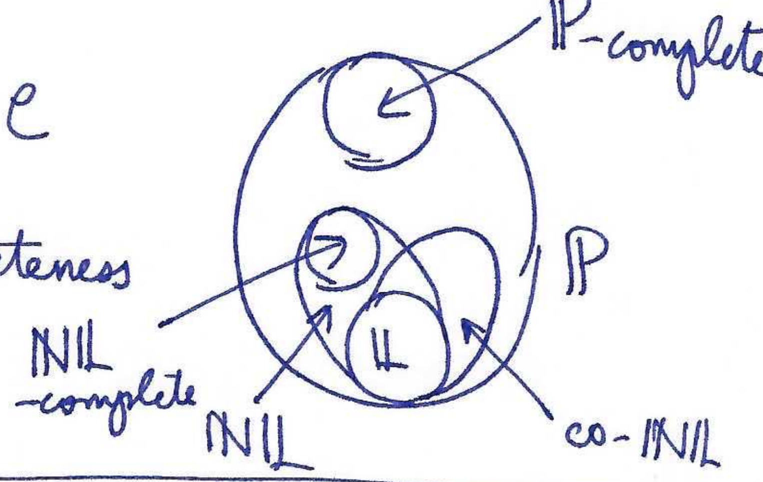
Def: A language C is $\mathbb{NL}$-complete if

(1) $C \in \mathbb{NL}$

(2) $\forall\ B \in \mathbb{NL}\quad B \leq_L C$

(We can also define $\mathbb{P}$-completeness using logspace reductions.)



$$\mathcal{L} \leq_p \mathcal{L}' \quad \mathcal{L}' \leq_p \mathcal{L}'' \Rightarrow \mathcal{L} \leq_p \mathcal{L}''$$ (closure of polynomials)

$$\boxed{\text{Thm: } C \leq_L C',\ C' \leq_L C'' \Rightarrow C \leq_L C''}$$ Logspace reductions can be chained.

Proof: Let f reduce C to C' and g reduce C' to C''

f is implicitly logspace computable by logspace machine M_f

g " " " " " " M_g.

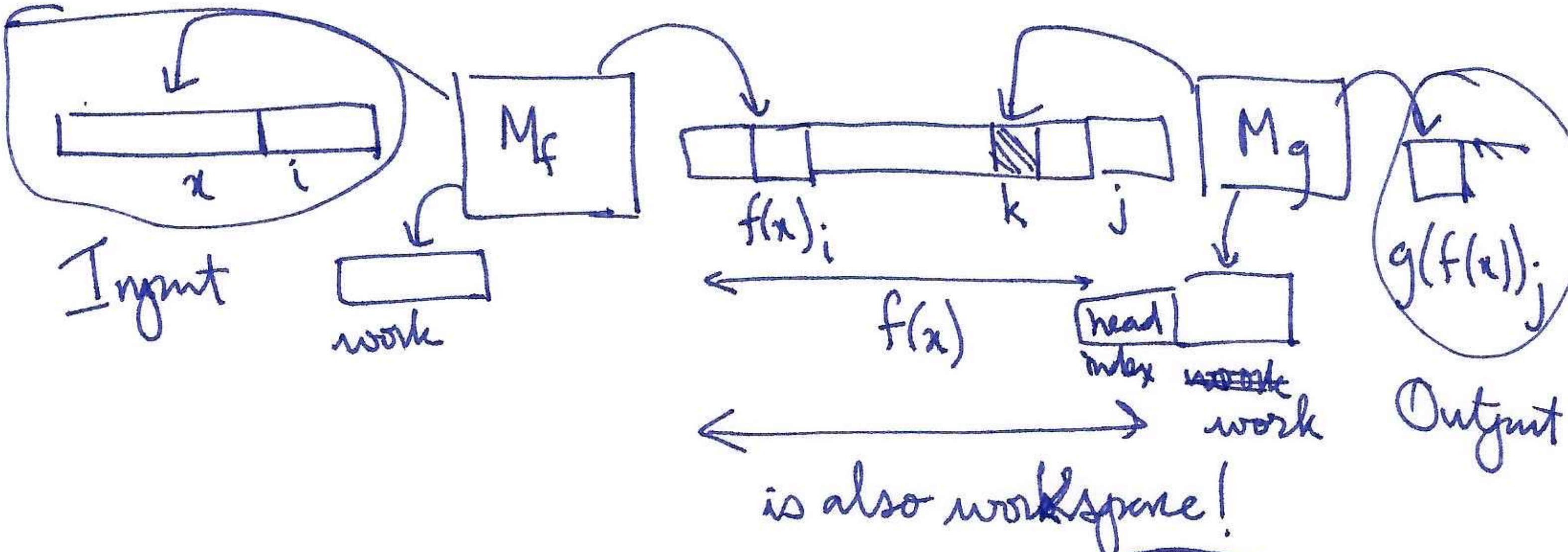
We want to construct reduction h that is imp. log comp. (from C to C'')

