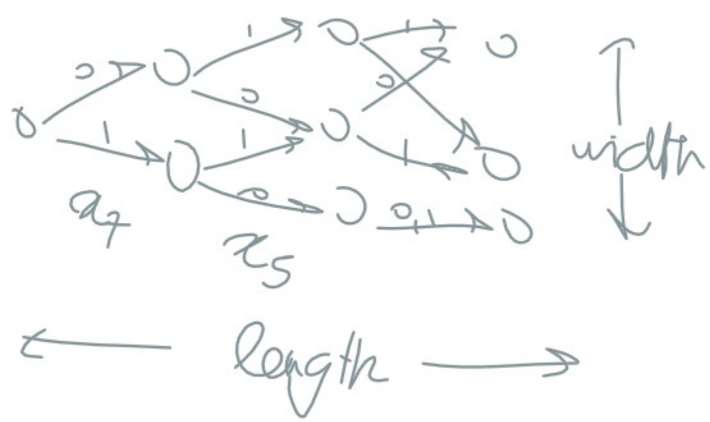
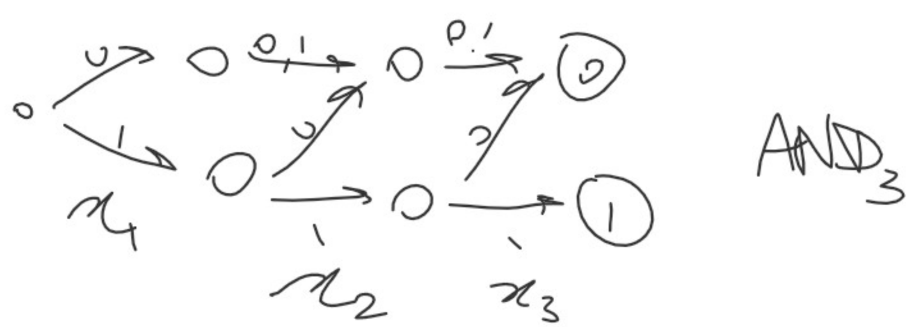
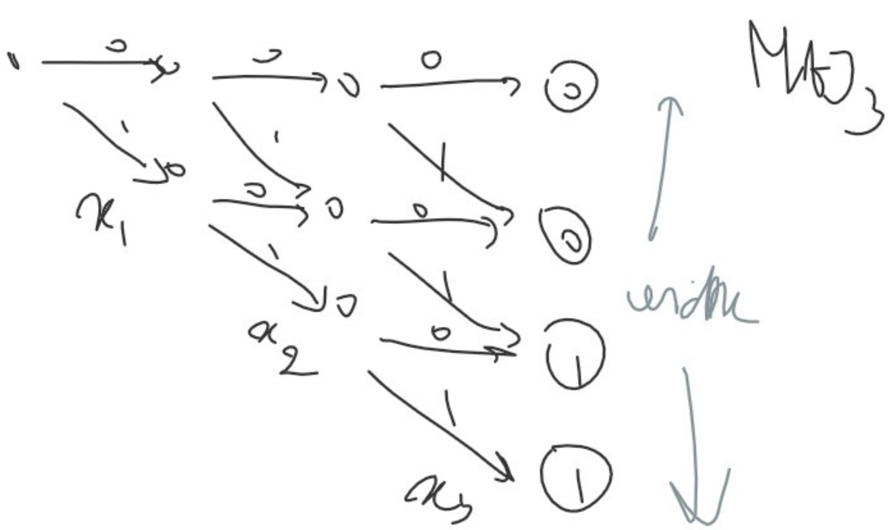
Barrington's theorem

