y is also assigned T.
- by induction, if x connected to y by a path, if x assigned T, so is y.
- otoh if $\neg x$ assigned T (so x assigned F) and connected to y, then y also assigned T
- one of $x, \neg x$ is assigned T, hence so is the other, contradiction

($\Rightarrow$) - now assume there are no such paths we check there is a valid truth assignment satisfying $\wedge c_i$
- we claim that if t is a partial (valid) truth assignment s.t. if $(x,y) \in E$ and $t(x) = T$ then $t(y) = T$, then t can be extended to a larger truth assignment t' w/ the same property

- sps we have such a t, and $x \notin dom(t)$
- either there is no path from x to $\neg x$, or via versa.
- assume no path from x to $\neg x$
- extend t by defining $t'(x) = T$ and also $t'(y) = T$ for all y reachable from x by a directed path (second case analogous, but setting $t'(x) = F$)

- we check that t is a well-defined truth assignment extending t
- observe: there is a path from x to y in G iff there is a path from $\neg y$ to $\neg x$

$$\neg x \vee (\cdot) \to \neg(\cdot) \vee (\cdot\cdot) \to \cdots \to \neg(\cdots) \vee y$$
$$\Leftrightarrow \quad y \vee \neg(\cdots) \to \cdots \to (\cdot\cdot) \vee \neg(\cdot) \to (\cdot) \vee \neg x$$

- follows: can't be any path from x to both y and $\neg y$, for any y
  - cause then: paths from $\neg y$ to $\neg x$
    y to $\neg x$
  - then: path from x to $\neg x$, against hypothesis

$\Rightarrow$ new truth assignments valid

- so only need to check: if there is a path from x to some y then already $t(y) = T$
- if not then $t(y) = F$, so $t(\neg y) = T$
- but there is a path from $\neg y$ to $\neg x$ so $t(\neg x) = T$ so $t(x) = F$, contradicting $x \notin \mathrm{dom}(t)$.

thus, beginning from empty truth assignment, can get one w/ complete domain satisfying this property.

but then if $a \vee b$ is one of our clauses, can't have $t(a) = t(b) = F$

Why: if $t(a) = F$ then $t(\neg a) = T$ and hence $t(b) = T$ since there is a directed path (of length 1) from $\neg a$ to b in G ✓

Hence t satisfies $\wedge C_i$ ✓

Since checking if there are directed paths from x to $\neg x$ for every $x = x_i$ for some i, is poly in size of G, 2-SAT is in P ✓

The class NP

- we define two classes (of problems) which are (possibly) intractable: NP and EXP
- to define NP, need notion of a non-deterministic Turing machine (NTM)
- difference from reg. TM is: from a given state and tape input, can transition to more than one new state (transition relation instead of transition function)

- more formally, an NTM consists of:
  - an alphabet A
  - a finite set of states Q_i, $i \in \{s, y, n, 1, \ldots k\}$ (Q_s initial, Q_n, Q_y final states)
  - finite collection of instructs (Q_i, a, b, D, Q_j) s.t. $\forall i \neq y, n$ $\forall a \in A$ there is at least one such instruction (could be more); as before $D \in \{R, L\}$, Q_j can be halting

→ same meaning: if in state Q_i, tape reads a, then write b, move D, go to state Q_j.

- Given input word w, new more than one way computation of M can proceed
- i.e. have a computation tree for M, w:
  - each node reps a configuration (word printed + current state + head position) descendents rep configs to which M can possibly transition (one for each instruction corring to current config.)

- given word w, say M accepts w if there is a path in computation tree for M, w from root to a

config. halting in state q_y.

- given $t : \mathbb{N} \to \mathbb{N}$ a function, M runs in time t if for every w, height of comp'n tree for w is $\leq t(|w|)$
  (i.e. on input w all possible computations halt in time $\leq t(|w|)$)

Def'n Fix A an alphabet an $L \subseteq A^*$ is a language. We say L is in NP if there is an n.t.m M (on alphabet $\supseteq A$) and a polynomial t s.t. M runs in time t and

$w \in L$ iff M accepts w.

