Thm $\overline{PATH} \in NL$

$\overline{PATH} \simeq \{\langle G,s,t\rangle : G$ does not have a path from s to $t\}$

Cor $NL = coNL$ complements of languages in NL

Cor For any space bound $S(n) \geq \log_2 n$

$NSPACE(S(n))$ is closed under complement

Proof Imagine that we have the value

Count = # of vertices of G reachable from s

NoPath(s, t, Count, i)

Reach ← 0

For all vertices $v \neq t$, $v \in G$

Guess whether v is reachable from s

if Guess is yes then

Guess & verify a path of length $\leq$ ~~n~~ i

from s to v, one vertex at a time

if path found Reach ← Reach+1

else reject

end for

if reach=count then accept

else reject

≥ Reach paths from s to vertices other than t

If ≥ Count such paths, t is not reachable

How do we compute Count?

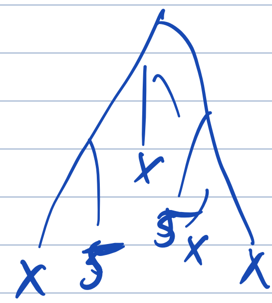
Idea: "inductive counting"

Define: Count_i = # of vertices reachable from s via paths of length $\leq i$

$\therefore$ $\text{Count}_0 = 1$ $\{s\}$

$\text{Count} = \text{Count}_n$

This will be via a nondeterministic algorithm: such an alg. will have some paths that reject but any branch that does not reject will compute the correct value.



We can't afford to store all the Count_i vars but we only need vars for the current & next i, counti, countiplus1

$i \leftarrow 0$, counti $\leftarrow 1$

for $i = 0$ to $n-1$ do

    countiplus1 $\leftarrow 0$

    for all vertices $v \in G$ do

        if $v = s$ then

            countiplus1 $\leftarrow$ countiplus1 + 1

        else

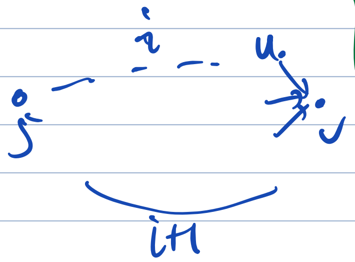
            Guess whether v is reachable from s via a path of length $\leq i+1$

            If guess for v is yes

                Guess & verify a path of length $\leq i+1$ from s to v, one vertex at a time

                if found then countiplus1 $\leftarrow$ countiplus1+1

                else reject



If guess for v is no
    For all predecessors u of v in G
        "Check that no path of length $\leq i$ from s to u in G"
        if NoPath(s, t, count$_i$, i) is false
            then reject
    end for
end if
end for
Count $\Leftarrow$ count$_i$ plus 1

Clearly only a constant # of counters and vertices need to be stored $\therefore O(\log n)$ space $\blacksquare$

Cor $NL = coNL$

Pf Let $A \in NL$. Then $A \leq^L_m PATH$

so $\overline{A} \leq^L_m \overline{PATH}$ & Since $\overline{PATH} \in NL$

By properties of $\leq^L_m$ we have

$\overline{A} \in NL \therefore A \in coNL$.

$\therefore NL \subseteq coNL$ and since $\overline{\overline{A}} = A$

we have $NL = coNL$ $\blacksquare$

Cor $NSPACE(S(n))$ is closed under complement

Pf $NSPACE(S(n))$ corresponds to reachability in $2^{O(S(n))}$ size graph. Same ideas hold $\blacksquare$

To consider even simpler complexity classes than L we use circuits not TMs

"NC" stands for "Nick's Class"

<u>Def^n</u> Let $k \geq 0$ be an integer.
Let NC^k be the set of functions computable by ordinary fan-in ≤ 2 $\wedge, \vee, \neg$ gates of polynomial size and $O(\log^k n)$ depth.

Note: if circuit has depth d then its size is $\leq 2^d$

$\therefore$ $O(\log n)$ depth automatically implies $n^{O(1)}$ size
"NC^1"

Fact: Any size S Boolean formula can be rebalanced to have $O(\log S)$ depth and so

Computable by polysize formulas $\Rightarrow$ in NC^1

eg. $x_1 \oplus x_2 \oplus \cdots \oplus x_n \in NC^1$ | Also $x_1 \vee \cdots \vee x_n \in NC^1$

"Parity"

see the circuit we gave before

For circuits there is a different circuit for every input length. but we can relate circuit classes to TM classes by requiring that the circuit for inputs of length n be computable in $O(\log n)$ space from 1^n.

