Savitch's theorem

$S : \mathbb{N} \to \mathbb{N}$ with $S(n) \geq \log n \quad \forall n$

$$NSPACE(S(n)) \subseteq DSPACE(S(n)^2)$$

Proof: $L \in NSPACE(S(n))$

M : NDTM M deciding L in space $S(n)$.

Task: determining whether $x \in L$?

$G_{M,x}$: configuration graph of M on i/p x.

REACH(u, v, i) : returns "YES" if $\exists$ a path from u to v of length at most 2^i
returns "NO" otherwise.

$x \in L \iff$ REACH$(C_{start}, C_{accept}, \log N)$

vertices in $G_{M,x}$: $N = 2^{O(S(n))}$

can be computed deterministically using $O(S(n)^2)$ - space

REACH (u, v, i)

for all vertices y

if REACH$(u, y, i-1)$ & REACH$(y, v, i-1)$ both return "YES"

then return "YES".

return "NO".

$u \overset{2^i}{\rightsquigarrow} v$

$u \overset{2^{i-1}}{\rightsquigarrow} y \overset{2^{i-1}}{\rightsquigarrow} v$

Space required for REACH(u, v, i) = $s_{N,i}$

$s_{N,i} = 2 s_{N,i-1} + \log N$

$s_{N,i-1}$

space used for the recursive call REACH$(u, y, i-1)$ can be re-used for REACH$(y, v, i-1)$

$s_{N,i} = s_{N,i-1} + \log N$

$s_{N,\log N} = O(\log^2 N) = O(S(n)^2)$

Corollary: $NSPACE(n^c) \subseteq DSPACE(n^{2c})$

$\Rightarrow$ $NPSPACE \subseteq PSPACE$

$\Rightarrow$ $PSPACE = NPSPACE$

PSPACE Completeness

A language B is PSPACE-hard if for every $A \in PSPACE$, $A \leq_p B$.

If B is PSPACE-hard & $B \in PSPACE$, then B is PSPACE-Complete

TQBF True Quantified Boolean formula
= set of all true quantified Boolean formulae

TQBF is PSPACE-Complete.

Quantified Boolean formula

$$Q_1 x_1 Q_2 x_2 \cdots Q_n x_n \; \phi(x_1, x_2, \ldots, x_n)$$

→ prenex normal form (all quantifiers appear before the formula)

Q_i : $\exists$ or $\forall$

ϕ: Boolean formula in variables $x_1, x_2, \ldots, x_n$

When all variables are quantified, formula is either true or false.

E.g.

$\psi_1 = \forall x \exists y \; (x \vee y) \wedge (\neg x \vee \neg y)$ TRUE

$\phi_1 \in TQBF$

$x = 0, y = 1$

$x = 1, y = 0$

$\psi_2 = \exists x \forall y \; (x \vee y) \wedge (\neg x \vee \neg y)$ FALSE

$\phi_2 \notin TQBF$.

TQBF is PSPACE-Complete

I TQBF is in PSPACE

$$\Psi = Q_1 x_1 Q_2 x_2 \cdots Q_n x_n \; \phi(x_1, x_2, \ldots, x_n)$$

Size of ϕ : m

$\Psi|_{x_i=b}$: all occurences of x_i in ϕ replaced with b.

IS_TRUE(Ψ, i)

if $Q_i = \forall$, then

return "YES" if IS_TRUE($\Psi|_{x_i=0}$, $i+1$) and IS_TRUE($\Psi|_{x_i=1}$, $i+1$)

if $Q_i = \exists$ then

return "YES" if IS_TRUE($\Psi|_{x_i=0}$, $i+1$) or IS_TRUE($\Psi|_{x_i=1}$, $i+1$)

Ψ is true iff IS_TRUE(Ψ, 1) = "YES"

$s_{n,m}$: space required for IS_TRUE

Writing down $\psi|_{x_i=0}$ requires $O(m)$ space.

$$s_{n,m} = s_{n-1,m} + O(m)$$

$$= O(n \cdot m) \quad : \text{polynomial in } n, m$$

$\therefore$ TQBF $\in$ PSPACE

II $\forall$ L $\in$ PSPACE, $L \leq_p$ TQBF

L $\in$ PSPACE, M : TM deciding L in $S(n)$ space.
S : polynomial.

$x \longmapsto \psi$ s.t. $x \in L$ iff $\psi \in$ TQBF

$G_{M,x}$: configuration graph of M on i/p x.

$m = O(S(n))$: no. of bits to represent a vertex of $G_{M,x}$

We construct a formula ψ' inductively

$$\psi_0 = \phi_{M,x}, \psi_1, \psi_2, \ldots, \psi_m = \psi'$$

ψ_i defined in terms of ψ_{i-1} $\forall i = 1, \ldots, m$.

$\psi_i(C, C') = 1$ iff $\exists$ a path of length at most 2^i from C to C' in $G_{M,x}$.

$$\psi_i(C, C') = \exists C'' \; \psi_{i-1}(C, C'') \wedge \psi_{i-1}(C'', C') \quad \text{X}$$

size of the formula becomes exponentially large

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

Only one occurrence of ψ_{i-1} ensures that size of formula does not blow-up.

$$size(\psi_i) = size(\psi_{i-1}) + O(m)$$

$$size(\psi') = size(\psi_m) = O(m^2).$$

$$\psi = \psi'(C_{start}, C_{accept}) \qquad |\psi| = O(s(n)^2)$$

↓ can be constructed in poly-time

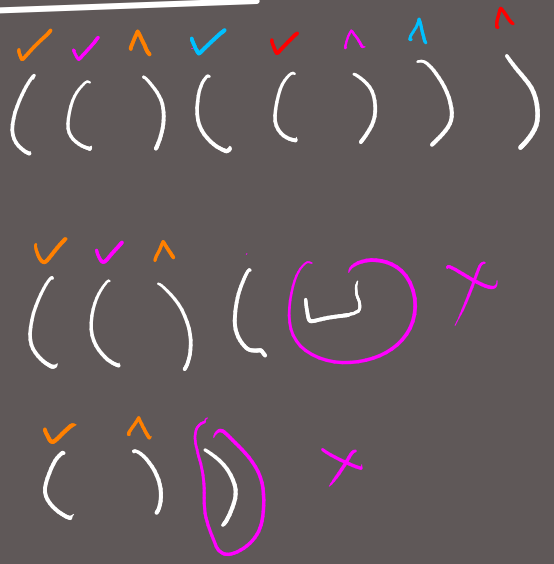
ψ is true i.e., $\psi \in$ TQBF iff M accepts x.

i.e., $x \in L$.

Log-space Computation

Problems in L ?

Balanced parantheses



Space required
$O(n)$
n: length of i/p string.

Can be solved in $O(\log n)$ space.

Maintain a counter k. (initialised 0).

On seeing '(' k++

On seeing ')' k--

Reject when k=0 & you encounter ')'

Accept when entire i/p is read & k=0.

$k \leq n$ No. of cells req. to store $k = O(\log n)$.