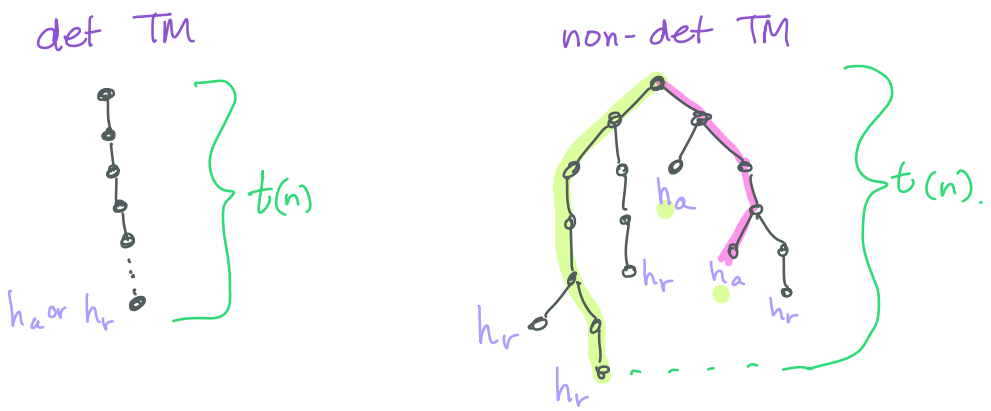
Theorem 6.1.2: $\forall$ multitape $t(n)$-time TM,
$\exists$ a $O(t^2(n))$-time single tape TM.

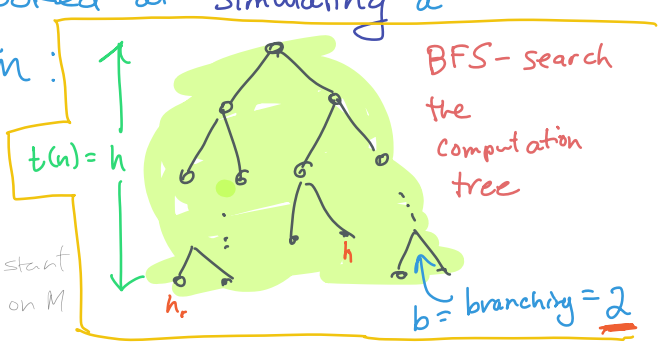
We covered, . . . , how to convert multi-tape to single-tape, and it turns out that conversion "squares" the run-time.

Defn 7.9 Let $N$ be a non-det decider TM. The running time of $N$ is a function $t: \mathbb{N} \to \mathbb{N}$ where $t(n)$ is the max # of steps that $N$ uses on any branch of a computation on an input of size $n$.




Thm 7.11 Every $t(n)$-time (single-tape) non-det TM has an equivalent $2^{O(t(n))}$-time (single-tape) det-TM.

Proof sketch: We already looked at simulating a non-det TM with a det-TM:
$\exists$ $O(b^{t(n)})$ nodes in a tree of height $t(n)$ and branching factor $b$. ← fixed constant dependent on M



$$b^{t(n)} = \left(2^{\lg b}\right)^{t(n)} = 2^{\lg b \cdot t(n)} \in 2^{O(t(n))}$$

BFS runs in time linear in number of nodes visited, using multiple tapes – the "slow down" due to single-tape is absorbed in the big-O. ▨

## 6.2 The Class P

Defn 7.12 P = class of languages decidable in poly time by a det-TM

$$\text{i.e } P = \bigcup_k \text{TIME}(n^k)$$

P is the class of feasibly computable tasks when the input is fairly large. P is also largely independent of underlying computational model TM C++ $\lambda$-calculus

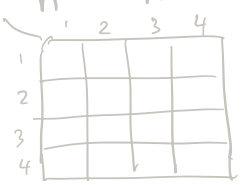
Eg Path = $\{ \langle G, s, t \rangle \mid$ G is a directed graph that has a s-t path $\}$

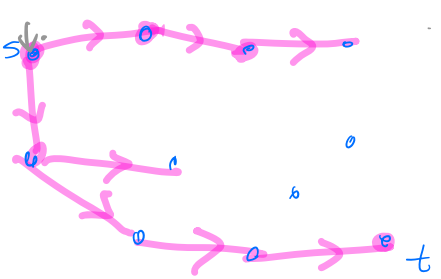
Thm: Path $\in$ P

Proof: "mark s
- $\forall$ edges $(\vec{u,v})$, if u is marked, v unmarked, mark v.
- Stop when no new vertices are marked.
- if t is marked, ACCEPT. otherwise REJECT."

This alg is $\in O(n^3)$.

n = number of vertices



# 6.3 The Class NP

Number of tapes – only a polynomial factor change to running time to switch to single tape.

