▷ A proof for $P \stackrel{?}{=} NP$ will be non-relativizing.

More on space complexity

Defn: • **Nspace(f(n))** $:= \{L \mid \exists$ NDTM M that decides L using $O(f(n))$ space$\}$.

• **Npspace** $:= \bigcup_{c \in \mathbb{N}} \text{Nspace}(n^c)$.

• **Pspace** $:= \bigcup_{c \in \mathbb{N}} \text{Space}(n^c)$.

• **NL** $:=$ Nspace$(\log n)$

• **L** $:=$ Space$(\log n)$.

– Eg. of a problem in L?
Addition, multiplication!

▷ $P^{\text{Pspace}} = NP^{\text{Pspace}} = \text{Pspace}$.

Proposition: (i) $\text{Dtime}(f) \subseteq \text{Space}(f) \subseteq \text{Nspace}(f) \subseteq \text{Dtime}(2^{O(f)})$.

(ii) $P \subseteq NP \subseteq \text{Pspace} \subseteq \text{EXP}$.

Proof: • Observe that a TM using $f(n)$ space can have $O(2^{f(n)})$ configurations.
(More than this makes the computation cyclic.)

• So, $2^{f(n)}$ upper bounds the time complexity. □

Pspace completeness

Defn: A language B is Pspace-complete if
- $B \in$ Pspace, and
- $\forall A \in$ Pspace, $A \leq_p B$.

- We saw that $\exists$-quantifiers gave us an NP-complete problem.
  What happens if we also use $\forall$?

Defn: $\text{TQBF} := \{ \overbrace{Q_1 x_1 \cdots Q_n x_n\ \phi(\bar{x})}^{\text{quantified boolean formula}} \mid Q_i\text{'s are quantifiers, } \phi \text{ is a boolean}$

formula in $x_1,\dots,x_n$ & $Q_1x_1\cdots Q_nx_n\,\Phi(\bar{x})$ is true$\}$.

<u>Lemma 1</u>: TQBF $\in$ Pspace.

<u>Proof</u>:

- Let $\psi := Q_1x_1\cdots Q_nx_n\,\Phi(\bar{x})$ be a QBF with $|\Phi| =: m$.
- We can check its truth by the following recursive algorithm:

ψ is true iff

$Q_1 = \exists$ & $(Q_2x_2\cdots Q_nx_n\,\Phi(0,x_2,\dots,x_n)$ or $Q_2x_2\cdots Q_nx_n\,\Phi(1,x_2,\dots,x_n))$

$Q_2 = \forall$ & $(Q_2x_2\cdots Q_nx_n\,\Phi(0,x_2,\dots,x_n)$ & $Q_2x_2\cdots Q_nx_n\,\Phi(1,x_2,\dots,x_n))$.

- Its space complexity $s(n,m)$ is given by:

$s(n,m) \le \underline{s(n-1,m)} + O(m)$. ← we reuse space, & not double it

Improve to $O(n+m)$ → $\Rightarrow s(n,m) = O(nm)$.

$\Rightarrow$ TQBF $\in$ Pspace. $\square$

<u>Lemma 2</u>: $\forall L \in$ Pspace, $L \leq_p$ TQBF.

<u>Proof</u>:

- Let M be an $S(n)$-space TM deciding $L \in$ Pspace.
- We will construct a QBF $\psi_{M,x}$, of size $O(S(n)^2)$, whose truth depends on M accepting x.

- By Cook-Levin reduction, we have a formula $\varphi_{M,x}(C,C')$, of size $O(S)$, that is true iff $C \to C'$ is a valid <u>transition</u> step of configurations of M on x.
- Now, M on input x ($|x| =: n$) can have at most $2^{d \cdot S(n)}$ distinct configurations (for some constant d).
- It is possible to design a QBF $\psi_i(C, C')$ that captures whether C to C' is reachable in $\leq 2^i$ transition steps of M:

Recursive defn

$$\psi_i(C,C') := \exists C'' \left(\psi_{i-1}(C,C'') \wedge \psi_{i-1}(C'',C')\right).$$

- This gives us the QBF
$$\psi_{M,x} := \psi_{d\cdot S(n)}(C_{\substack{start\\ \text{with } x}}, C_{accept}).$$

- Clearly, this is of size $2^{d\cdot S(n)}$.