$h(x) = g(f(x))$

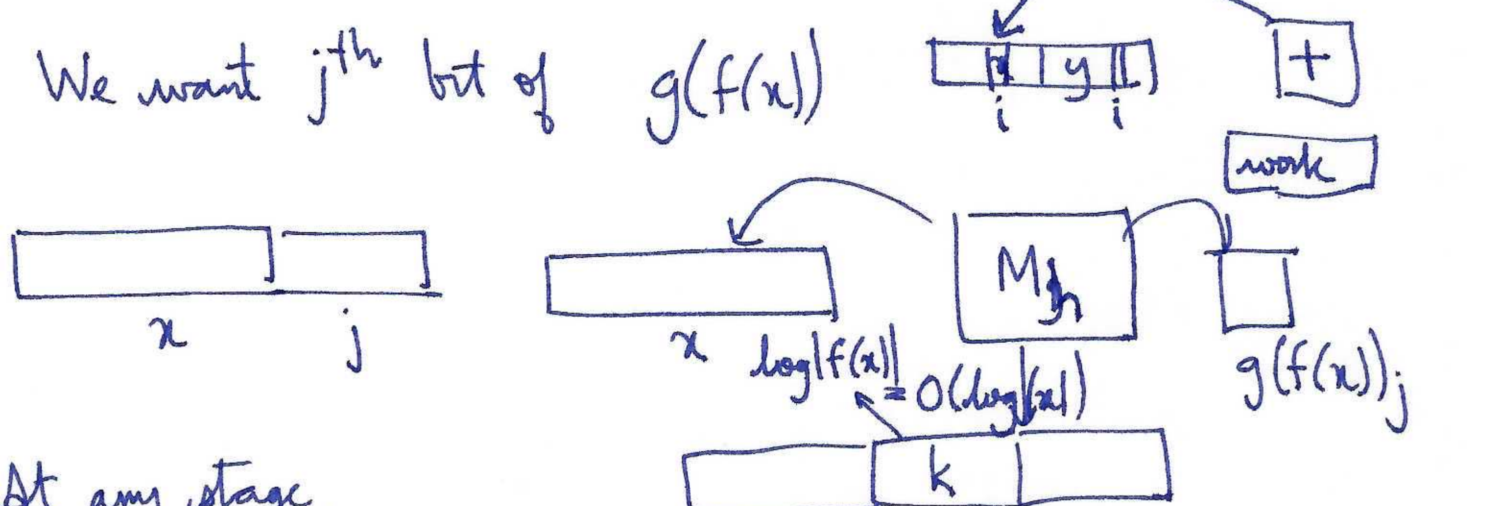
(Failed proof) ~~Use M~~ Let's start with x. $\rightarrow |f(x)|$ bits polynomial!

M_f can $\boxed{\text{write down } f(x)}$ in logspace, by taking input $\langle x, i\rangle$ (for all $i \leq |x|^c$).

M_g can write down the j^{th} bit of $g(f(x))$. ~~using M_g~~.



We want j^{th} bit of $g(f(x))$



At any stage,
k has some value
and M_h is simulating
M_g.

workspace of M_f $\log|x|$

index of "head"/ $f(x)$ that "head" reads

workspace of M_g $\log|f(x)| = O(\log|x|)$

(1) M_h runs M_f on $\langle x, k\rangle$ to get $f(x)_k$

(2) Based on $f(x)_k$ and state of M_g (and workspace of M_g), determine next state of M_g and whether head (of M_g's input) moves $\leftarrow / \rightarrow$.

(3) If $\leftarrow$, decrement k, else increment.

This is a logspace simulation of M_g on input $f(x)$.

k can be stored in logspace.

D~~IR~~PATH $= \{ \langle G, s, t \rangle \mid$ G is a directed graph,
PATH s,t are vertices and $\exists$ path from s to t $\}$

<u>Thm</u>: PATH is NL-complete.

<u>Proof</u>: (1) Prove that PATH $\in$ NL.

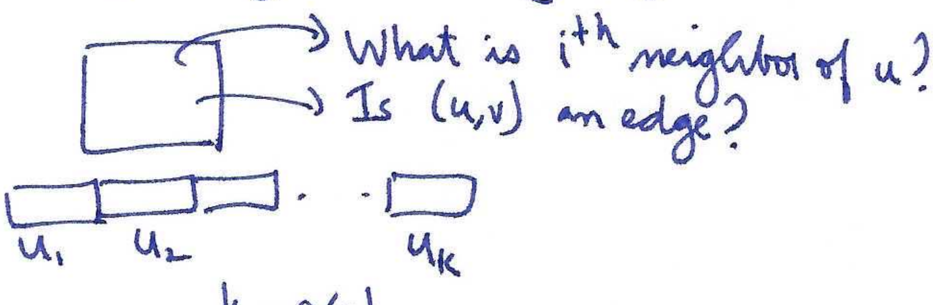
What is a logspace machine in the context of graph algorithms?