Thursday, October 13, 2022 13:52

Branching programs





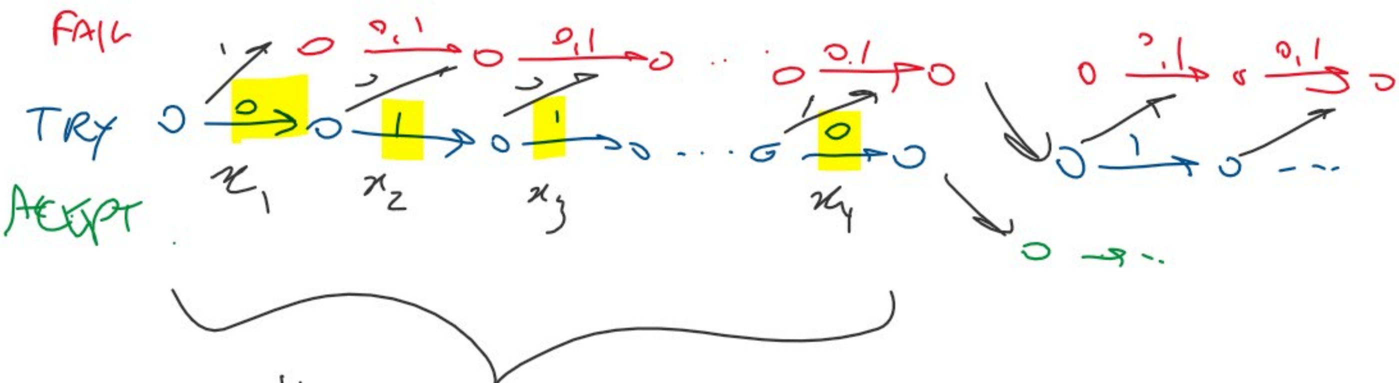


Barrington's

Thm MAJ_n has BPs of

- $O(1)$ width WTF?
- poly(n) length

Actually, everything has $O(1)$ width



this part accepts iff $x = 011 \cdots 0$

Just repeat for each y st. $f(y) = 1$.

NOT about "it's in this state",
about "it's in this state at this time step."

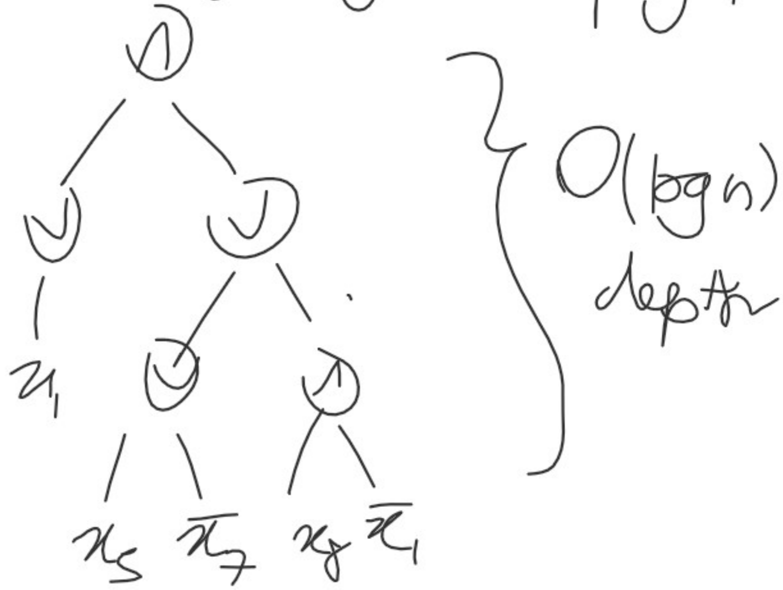
about "it's in this state at this time step".

hide memory in time counter!

NC^1

Actually works for any $f \in NC^1$.

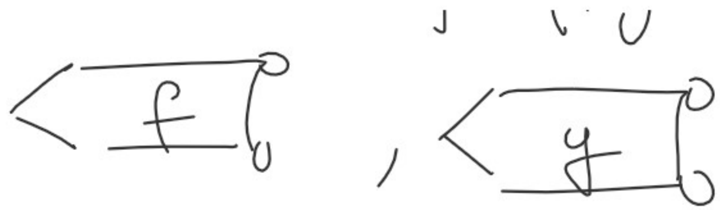
size := # gates = poly(n)



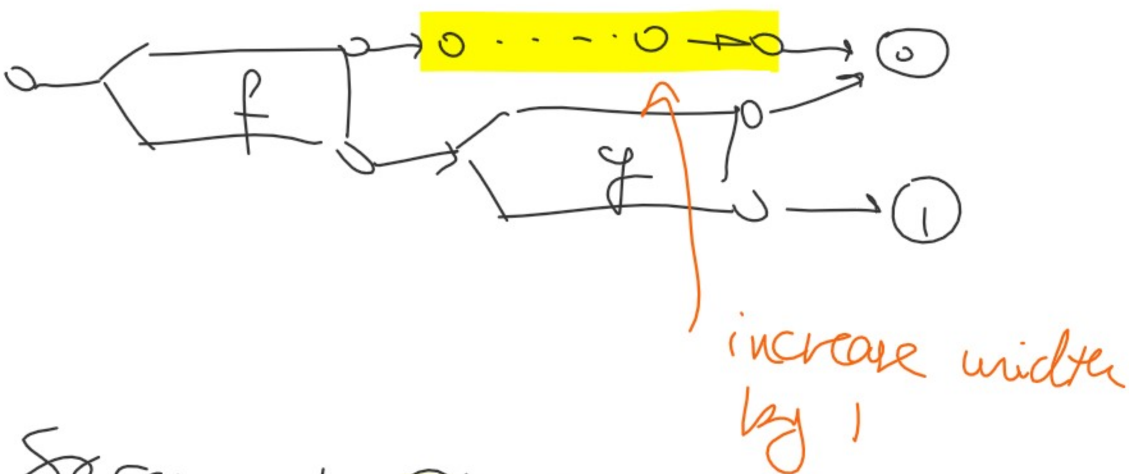
Exercise: prove that MAJ $\in NC^1$.