Non-determinism – exponential change in running time to remove non-determinism.

$A$ = a language that has a poly-time non-det decider

$A_{cert}$ = a language $\{(w, c) \mid w \in A$ and $c$ is a certificate of $w$'s membership in $A\}$

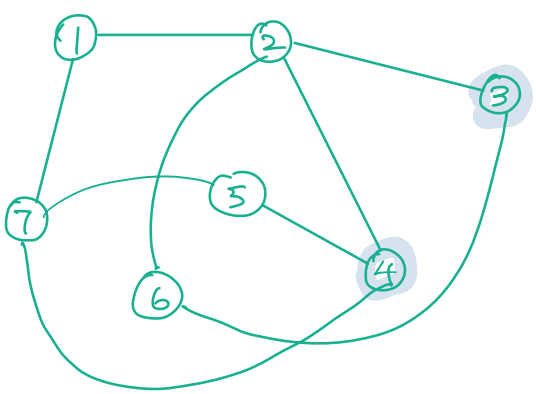
where $A_{cert}$ has a poly-time det decider, $V_A$, called a Verifier for $A$.

Ham Path = path that visits each node exactly 1$^c$

s,t-HamPath = $\{\langle G, s, t\rangle \mid$ undirected graph $G$ has a Ham Path from $s$ to $t\}$

s,t-HamPath$_{cert}$ = $\{\langle G, s, t, P\rangle \mid G$ is an undirected graph, and $P$ is an $s$-$t$ path in $G.\}$

G:



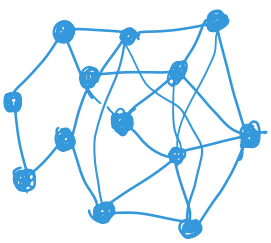

$\langle G, 3, 4\rangle \in$ s,t-HamPath? $\langle G, 3, 4, (3,6,2,1,7,5,4)\rangle \in$ s,t-HamPath$_{cert}$?

A poly-time verifier for Hampath would be a program that in poly-time can check that a given sequence of verts certifies $\langle G, s, t \rangle$'s membership in Hampath...

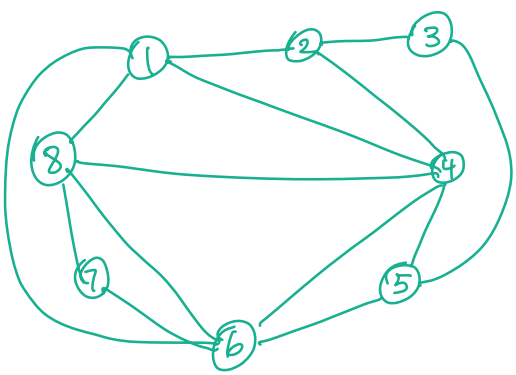
ie that the given path $v_1, v_2, \ldots, v_n$
- starts with $s$, ends in $t$
- contains each vertex exactly once
- $\forall i,\ 1 \le i < n,\ (v_i, v_{i+1}) \in E(G)$

Defn 6.3.6 NP is class of languages that have poly-time verifiers. (det-TM)

Eg. a clique is a subgraph of Graph where all vertices are mutually adjacent



Clique = $\{ \langle G, K \rangle \mid G$ is a graph that contains a clique of size $K \}$.




← Does G have a clique of size 4?

← Is , , a clique of size 4?

Theorem: $s,t$-HamPath $\in$ NP.

Proof: The path itself is a certificate of membership. that $\langle G, s, t\rangle \in$ $s,t$-HamPath. The path $v_1, v_2, \ldots, v_n$ can be verified in poly time as follows:

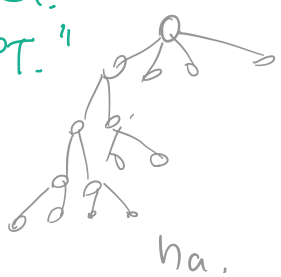
1. confirm $v_1 = s$ and $v_n = t$
2. check that every vertex in $V(G)$ is on the path once.
3. confirm that $(v_i, v_{i+1}) \in E(G)$ $\forall\, i,\ 1 \le i < n$.

This can be done in $O(n^2)$ time. ▨

There is also a poly-time non-det TM N that decides $s,t$-HamPath.

N = " on input $\langle G, s, t\rangle$, where G is a graph, and $s, t \in V(G)$:

1. Write down a permutation $v_1 = s, v_2, \ldots, v_n = t$ of $V(G)$.
2. Check that for each $i$, $1 \le i < n$, $(v_i, v_{i+1}) \in E(G)$
   - if any fail the test, REJECT.
   - if all pass, ACCEPT. "



ha.

Note: graph $G = (V, E)$
$V(G)$ = vertex set
$E(G)$ = edge set.

