

Capturing Nondeterministic Computation

$T(M, x, c)$ corresponds to a deterministic computation.

Each T_{ij} entry in its computation table can be captured by a Boolean circuit C with $3m + 1$ input gates (includes c_{i-1}) and $3m$ output gates.

Make the choice gate a variable gate.

The overall circuit is satisfiable iff there is a sequence of choices that ends in yes state (i.e., $x \in L$).

The reduction takes polynomial time.

Logical Characterization of NP

Let \mathcal{G} be a set of finite graphs — i.e., a graph theoretic property.

We say that \mathcal{G} is expressible in existential second-order logic if there is an existential second-order logic sentence $\exists P\phi$ such that $G \models \exists P\phi$ iff $G \in \mathcal{G}$.

The computational problem corresponding to \mathcal{G} is to decide, given a graph G , whether $G \in \mathcal{G}$.

We can denote by NP the sets of graph-theoretic properties whose corresponding computational problem is in NP.

Theorem 1. [Fagin's Theorem] *The class of all graph-theoretic properties expressible in existential second-order logic is precisely NP.*

Proof

If \mathcal{G} is expressible in existential second-order logic, we have shown that it is in NP.

We will show the converse. Let \mathcal{G} be a property in NP.

That is, there is a NTM M whether $G \in \mathcal{G}$ for some G with n nodes in time n^k ($k > 2$).

We will construct a second-order expression $\exists P\phi$ such that $G \models \exists P\phi$ iff $G \in \mathcal{G}$.

The input of M is the adjacency matrix of the graph and is given in the input string as follows: between any two entries we have $n^{k-2} - 1$ blanks.

P will be a Cartesian product of a list of new relations P_1, \dots, P_m .

That is, we will describe an equivalent relation of the form $\exists P_1 \dots \exists P_m \phi$.