Attempt #1

Suppose have BPs for f, g



Then can do $f \cap g$ like this



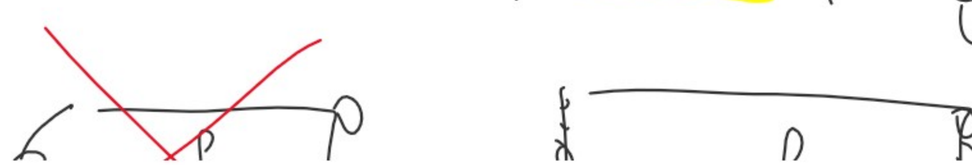
So can get O(log n) width, poly-length

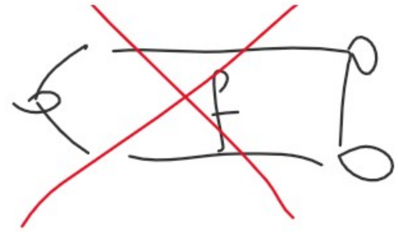
already surprising for MAJ!

Exercise 2 show how to do this via Chinese remainder

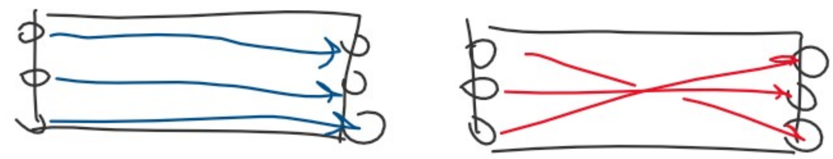
Key insight

Don't think about behavior from one node, think about the transformations the segment applies





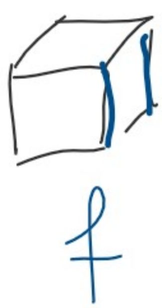
e.g.



if $f(x)=0$ if $f(x)=1$

Proof

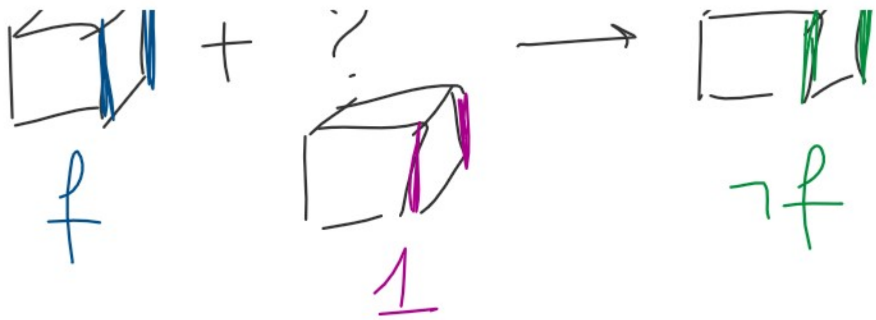
We'll work with ==width 8== (8 possible "states")

"not"  : a BP that

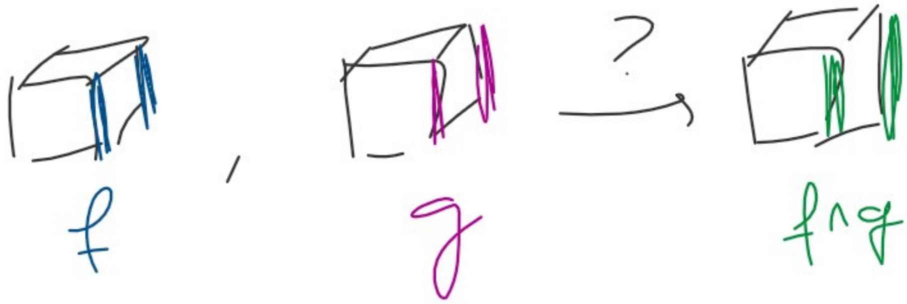
- if $f=0$, does nothing
- if $f=1$, swaps the states given by the edges

We'll concatenate these BPs together to obtain more complex behaviour.