How do we improve?

<u>Idea</u>: "Reuse" space for ψ_{i-1}!

Re-define,

- $\psi_i(C,C') := \exists C'' \; \forall D_1 \; \forall D_2$
$$\left[(D_1 = C \wedge D_2 = C'') \vee (D_1 = C'' \wedge D_2 = C') \Rightarrow \psi_{i-1}(D_1, D_2) \right].$$

- Notice that now we get a $\psi_{M,x}$ of size $(\lg 2^{d\cdot S(n)}) \cdot O(S(n)) = O(S(n)^2)$.
- Also, M accepts x iff $\psi_{M,x} \in TQBF$.

$\Rightarrow L \leq_p TQBF$. $\square$

- Lemma 1 & 2 prove:

<u>Theorem</u> (Meyer & Stockmeyer, 1972):
TQBF is Pspace-complete.

The QBF game

- The truth of a QBF $\psi :=$
$\exists x_1 \forall x_2 \exists x_3 \ldots \exists x_{2n-1} \forall x_{2n}\ \phi(x_1, \ldots, x_{2n})$
can be interpreted as a 2-player game.

- Say, Player-1 picks values for $x_1, x_3, \ldots$
& Player-2 " " " $x_2, x_4, \ldots$

- We declare Player-1 the <u>winner</u> iff ϕ is true in the end.
$\Rightarrow$ Deciding $\psi \overset{?}{\in}$ TQBF signifies whether there <u>is</u> a winning strategy for Player-1!
$\Rightarrow$ Suggests the Pspace-hardness of many board games, eg. Chess_n, Go_n, Checkers_n.

- Thus, the question $NP \overset{?}{\neq} PSPACE$ is asking whether "games" are harder than "puzzles"!

- Using the proof of the previous theorem we can also show $Nspace = Pspace$.

<u>Theorem</u> (Savitch 1970): $Nspace(S) \subseteq Space(S^2)$.

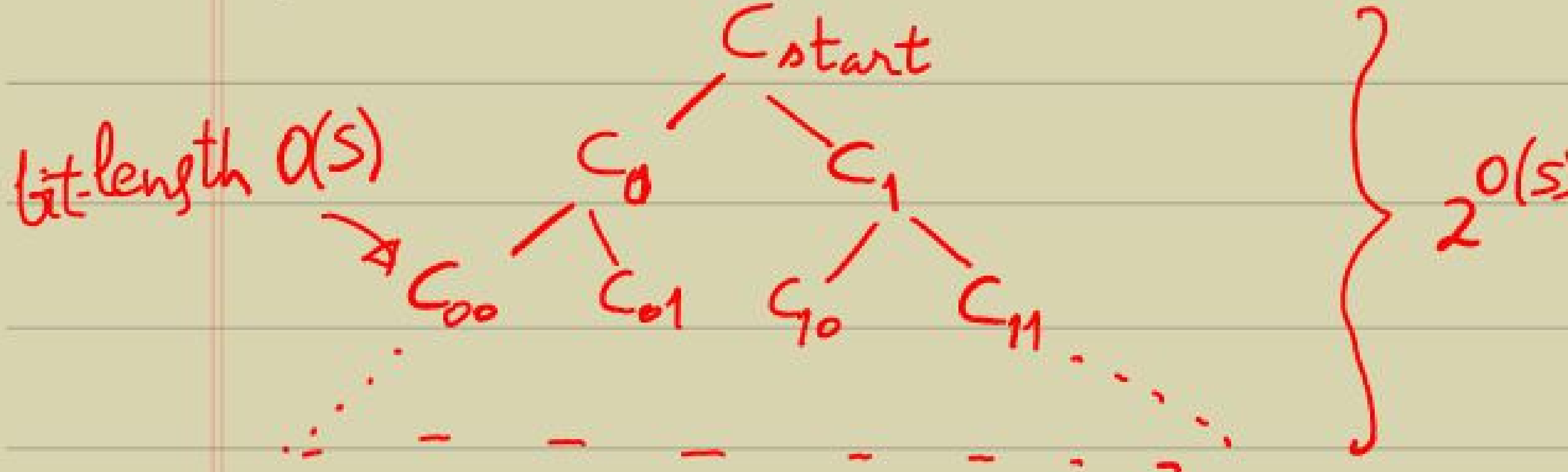
<u>Proof</u>:

- Let M be an $S(n)$-space NDTM deciding $L \in Nspace(S)$.
- As in the proof of Lemma 2, we design a QBF: $\psi_{M,x} = \psi_{d \cdot S(n)}(\underset{\text{with } x}{C_{start}}, C_{accept})$,

with the modification that $\varphi_{M,x}(C, C')$ captures <u>two</u> possible transition steps instead of a unique one.

- $\psi_{M,x}$ remains a QBF of size $O(S^2)$.
- So, easily checkable in $Space(S^2)$. $\square$

- Another way to interpret the previous result is through the configurations tree of the NDTM M:



▷ Each configuration requires $O(S)$ space & specifying a location in the tree requires $\lg 2^{O(S)}$ space.

$\Rightarrow$ Reachability $C_{start} \rightsquigarrow C_{stop}$ can be checked in $O(S^2)$ space.

(Alternately, do it in $2^{O(S)}$ time & space!)

- Reachability also plays an important role in the "small" classes NL, L, ...

NL-completeness

- By the previous discussion
$$\mathbb{L} \subseteq NL \subseteq P.$$

- For a meaningful notion of "hardness" in NL we need $\mathbb{L}$-reductions.

Defn: • We call $f: \{0,1\}^* \to \{0,1\}^*$ implicitly $\mathbb{L}$-computable if

If length is $\geqslant i$ then outputs the i-th bit

$$L_f := \{(x,i) \mid f(x)_i = 1\}, \;\&$$
$$L'_f := \{(x,i) \mid \underbrace{|f(x)|}_{\text{length}} \geqslant i\}$$
are in $\mathbb{L}$.

• We call $A \leq_{\mathbb{L}} B$ if $\exists$ implicitly $\mathbb{L}$-computable f s.t. $\forall x \in \{0,1\}^*$,
$$x \in A \text{ iff } f(x) \in B.$$

• We call B NL-complete if
$$B \in NL, \;\&$$
$$\forall A \in NL, \; A \leq_{\mathbb{L}} B.$$

Proposition: (i) $A \leq_{\mathbb{L}} B \leq_{\mathbb{L}} C \Rightarrow A \leq_{\mathbb{L}} C$.

(ii) $A \leq_{\mathbb{L}} B$ & $B \in \mathbb{L} \Rightarrow A \in \mathbb{L}$.

Proof:

(i) Let f, g be the two implicitly $\mathbb{L}$-computable fns. Observe that $g \circ f$ is also implicitly $\mathbb{L}$-computable.

(ii) Similar. $\square$

- We now see an NL-complete problem.

Defn: Path $:= \{(G, s, t) \mid \exists$ directed path $s \rightsquigarrow t$ in the directed graph $G\}$.

Lemma 1: Path $\in$ NL.

Pf:
- Let (G, s, t) be an input instance.
- Each vertex in G can be identified in $\lg|G|$ space.
- Thus, an NDTM M that simulates a walk in G (with origin s & accepts on reaching t),

has space complexity $O(\log n)$.

$\Rightarrow$ Path $\in$ NL. $\square$

Lemma 2: $\forall A \in NL$, $A \leq_{L} Path$.

