Space Complexity

To define space complexity, we need the notion of input/output work tapes.

Def. Let $S : \mathbb{N} \to \mathbb{N}$ be space-constrained. $SPACE(s(n))$ is the class of languages decided by a TM using at most $c \cdot s(n)$ space.

Q. Define $SPACE_k(s(n))$ wrt $k$-tape TMs ($k \leq k'$)

(R) $SPACE_k \subseteq SPACE_{k'}(s(n))$

(6) $SPACE_k(s(n)) = SPACE_{k'}(s(n))$

(B) Depends on values of $k, k'$

Because input head is read-only, we can allow $s(n)$ to be less than $n!$

$\text{PSPACE} = \bigcup_{c>0} SPACE(c^n)$
$\text{NPSPACE} = \bigcup_{c>0} INSPACE(c^n)$

$L = SPACE(\log n)$
$\text{NL} = INSPACE(\log n)$

Always assume $s(n) \geq \log n$
Obviously \text{IP} \subseteq \text{PSPACE}.

But \text{NP} \subseteq \text{PSPACE} \text{ (try all certificates, simply sense space!)}

\text{co-NP} \text{ is also in PSPACE.}

Configuration graphs:

Given \( M \), we can define config. graph:

Remember, we express current state of \( M \) by writing out tape contents \( c \), current state \( q \), and marking out head position.

\( T_0, q_1, \ldots = q_i, q_{i+1}, \ldots = q_e \)

(Augment alphabet with states)

Think of graph with each node with config string and directed edge from \( s_i \) to \( s_j \) if \( M \) moves from \( s_i \) to \( s_j \).

Define accepting config. to be unique. Assume \( M \) clears out tape on accepting state.

Define \( G_{M,x} \).

\( \begin{array}{c}\text{Start} \\
\end{array} \quad \begin{array}{c} \circ \\
\end{array} \quad \begin{array}{c}\text{Accepting config.} \\
\end{array} \)

\( M \text{ accepts } x \iff G_{M,x} \text{ has directed path from Start to Accepting config.} \)
For $s(n)$ time $M$, $G_{M,n}$ has

(R) poly $(s(n))$ nodes
(B) $2^{O(s(n))}$ nodes
(S) $2^{O(s(n))}$ nodes
(b) $2^{\text{poly}(s(n))}$ nodes

Consider outdegree in $G_{M,n}$.
(R) Always at most 1
(B) Always has node with $\text{deg} \geq 1$
(S) Depends.

Claim: $G_{M,n}$ has outdegree 1 iff $M$ is deterministic ($on^2$).

Thm: $\text{SPACE}(s(n)) \subseteq \text{INSPACE}(s(n)) \subseteq \text{DTIME}(2^{O(s(n))})$

Proof: To show $\text{SPACE}(s(n)) \subseteq \text{DTIME}(2^{O(s(n))})$ is easy. Simply run machine $2^{O(s(n))}$ time. If no accept in this time, reject.

More generally, $G_{M,n}$ can be constructed in $2^{O(s(n))}$ time. Checking for paths from source to sink can be done in $2^{O(s(n))}$ time.

Thus: (Savitch's Theorem 70) $\text{INSPACE}(s(n)) \subseteq \text{SPACE}(s(n)^2)$

So $\text{PSPACE} = \text{INSPACE}$!

Proof: Consider $L \in \text{INSPACE}(s(n))$. There is some non-deterministic $s(n)$-space machine $M$ deciding $L$.

Define $G := G_{M,n}$.

It suffices to check if $C_{\text{Start}}$ has path to $C_{\text{Accept}}$.

$G$ has $V = 2^{O(s(n))}$ vertices, so path could be very long.
Define procedure $\text{REACH}(C, C', i)$: 

- YES if there is a path from $C$ to $C'$ of length $\leq 2$.
- NO otherwise.

$\text{REACH}(C_{\text{start}}, C_{\text{acc}}, \log V)$ is true if $M$ accepts $w$.

$\text{REACH}(C_{\text{start}}, C_{\text{acc}}, i) = \exists C \left( \text{REACH}(C_{\text{start}}, C_{\text{acc}}, i-1) \wedge \text{REACH}(C_{\text{start}}, C_{\text{acc}}, i-1) \right)$

Q. Base case $\text{REACH}(C, C', 0)$ can be decided in:

(A) $\text{DTIME}(s(n))$
(B) $\text{NTIME}(s(n))$
(C) $\text{SPACE}(s(n))$
(B) $\text{NSPACE}(s(n))$

Thus, we have recursion tree. At every step, simply enumerate over all possible configurations for choice of $C$.

Q. What is depth of recursion tree?

(A) $O(s(n))$
(B) $\text{poly}(s(n))$
(C) $2^{O(s(n))}$

Q. What is size of recursion tree

(A) $O(2^{s(n)})$
(B) $2^{O(s(n))}$
(C) $\text{poly}2^{\text{poly}(s(n))}$

The space required is only the depth of the recursion tree, times storage of each configuration. Thus, this can be solved in $\text{DTIME DSPACE}(s(n))$. 