Theorem: dirPath is **NL**-complete

Let $A \in$ **NL**, decided by $c \cdot \log n$ space-bounded NTM M

Input: $\underline{w}$; output: dir.Graph G and vertices s,t such that:

M accepts $\underline{w}$ $\Leftrightarrow$ there is a path in G from s to t

$G=(V,E)$, V:=all configurations of M of size $c \cdot \log |\underline{w}|$

$(K_1,K_2) \in E$ $:\Leftrightarrow$ K_2 is a successor config of K_1

s:=start config of M on $\underline{w}$; t:=accept.config (wlog unique)

- M accepts $\underline{w}$ $\Leftrightarrow$ there is a path in G from s to t √
- How large is G? Constructible in logarithmic space?

qed

Immerman-Szelepcsényi

TECHNISCHE UNIVERSITÄT DARMSTADT

Complexity Theory

L = NL ? **=coNL ?** **= P ?** Compare „**P** vs. **NP**"

L vs. **NL**: **NL**-complete dirGraph, 2unSAT, nonBipartite

NL vs. **P**: **P**-complete problems

(probably do not admit an efficient parallelization)

NL vs. **coNL**: solved in 1987,

ACM Gödel Prize 1995 !

Theorem (Neil Immerman,

Róbert Szelepcsényi): NL=coNL

Proof: Show dirGraph$\in$ **coNL**

Theorem: For constructible $s(n)$,

NSPACE$(s(n))$=**coNSPACE**$(s(n))$.

dirGraph$\in$ coNL

TECHNISCHE UNIVERSITÄT DARMSTADT
Complexity Theory

Given $G=(V,E)$, $s,t\in V=\{1,\ldots,m\}$. Goal:
Logspace NTM accepts iff t not reachable from s.
$A_i := \{ v\in V :$ exists path in G of length $\leq i$ from s to $v \}$,
$c_i := \#A_i$, $i=0,\ldots,m-1$. $A_0=\{s\}$, $c_0=1$. Accept iff $t\notin A_{m-1}$

Def: NTM computes (partial multivalued) $f:\subseteq\Sigma^*\rightrightarrows\Sigma^*$ iff

- $\forall$ inputs $\underline{x}\in$ dom(f) there is an accepting computation.
- Every accepting computation outputs some $y\in f(\underline{x})$.

FNL is closed under composition! (proof?)

Lemma: For each i, $A_i\in$ **NL**.
Given (!) c_i, logspace NTM can even <u>enumerate</u> A_i:

- For each $v\in V$, 'guess' whether $v\in A_i$ (1) or not (0)
- If guessed 1: output, verify (**NL**) and increase counter
- In the end, accept iff counter$=c_i$!!!

dirGraph∈ coNL

Given $G=(V,E)$, $s,t\in V=\{1,\ldots,m\}$. Goal:
Logspace NTM accepts iff t not reachable from s.
$A_i := \{ v\in V : \text{exists path in } G \text{ of length } \leq i \text{ from } s \text{ to } v \}$,
$c_i := \#A_i$, $i=0,\ldots,m\text{-}1$. $A_0=\{s\}$, $c_0=1$. Accept iff $t\notin A_{m-1}$

Lemma: For each i, $A_i\in$ **NL**.
Given (!) c_i, logspace NTM can even enumerate A_i.

Lemma: Given (!) c_i, $A_{i+1}\in$ **coNL**:
Enumerate A_i and, if no edge to v found, accept.

Lemma: Given c_i, logspace NTM can compute c_{i+1}:
For each v, 'guess' whether $v\in A_{i+1}$ holds, and verify

Proof (Theorem): $1=c_0 \to c_1 \to c_2 \to c_3 \to \ldots \to c_{m-1}$

NL and Parallel Computation

Every problem in **NL** can be solved in parallel time $O(\log^2 n)$ by *polynomially many processors*!