Imagine G is adjacency matrix (n vertices).

In $O(\log n)$ space, we can store a vertex.

Meaning, a logspace machine can only store $O(1)$ vertices (vertex labels).

It can look up neighbors (using input and counting)



$k = O(1)$

PATH $\in$ NL. Because machine can non-deterministically guess the next ~~p~~ vertex in path from s to t.

(2) To prove $\forall \mathcal{C} \in \mathsf{NL}$, $\mathcal{C} \leq_L$ PATH

Let M be an NL machine deciding $\mathcal{C}$.

M uses $\leq \boxed{c \log n}$ space (c is constant).

M accepts (input) x iff in config. graph $G_{M,x}$
C_{start} can reach C_{Acc}

M accepts x iff $\langle G_{M,x}, C_{Start}, C_{Acc} \rangle \in$ PATH

Need to argue that given x $\langle \underline{G_{M,x}}, \underline{C_{Start}}, \underline{C_{Acc}} \rangle$
can be implicitly computed in logspace.

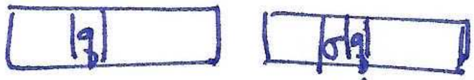
Compute:

1) i^{th} bit of C_{Acc} : easy (doesnt depend on x) $C_{Acc} = q_{Acc} \sqcup \sqcup \sqcup \sqcup$

2) i^{th} bit of C_{Start} : $C_{Start} = \underbrace{q_0}_{\text{fixed}}\boxed{\quad x \quad}$ i^{th} bit of C_{Start}
$i - \Theta(1)$ bit of x

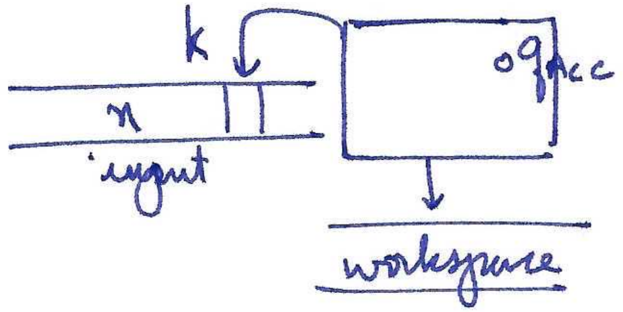
3) $(i,j)^{th}$ bit of ~~C_{Start}~~ $G_{M,x}$: C, C' are config. $|C|, |C'| = O(\log n)$
$(C, C')^{th}$ bit

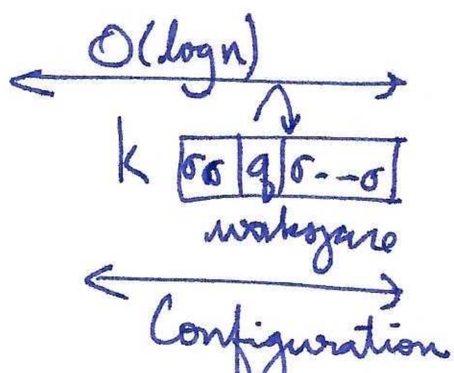
Can we determine if C' follows C (acc to NTM transitions of M) in logspace? Yes



C C'

Check that head moves correctly, and all other symbols are unchanged. (not touched by head)





$L \in NL$ M logspace NTM deciding L

$x \in L$?   Write out all config of M (fixing x)

At most poly(n). There are poly(n) vertices in $G_{M,x}$

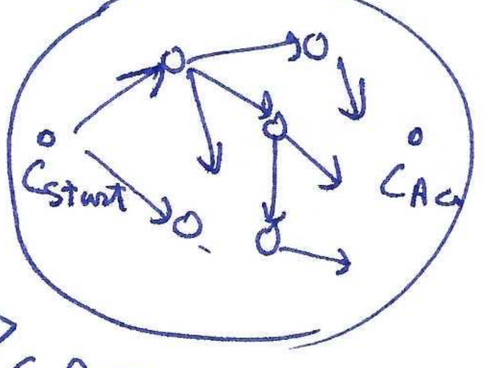
C_{Start} $\underset{k}{\boxed{0}}\ \boxed{q_0 \sqcup\sqcup\sqcup\sqcup\sqcup}$

C_{Acc} $\underset{k}{\boxed{0}}\ \boxed{q_{Acc}\,\sqcup\sqcup\sqcup\sqcup\sqcup}$

$x \in L$

iff

$\langle G_{M,x}, C_{Start}, C_{Acc} \rangle \in PATH$



$f(x) = \langle G_{M,x}, C_{Start}, C_{Acc} \rangle$

PATH is NL-complete.

UPATH = $\{ \langle G, s, t \rangle \mid$ G is a undirected graph and s is connected to t$\}$

USTcon

[Reingold 05] UPATH $\in$ L.