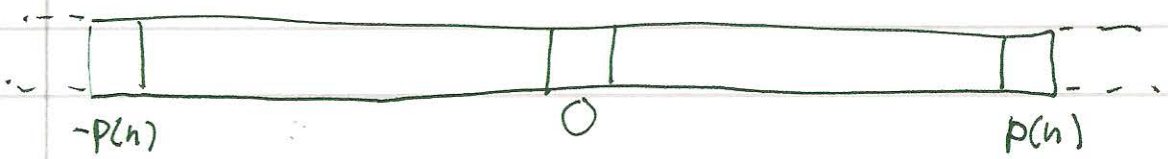
2) Cook's theorem and the complexity of variants of SAT

Lemma 9

Let any Turing machine M running in time at most $p(n) \geq n$ on inputs of length n, where p is a polynomial, be given.
Then, given any fixed input length n, there is a circuit C_n of size at most $O(p(n)^2)$ so that $\forall x \in \{0,1\}^n$:
$C_n(x) = 1 \Leftrightarrow M$ accepts x.
Furthermore, the function mapping 1^n to a description of C_n is polynomial time computable.

Proof

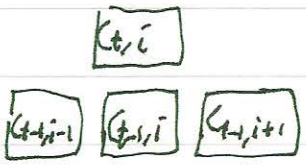


Define for input x, and each position $i \in \mathbb{Z}$ and for each point in time $t \in \{0, \ldots, p(n)\}$ a cell state vector $C_{t,i}$. It will hold:

- Symbol in cell
- Is tape head at cell?
  If so, record state of finite control.

We may encode $C_{t,i}$ as a string $C_{t,i} \in \{0,1\}^s$ where s only depends on M.

Now we see that without knowing x, we can compute $C_{t,i}$ $t > 1$ from $C_{t-1,i-1}$, $C_{t-1,i}$ and $C_{t-1,i+1}$.



~~We can have~~

We have a boolean function from these cells to the cell above. By propersition 8, there is a curcuit that does this. Call it D.

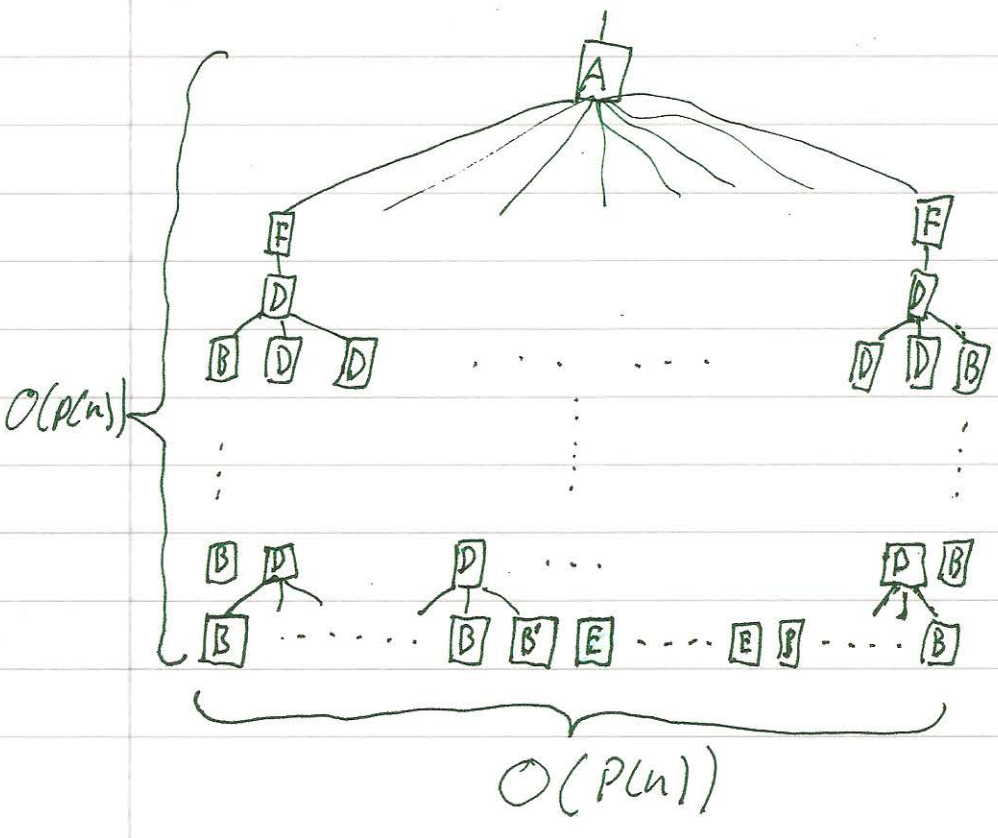
We then detine some helper curcuits:

$E: \{0,1\} \to \{0,1\}^s$ so that $E(b)$ is a cell state vector with $b \in \{0,1\}$ in the cell and head is not there.

$F: \{0,1\}^s \to \{0,1\}$ such that given a cell state vector it will output 1 iff the cell state $y \in \{0,1\}^s$ has the tape head and is in accepting state.

B is acollection of constant gates so that B's output describes a cell state vector with a blank and no head.

C_n is then:



The regular nature makes it easy to do an algorith that outputs C_n. It can hold D, E, F and simply paste them together. The polynomial Turing Thesis then says it can be done in polynomial time.

Theorem 11 : Circuit SAT $\in$ NPC

Clearly Circuit sat is in NP → just try all possible inputs.

Proof of NP-hardness

Let $L \in NP$

We must show:

$\forall x: x \in L \Leftrightarrow r(x) \in$ Circuit sat

Since $L \in NP$, we have $L' \in P$ and a polynomial p such that:

$\forall x: x \in L \Leftrightarrow \exists y \in \{0,1\}^* : |y| \leq p(|x|) \wedge \langle x, y \rangle \in L'$

~~Define $r(x) \equiv$~~

$r(x)$ is going to be a Circuit $C \equiv D_0 \vee D_1 \vee \cdots \vee D_{p(n)}$

Each circuit D_i takes i boolean inputs.

$D_i(y)$ $y \in \{0,1\}^i$ should be $1 \Leftrightarrow \langle x, y \rangle \in L'$

Define D_i as follows. M is a turing machine deciding L' in polynomial time. By Lemma 9 we have a circuit C_n that accepts inputs of length n.

$\forall x,y$: $n = |\langle x, y \rangle|$ $C_n(\langle x, y \rangle) = 1 \Leftrightarrow M$ accepts $\langle x, y \rangle$.

$\langle x, y \rangle = x_1 0 x_2 0 \cdots x_{n-2i} 1 y_1 0 \cdots y_i 0$

Now replace $x_i, 0, 1$ with constant gates. Let $y_1 \cdots y_i$ be input gates. D_i is this construction.

The reduction constructs $D_0 \cdots D_{p(|x|)}$ and wire them together with OR gates.

Appealing to The polynomial Church Turing Thesis, this can be done in polynomial time

Theorem 12: Circuit SAT $\leq$ SAT

Given a circuit (with one output we should define a reduction $\phi = r(C)$, where ϕ is a CNF formula. r should be polynomially time computable map.

We ~~define~~ let ϕ ~~has~~ have a variable for each gate.

ϕ will have the following clauses for each gate

AND $\quad g \Leftrightarrow h_1 \wedge h_2$

$(\neg g \vee h_1) \wedge (\neg g \vee h_2) \wedge (g \vee \neg h_1 \vee \neg h_2)$

OR $\quad g \Leftrightarrow h_1 \vee h_2$

$(g \vee \neg h_1) \wedge (g \vee \neg h_2) \wedge (\neg g \vee h_1 \vee h_2)$

NOT $\quad g \Leftrightarrow \neg h$

$(\neg g \vee \neg h) \wedge (g \vee h)$

COPY $\quad g \Leftrightarrow h$

$(\neg g \vee h) \wedge (g \vee \neg h)$

Constant gates $\quad 1 \quad 0$

$(g) \quad (\neg g)$

Output gate

(g)

ϕ clearly have the desired properties and by appealing to the Polynomial Church Turing Thesis r is a polynomial time computable map.