- since t.m.'s are n-tms follows that $P \subseteq NP$
- whether $NP \subseteq P$ is among most famous open problems in math.

- we formulate alternative def'n of NP
- idea is: NP problems can be solved in P time... given "polynomial amount" of extra info at start.

Prop'n $L \in NP$ iff there is a poly t and t.m. M (on alphabet $\Sigma \supseteq$ alphabet of L) (assume Σ contains a "separator symbol" ;) s.t.

$w \in L$ iff

$\exists u \in \Sigma^{<N}$ with $|u| \leq t(|w|)$ s.t. on input $w;u$ M halts in $\leq t(|w|)$-many steps in state Q_y.

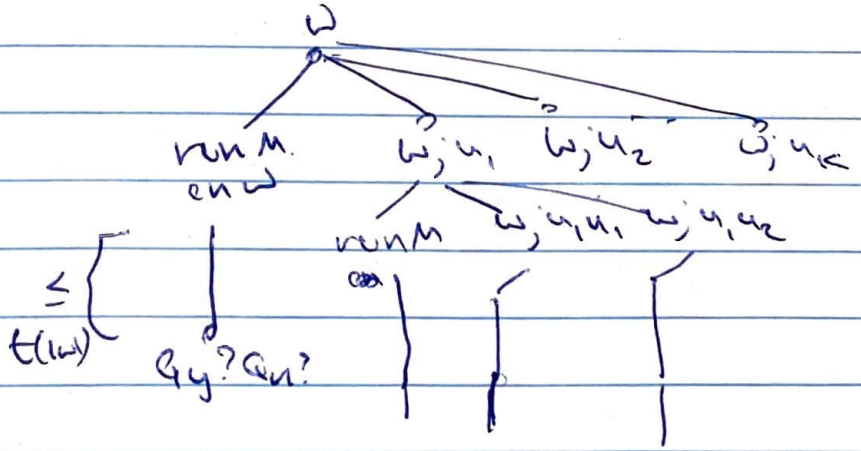
(Such a u is called a <u>witness</u> or <u>certificate</u> for w)

Pf ($\Rightarrow$) ("Given a solution, check that it works")

- Sps $L \in NP$ and K, p. are n.t.m. and polynomial witnessing this
- Construct a t.m. M that, on input $w;u$, checks if u encodes a possible computation from K on input w to a halting config'n in state Q_y (i.e. a path in computation tree witnessing K accepts w)
- since length of such a path is $\leq p(|w|)$ there is a poly t so that M can be constructed to halt in $\leq t(|w|)$ many steps given such $w;u$.

($\Leftarrow$) ("Check all possible solutions")

- Now suppose M, t are as in statement of prop'n
- We construct an n.t.m K that runs in poly time and decides L.
- K is machine that, on input w, runs M on all possible inputs $w;u$ for words u up to length $t(|w|)$
- Slightly more specifically: K does (non-deterministic) computations of two kinds. (1) print some letter u_i at end of input word (2) run M on current word
- on input w, only prints letters up to $t(|w|)$-many times
- height of computation tree of K, w on order of $t(|w|)$



NP-complete problems

- many natural problems are not only NP, but NP-complete.

Problem (SAT) given variables $x_1, \ldots, x_k$ and clauses $c_1, \ldots, c_n$, each a disjunction (of any length) of variables x_i and negated variables $\neg x_i$, decide whether

$$\bigwedge_i c_i$$

is satisfiable.

Theorem (Cook-Levin) SAT is NP-Complete.

PF: need to show ① SAT is in NP ② any other NP prob can be reduced to SAT.

- ① is clear: checking if a truth assignment satisfies $\bigwedge c_i$ only takes poly time (i.e. truth assignments work as certificates)
- We prove ② — point is that computations of a t.m. can be encoded in a SAT problem.