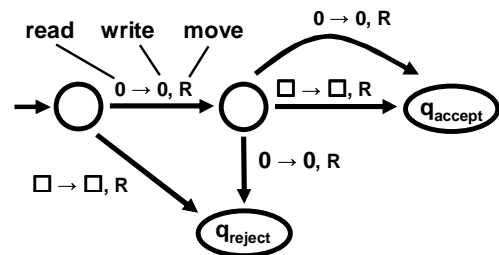


NON-DETERMINISTIC TURING MACHINES AND NP

THURSDAY OCTOBER 20

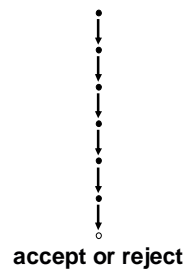


NON-DETERMINISTIC TMs

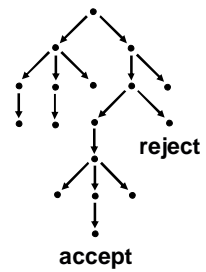
...are just like standard TMs, except:

1. The machine may proceed according to several possibilities
2. The machine accepts a string if there exists a path from start configuration to an accepting configuration

Deterministic
Computation



Non-Deterministic
Computation



Definition: A Non-Deterministic TM is a 7-tuple
 $T = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$, where:

Q is a finite set of states

Σ is the input alphabet, where $\square \notin \Sigma$

Γ is the tape alphabet, where $\square \in \Gamma$ and $\Sigma \subseteq \Gamma$

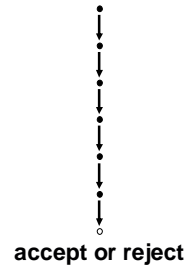
$\delta : Q \times \Gamma \rightarrow 2^{(Q \times \Gamma \times \{L,R\})}$

$q_0 \in Q$ is the start state

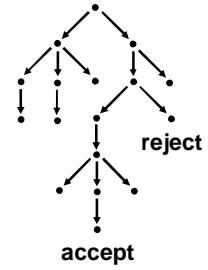
$q_{\text{accept}} \in Q$ is the accept state

$q_{\text{reject}} \in Q$ is the reject state, and $q_{\text{reject}} \neq q_{\text{accept}}$

**Deterministic
Computation**



**Non-Deterministic
Computation**



Definition: $NTIME(t(n)) = \{ L \mid L \text{ is decided by a } O(t(n))\text{-time non-deterministic Turing machine } \}$

$$TIME(t(n)) \subseteq NTIME(t(n))$$

BOOLEAN FORMULAS

A satisfying assignment is a setting of the variables that makes the formula true

A Boolean formula is satisfiable if there exists a satisfying assignment for it

$$a \wedge b \wedge c \wedge \neg d$$

$$\neg(x \vee y) \wedge x$$

$$\text{SAT} = \{ \phi \mid \phi \text{ is a satisfiable Boolean formula} \}$$

A 3cnf-formula is of the form:

$$(x_1 \vee \neg x_2 \vee x_3) \wedge (x_4 \vee x_2 \vee x_5) \wedge (x_3 \vee \neg x_2 \vee \neg x_1)$$

clauses

$$(x_1 \vee \neg x_2 \vee x_3)$$

$$(x_3 \vee x_1) \wedge (x_3 \vee \neg x_2 \vee \neg x_1)$$

$$(x_1 \vee x_2 \vee x_3) \wedge (\neg x_4 \vee x_2 \vee x_1) \vee (x_3 \vee x_1 \vee \neg x_1)$$

$$(x_1 \vee \neg x_2 \vee x_3) \wedge (x_3 \wedge \neg x_2 \wedge \neg x_1)$$

$$\text{3SAT} = \{ \phi \mid \phi \text{ is a satisfiable 3cnf-formula} \}$$

$$\text{3SAT} = \{ \phi \mid \phi \text{ is a satisfiable 3cnf-formula} \}$$

Theorem: 3SAT \in NTIME(n^2)

On input ϕ :

$$\text{NP} = \bigcup_{k \in \mathbb{N}} \text{NTIME}(n^k)$$

Theorem: $L \in NP$ if and only if there exists a poly-time Turing machine V with

$$L = \{ x \mid \exists y \ |y| = \text{poly}(|x|) \text{ and } V(x,y) \text{ accepts} \}$$

Proof:

(1) If $L = \{ x \mid \exists y \ |y| = \text{poly}(|x|) \text{ and } V(x,y) \text{ accepts} \}$ then $L \in NP$

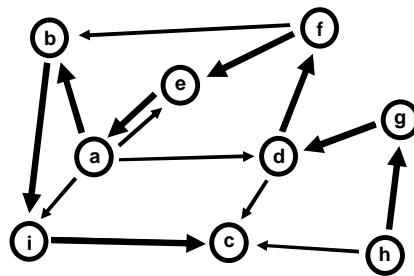
(2) If $L \in NP$ then
 $L = \{ x \mid \exists y \ |y| = \text{poly}(|x|) \text{ and } V(x,y) \text{ accepts} \}$

$3SAT = \{ \phi \mid \exists y \text{ such that } y \text{ is a satisfying assignment to } \phi \text{ and } \phi \text{ is in } 3cnf \}$

$SAT = \{ \phi \mid \exists y \text{ such that } y \text{ is a satisfying assignment to } \phi \}$

A language is in NP if and only if there exist polynomial-length certificates for membership to the language

HAMILTONIAN PATHS

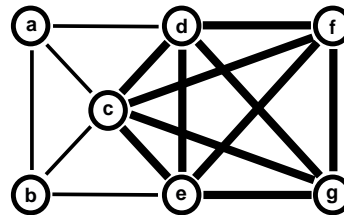


HAMPATH = { (G,s,t) | G is a directed graph with a Hamiltonian path from s to t }

Theorem: HAMPATH \in NP

The Hamilton path itself is a certificate

K-CLIQUE



CLIQUE = { (G,k) | G is an undirected graph with a k-clique }

Theorem: CLIQUE \in NP

The k-clique itself is a certificate

NP = all the problems for which once you have the answer it is easy (i.e. efficient) to verify

P = NP?

If P = NP...

Mathematicians would be out of a job

Cryptography as we know it would not be possible

We could determine if cold fusion is possible

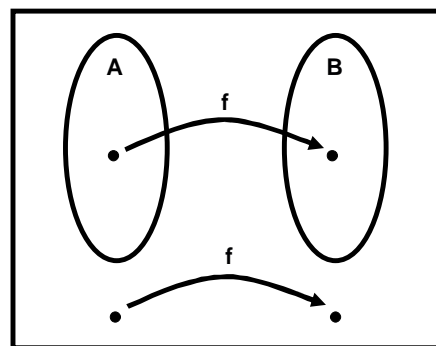
POLY-TIME REDUCIBILITY

$f : \Sigma^* \rightarrow \Sigma^*$ is a polynomial time computable function if some poly-time Turing machine M , on every input w , halts with just $f(w)$ on its tape

Language A is polynomial time reducible to language B , written $A \leq_p B$, if there is a poly-time computable function $f : \Sigma^* \rightarrow \Sigma^*$ such that:

$$w \in A \Leftrightarrow f(w) \in B$$

f is called a polynomial time reduction of A to B



Theorem: If $A \leq_p B$ and $B \in P$, then $A \in P$

Proof: Let M_B be a poly-time (deterministic) TM that decides B and let f be a poly-time reduction from A to B

We build a machine M_A that decides A as follows:

On input w :

1. Compute $f(w)$
2. Run M_B on $f(w)$

WWW.FLAC.WS

Read Chapter 7.3 of the book for next time