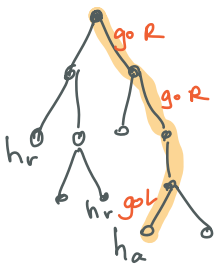
Theorem: 7.20: A language is $\in$ NP iff it is decided by some non-det poly-time TM.

Proof: ($\Rightarrow$) $\exists$ language A has a poly-time verifier V.

Then we can construct a poly-time TM N that "guesses" the certificate and then run's the verifier on that certificate. If it "lucky-guessed" a correct certificate, the computation will accept, if $w \in A$. And N runs in poly-time.

($\Leftarrow$) If A has a poly-time non-det decider N.

Then we can construct a verifier V that has, as its certificates, DIRECTIONS to the $h_a$ state in the computation tree of N on w.




(go R, go R, go L)

Then run N, and follow directions when given a choice. deterministic, and poly-time. ▨

Defn 7.21 (MW 6.3.6) $NTIME(t(n)) = \{ L \mid$ L is a language decided by a non-det TM in $O(t(n))$-time $\}$

Corollary: $NP = \bigcup_k NTIME(n^k)$.

Clique $= \{ \langle G, k \rangle \mid$ G is an undirected graph that has a clique of size k $\}$.

Theorem: Clique $\in$ NP.

Proof: The poly-time checkable CERTIFICATE OF MEMBERSHIP is a set of vertices from G.

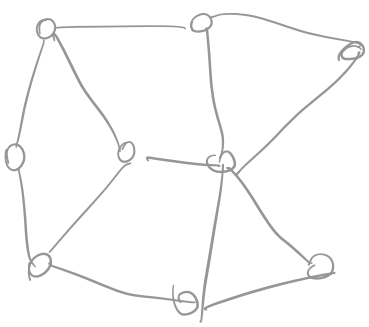
VERIFYING that the certificate does, indeed, prove membership is done as follows:

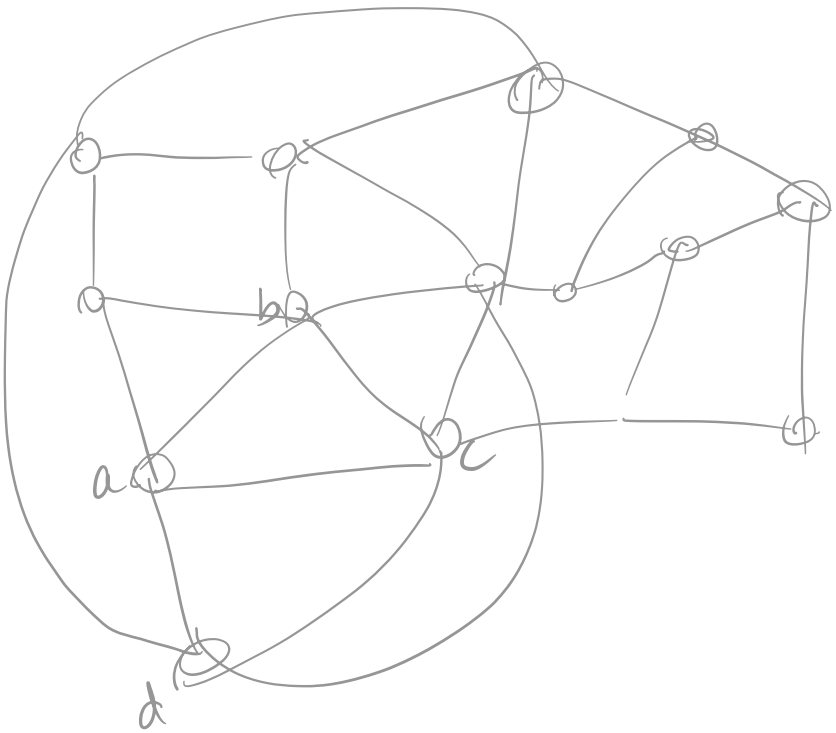
1. count the vertices — there should be $K$
2. For each pair of verts $i, j$, check that $(i, j) \in E(G)$ ($K$ edges of $G$.)

The running time for step 1 is $O(n)$ since $K < n$

The running time for step 2 is $O(n^2)$ (actually $O(K^2)$)

$\therefore$ total running time is $O(n^2)$, ie is polynomial time. ▨






$\{9, 15, 26, 42, 105, 107\}, 205$

$SubsetSum = \{\langle S, K\rangle \mid S$ is a multiset of integers, $K$ is an integer, and $\exists$ a subset of $S$ that sums exactly to $K\}$.

Alg to decide if $\langle S, K\rangle \in SubsetSum$:

Certificate of membership of $\langle S, K\rangle \in SubsetSum$, and alg to verify the certificate: