On Succinctness

Theorem. Let $\varepsilon > 0$. For large n, the fraction of functions

$$\{0,1\}^n \to \{0,1\}$$

that can be represented by formulae of size up to

$$2^{(1-\varepsilon)\cdot n}$$

tends to zero.

In other words, almost all function require exponentially large formulae.

The reason is quite simple: there just not enough short formulae!

> **Proof.** Functions $\{0,1\}^n \to \{0,1\}$ have 2^n possible different inputs; hence
>
> $$2^{2^n}$$
>
> of them.
>
> ...

Note that for each input, we can independently set the output to 0 or 1. This gives

$$\underbrace{2 \cdot 2 \cdot \ldots \cdot 2}_{2^n} = 2^{2^n}$$

possibilities.

> ...
>
> Formulae with n variables: $n + C$ symbols.
>
> ...

Note that, besides the variables $p_1, \ldots, p_n$, the only symbols we're using are $\wedge, \vee, \neg, (,)$. The result will not change (just take a bigger C) if we allowed a finite set of more connectives (like $\to, \leftrightarrow$).

...
Hence $(n+C)^s$ formulae of size s. If $s = 2^{(1-\varepsilon)n}$ we get

$$(n+C)^{2^{(1-\varepsilon)n}}$$

of them.
The fraction is

$$\frac{(n+C)^{2^{(1-\varepsilon)n}}}{2^{2^n}} = 2^{2^{(1-\varepsilon)n}\cdot\log(n+C)-2^n}$$

which tends to 0, as

$$2^n \gg 2^{(1-\varepsilon)n} \cdot \log(n+C)$$

for large n.

1.3 Model-Checking Satisfiability

Model checking propositional logic is easy; by the very definition of the semantics, the truth of a formula only depends on the truth of its subformulae.

$\boxed{\textbf{Model Checking in Propositional Logic}}$

Given: Propositional formula $\varphi \in \mathsf{PL}[p_1, \dots, p_n]$ and $\underline{\mathrm{a}} \in \{0,1\}^n$

Question: $\underline{\mathrm{a}} \models \varphi$?

Solvable in polynomial time (essentially $\mathcal{O}(|\varphi|)$): just compute truth value following the buildup of φ.

Note the argument: the truth can be defined merely in terms of the truth values of smaller formulae—and there are only a small number of them. We will see similar "dynamic programming" arguments later in the course as well.

The satisfiability problem is (believed to be) much harder.

Satisfiability in Propositional Logic

Given: Propositional formula $\varphi \in \mathsf{PL}[p_1, \ldots, p_n]$.

Question: Is there some $\underline{\mathrm{a}} \in \{0,1\}^n$ such that $\underline{\mathrm{a}} \models \varphi$?

This problem is NP-complete.

Before we're going to prove this fact, let's explain first, what NP-complete means.

$\boxed{\mathsf{NP}}$

Definition. A problem is said to be in NP iff it can be solved by a non-deterministic Turing machine in polynomial time.
"NP *is verifying proofs*"

Conjecture. $\mathsf{P} \neq \mathsf{NP}$.

The conjecture $\mathsf{P} \neq \mathsf{NP}$ is believed by most computer scientists. Nevertheless, it is an open problem (in fact, one of the millennium problems) to prove or disprove it.

NP-completeness

Definition. A problem L is NP-complete iff

- it belongs to NP
- for any problem L' in NP there is an easy (say, in polynomial time) function f such that

$$x \in L' \text{ iff } f(x) \in L$$

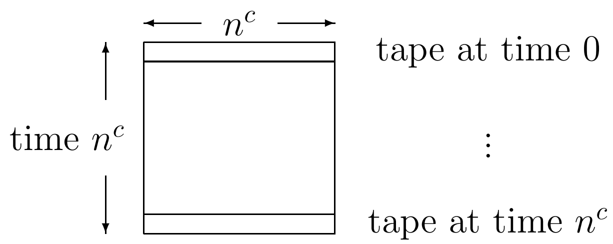
In other words, a problem is NP-complete if it is "the hardest" problem in NP. In particular, if one NP-complete problem can be solved in polynomial time, then $\mathsf{P} = \mathsf{NP}$.

Now, let's prove the stated complexity-result about the satisfiability problem.

Theorem. The satisfiability problem for propositional logic is NP-complete.

Proof Sketch.
"in NP": just guess $\underline{a}$. Then verify as in the model-checking problem.

"NP-hard": given a TM deciding L in time n^c, it uses at most that space. So its runs can be described by an $n^c \times n^c$ table.



Introducing variables for "at time t the symbol at position i is a" and "at time t the head is at position i" leads to local conditions on this scheme, expressible in propositional logic.