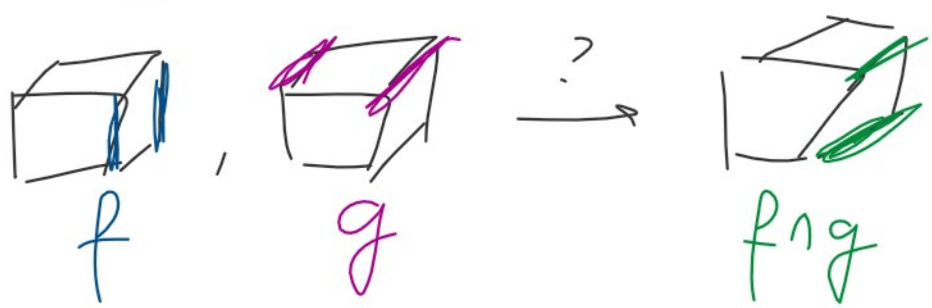
Warmup : $\neg$



Real deal : ∧



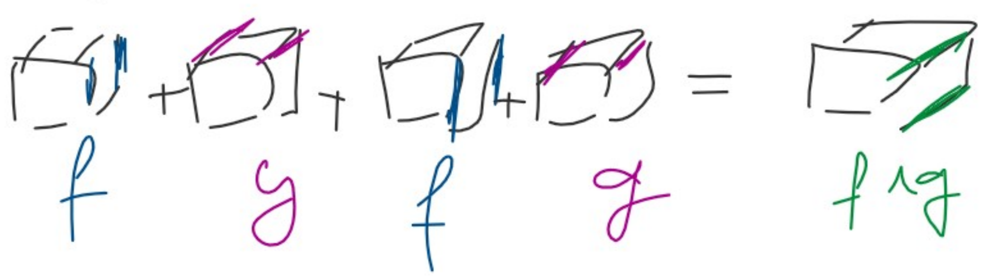
Hint #1 :



Hint #2 :

but...

Answer :



If either $f=0$ or $g=1$, the whole thing cancels out whole thing cancels out

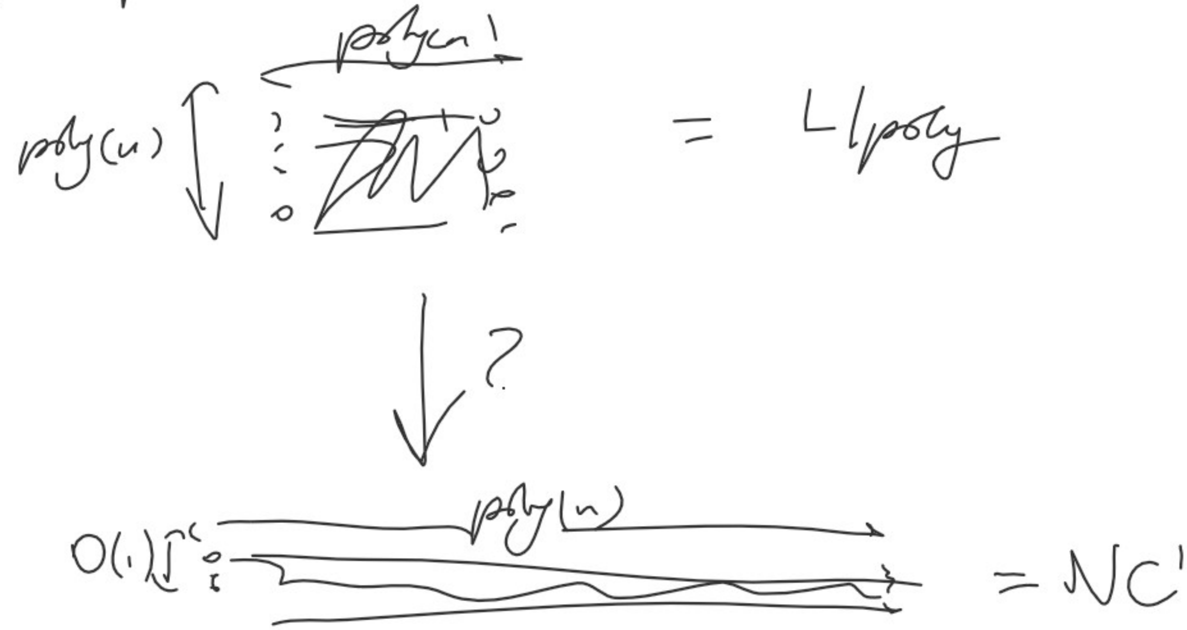
Finishing up

- can get V from 1, 7
- blow up length by 4 by width constant

$\rightarrow$ length $O(4^d) = O(4^{O(\log n)}) = \mathrm{poly}(n)$

Conclusion

Since length is poly(n), we were able to hide $O(\log n)$ bits of info in program counter. But is this always possible?



Would be huge!