With this requirement we have

$$NC^1 \subseteq L$$

AC^k set of functions computable by polynomial size circuits with unbounded fan-in $\wedge$, $\vee$ gates plus $\neg$ of depth $O(\log^k n)$

$$NC^k \subseteq AC^k \subseteq NC^{k+1}$$

since we just expand the unbounded fan-in gates into trees

AC^0 = functions computable by poly size circuits of unbounded fan-in $\wedge, \vee, \neg$ gates of $O(1)$ (constant) depth

eg. Integer addition $\in AC^0$

$$\begin{array}{rccccc} & c_n & c_{n-1} & & c_2 & c_1 \\ \text{Given} & & x_{n-1} & x_{n-2} & \cdots & x_0 \\ + & & y_{n-1} & y_{n-2} & \cdots & y_0 \\ \hline \end{array}$$

We want to compute the sum $z_n z_{n-1} \cdots z_0$

This would be easy if we had the carries $c_n c_{n-1} \cdots c_1$

since $z_i = x_i \oplus y_i \oplus c_i$

Define two vectors:

$g_{n-1} \cdots g_0$, $p_{n-1} \cdots p_1$

where $g_i = x_i \wedge y_i$ g stands for generate

and $p_i = x_i \vee y_i$ p stands for propagate

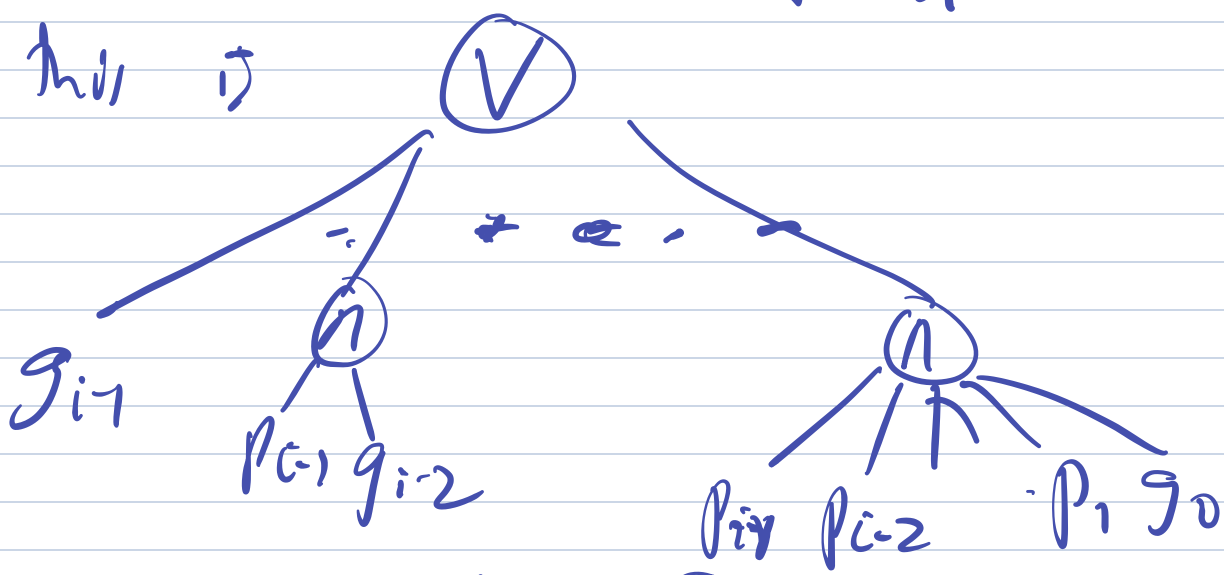
Note that $c_i = 1$ iff there is some $j < i$ s.t. $g_j = 1$ and every position between j and i propagate that carry.

That is:

$$c_i = g_{i-1} + p_{i-1}g_{i-2} + p_{i-1}p_{i-2}g_{i-3} + p_{i-1}p_{i-2}p_{i-3}g_{i-4} + \dots + p_{i-1}p_{i-2}\cdots p_1 g_0$$

where we write this using sum of product form

this is



which is depth 2.

Total depth is constant.

Fact: Parity $\notin AC^0$

Cor: Integer Multiplication $\notin AC^0$