb{N}$.

Question: Does $G$ contain an independent set of size $k$, i.e., a set $V' \subseteq V$ with $|V'| = k$ and $(u, v) \notin E$ for all $u, v \in V'$?
Example: Independent Set

Deciding Independent Set nondeterministically:

Let $G, k$ be the input, where $G = (\{1, \ldots, n\}, E)$.

1. Use nondeterminism to generate $k$ numbers $v_1, \ldots, v_k \in \{1, \ldots, n\}$ (store them in $k$ registers).
2. For all $i = 1, \ldots, k$ and $j = i + 1, \ldots, n$, return 0 if
   - $v_i = v_j$ or
   - $(v_i, v_j) \in E$.
3. Accept the input (i.e., return 1).

The overall running time is $O(n^2)$.

$\Rightarrow$ Independent Set is in NP.
Example: Satisfiability (SAT)

Satisfiability (SAT)

Input: A propositional formula $\varphi$ in conjunctive normal form (CNF).

Question: Is $\varphi$ satisfiable, i.e., is there an assignment of truth values to the boolean variables that makes $\varphi$ true?

Example: $(\neg x_1 \lor x_2 \lor x_3) \land (x_1 \lor \neg x_2 \lor \neg x_3) \land (x_1 \lor x_2) \land (\neg x_1 \lor \neg x_2)$

Satisfiable or not?
Example: Satisfiability (SAT)

Deciding SAT nondeterministically:

Let $\varphi = C_1 \land \cdots \land C_k$ contain $m$ variables $x_1, \ldots, x_m$.

1. Use nondeterminism to generate $m$ bits $b_1, \ldots, b_m \in \{0, 1\}$ (store them in $m$ registers).

2. For $i = 1, \ldots, k$, return 0 if $C_i$ neither contains
   - a literal $x_j$ such that $b_j = 1$ nor
   - a literal $\lnot x_j$ such that $b_j = 0$.

3. Accept the input (i.e., return 1).

The overall running time is $O(n)$.

$\Rightarrow$ SAT is in NP.
Example: Hamiltonian cycle (HAM)

Input: An undirected graph $G = (V, E)$.
Question: Does $G$ contain a simple cycle of length $|V|$?
Example: Hamiltonian cycle (HAM)

Deciding HAM nondeterministically:

Let $G = (\{1, \ldots, n\}, E)$.

1. Start by setting $v = 1$.
2. Repeat $n - 1$ times:
   - If $v$ is marked, return 0; otherwise, mark $v$.
   - Nondeterministically choose a node $v'$ such that $(v, v') \in E$ and make $v'$ the new $v$.
3. Return 1 if $v = 1$, and 0 otherwise.

The overall running time depends on the details, but does certainly not exceed $O(n^3)$.

$\Rightarrow$ HAM is in NP.
NP can be defined entirely without nondeterminism.

A binary relation \( R \subseteq \mathbb{N}^* \times \mathbb{N}^* \) is polynomially bounded if there is a polynomial \( p \) such that \( \|v\| \leq p(\|u\|) \) for all \((u, v) \in R\).

**Theorem (characterization of NP by witnesses)**

A decision problem \( A \) is in \( \text{NP} \) if and only if there is a polynomially bounded binary relation \( R \in \mathcal{P} \) such that \( A = \{u \mid (u, v) \in R \text{ for some } v\} \).

If \((u, v) \in R\), then \( v \) “witnesses” that \( u \in A \).
If $A \in \text{NP}$ then $R$ exists:

Let $N$ be an nRAM that decides $A$ in time $O(p(n))$.

- For a computation $C$, encode the $m$ nondeterministic choices made (when executing instructions $\text{goto } k \mid l$) as a string $\text{enc}(C) \in \{0, 1\}^m$.
- Let $(u, v) \in R$ if and only if $v = \text{enc}(C)$ for a computation $C$ that accepts $u$ (i.e., returns 1).

$R$ is polynomially bounded because $|v| \leq p(||u||)$ for $(u, v) \in R$.

$R \in \text{P}$ by the RAM $M$ that, with input $(u, v)$, uses $v$ to simulate the computation $C$ of $N$ given by $v = \text{enc}(C)$.

$\Rightarrow M$ accepts $(u, v)$ if and only if $C$ accepts $u$. 
NP by Polynomially Bounded Relations and Witnesses

If $R$ exists then $A \in \text{NP}$:

Let $M$ be a RAM that decides $R$ in polynomial time, and let $\|v\| \leq p(\|u\|)$ for all $(u, v) \in R$. An nRAM $N$ deciding $A$ works as follows:

1. With input $u$, generate nondeterministically any $v \in \mathbb{N}^*$ with $\|v\| \leq p(\|u\|)$.
2. Continue deterministically like $M$ to decide whether $(u, v) \in R$.

$N$ runs in polynomial time, since phase 1 takes $O(\|v\|) = O(p(\|u\|))$ steps and phase 2 is the computation of $M$ (Composition Theorem!).

$N$ decides $A$, because

$$u \in A \iff \exists v. (u, v) \in R \iff N \text{ has an accepting computation}.$$
Section 10.2 in the textbook by Johnsonbaugh and Schäfer has a number of examples of well-known problems in \textbf{NP}.