Proof:

- Let M be an NDTM deciding A in space $\leq d \cdot \log n$.
- We consider an f that maps an input x of A to $f(x) = (G, s, t)$ where G is the configuration graph of $M(x)$:
  (1) the vertices of G are all the configs. of $M(x)$,
  (2) s & t are the "start" & "accept" configs. respectively,
  (3) the edge (C, C') is there iff $C \to C'$ is a valid transition step of $M(x)$.

- Obviously, $x \in A$ iff $f(x) \in$ Path.

- More importantly, f is implicitly $\mathbb{L}$-computable:

  (1) the list of vertices is $\mathbb{L}$-computable,

  (2) given (C, C') we can check whether its a valid transition of $M(x)$ in $O(|C| + |C'|) = O(\lg |x|)$ space.

  (Basically, scan C, C' & refer the δ-fn. of the TM M.) $\square$

<u>Theorem</u>: Path is NL-complete.

<u>Open</u>: $NL \neq \mathbb{L}$?

<u>Corollary</u>: $\overline{\text{Path}}$ is coNL-complete.

<u>Pf</u>: ◦ Use the same f as before. $\square$

<u>Qn</u>: Is $NL = coNL$?

$\mathsf{NL} = \mathsf{coNL}$

Theorem (Immerman & Szelepcsényi, 1987): $\mathsf{NL} = \mathsf{coNL}$.

Proof:

- It suffices to show $\overline{\text{Path}} \in \mathsf{NL}$.
- I.e. design a logspace-algorithm A s.t. for an input instance (G, s, t), $\exists$ sequence of "guesses" u for A with $A(\langle G, s, t\rangle, u) = 1$ iff $\nexists$ path $s \rightsquigarrow t$ in G.
- Idea: path counting!

- Let $n := |V(G)|$, and C_i be the set of vertices reachable from s in $\leq i$ steps.

$\triangleright$ A logspace-machine can easily certify whether a vertex v is in C_i.

- Now, the plan is to design logspace-machines that could certify:

(i) a vertex $v \notin C_i$, given $|C_i|$, and
(ii) $|C_i| = c$, given c and $|C_{i-1}|$.
(Clearly, this proves $\overline{\text{Path}} \in NL$.)

- Certifying $v \notin C_i$, given $|C_i|$:

The certificate is simply a list of vertices $v_1 < v_2 < \ldots < v_{|C_i|}$ in increasing order, all in C_i, and none equal to v.

To do this the algorithm guesses v_j, v_{j+1} & checks: $v_j < v_{j+1}$ & $v_j, v_{j+1} \in C_i$. Finally, it counts that $|C_i|$ many v_j's were guessed.

(ii) in two steps:

- Certifying $v \notin C_i$, given $|C_{i-1}|$:

The certificate is again a list $v_1 < \ldots < v_{|C_{i-1}|}$ in increasing order, all in C_{i-1}, none equal to v or its neighbour.

Clearly, this certificate is guessable

& checkable by a logspace-machine.

- Certifying $|C_i| = c$, given $|C_{i-1}|$:

For every vertex v, $v \in C_i$ resp. $v \notin C_i$ are certifiable in logspace, given the value $|C_{i-1}|$.

Thus, the machine can go through each vertex v & count the correct value $|C_i|$.

- This finishes the proof of $\overline{\text{Path}} \in NL$. $\square$

Corollary: $\forall S(n) = \Omega(\lg n)$, $\text{Nspace}(S) = \text{coNspace}(S)$.

Proof:

- Let M be a $S(n)$-space NDTM deciding L.

▷ The configuration graph G of $M(x)$ is $S(n)$-space computable (assuming $S(n) > \lg n$ so that x can be read!).

- We have: $x \notin L$ iff $\langle G, C_{start}, C_{accept} \rangle \notin Path$.

- Now, invoke the previous proof to deduce $\bar{L} \in Nspace(S)$. $\square$