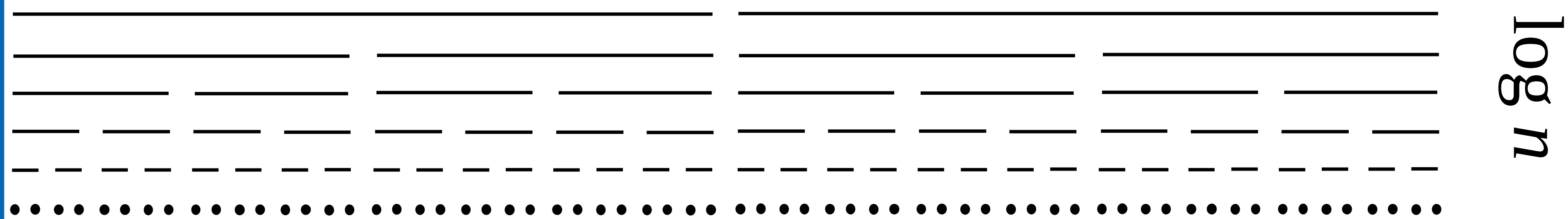
- dirPath $\leqslant_L$ Boolean Matrix powering:
  - $G=(V,E) \to$ adjacency matrix $A \in \{0,1\}^{V \times V}$ of G:
  - $A_{u,v}>0 \;:\Leftrightarrow\; v$ reachable from u in ≤ 1 step
  - $(A^k)_{u,v}>0 \Leftrightarrow v$ reachable from $u \in V$ in $\leq k$ steps
- goal: $(A^k)_{s,t}$ for some $k \geq |V| =: n$.
  - rept.squaring: $A \to A^2 \to A^4 \to A^8 \to \ldots$: $O(\log n)$
  - each phase = matrix multipl.; n^2 dot products
  - each dot product in parallel time $O(\log n)$

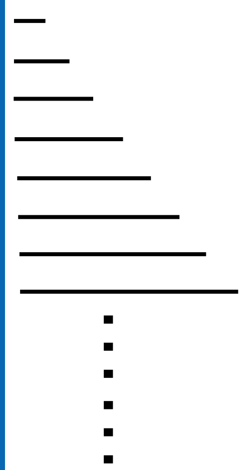
More Parallel Algorithms

Prefix Sum: Given $(x_1,\ldots,x_n)$, calculate <u>all</u> sums

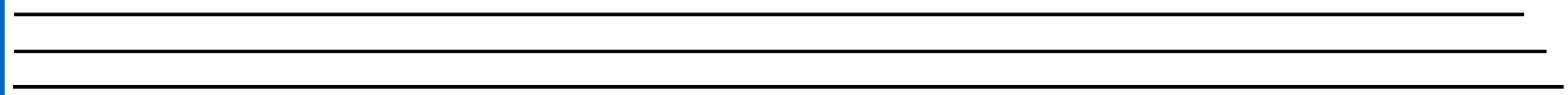
$$x_1,\ x_1\oplus x_2,\ x_1\oplus x_2\oplus x_3,\ \ldots,\ x_1\oplus x_2\oplus\ldots\oplus x_{n-1},\ x_1\oplus x_2\oplus\ldots\oplus x_{n-1}\oplus x_n$$

in logarithmic parallel time √ using $O(n\cdot\log n)$ gates





for any <u>associative</u> operation ⊕



Carry Look-Ahead Adder

Prefix Sum: Given $(x_1,\ldots,x_n)$, calculate <u>all</u> sums

$$x_1,\ x_1 \oplus x_2,\ x_1 \oplus x_2 \oplus x_3,\ \ldots,\ x_1 \oplus x_2 \oplus \ldots \oplus x_{n-1},\ x_1 \oplus x_2 \oplus \ldots \oplus x_{n-1} \oplus x_n$$

in parallel time O(log *n*) for any <u>associative</u> operation $\oplus$

Long Addition: Given $(a_0,\ldots,a_{n-1})$ and $(b_0,\ldots,b_{n-1})$, calculate $(c_0,\ldots,c_{n-1},c_n) := (a_0,\ldots,a_{n-1}) + (b_0,\ldots,b_{n-1})$ in logarithmic parallel time? *ripple-carry adder*

i-th carry $z_i = g_i \vee (p_i \wedge z_{i-1})$ *'generate', 'propagate'*

where $g_i := a_i \wedge b_i$

and $p_i := a_i \vee b_i$

$$(z_i,0) = (z_{i-1},0) \otimes (g_i, p_i)$$

$$= ((z_{i-2},0) \otimes (g_{i-1}, p_{i-1})) \otimes (g_i, p_i)$$

$$(g,p) \otimes (g',p') := (\ g' \vee (p' \wedge g),\ p' \wedge p\ )$$ associative!