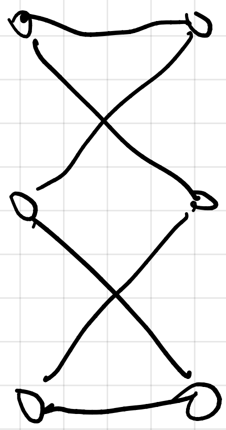
$\boxed{\text{13 Sep 2023}}$ Parallel Algorithms, Part I

Recap: G B substitute random # for each x_{ij}

Lovasz's Algorithm



$$\Longrightarrow \begin{bmatrix} x_{11} & x_{12} & 0 \\ x_{21} & 0 & x_{23} \\ 0 & x_{32} & x_{33} \end{bmatrix}$$

$$\Downarrow$$

$\det(B) = 0\,?$ $\xrightarrow{\text{yes}}$ no perfect matching

$\xrightarrow{\text{no}}$ $\exists$ perfect matching

To bound $\Pr(\text{false negative})$ use Schwarz-Zippel:

If $P(X_1, \dots, X_m)$ is a polynomial over field $\mathbb{F}$ and max degree of any variable in any monomial of P is d, and $S \subseteq \mathbb{F}$ is a subset of s elements, then when we select random elements of S for $X_1, \dots, X_m$, $\Pr(P=0) \leq \frac{md}{s}$.

In Lovasz Alg, use a field large enough that S can be taken to have $\geq \frac{n^2}{\delta}$ elements. For determinant polynomial $d = 1$ (determinant is "multilinear") and $m = \#$ variables in matrix $= \#$ edges in $G \leq n^2$

$$\frac{md}{s} \leq \frac{n^2 \cdot 1}{n^2/\delta} = \delta.$$

$\Pr(\text{Lovasz yields false negative}) \leq \delta.$

Def. The exponent of matrix multiplication, ω, is the smallest constant c such that $\exists$ algorithms to multiply two $n \times n$ matrices in $O(n^{c+\varepsilon})$ time for all $\varepsilon > 0$.

Facts. The running time for

- matrix inversion
- determinant
- LU factorization

+ deciding membership in a CFG

are all $O(n^{\omega+\varepsilon})$ for all $\varepsilon > 0$.

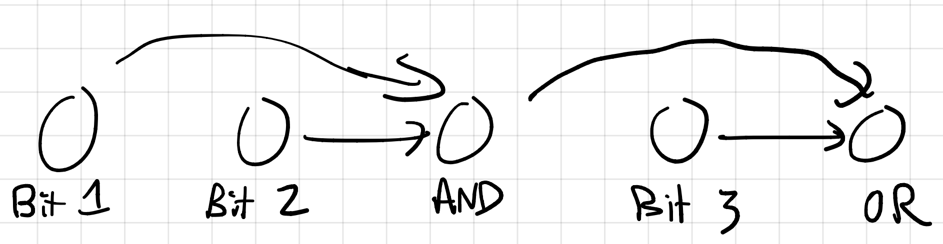
We know $\omega < 2.373$ (Alman & Williams)
and $\omega \geq 2$

Lovasz runs in $O(n^{\omega+\varepsilon}) = O(n^{2.373})$.
Compare with Hopcroft-Karp: $O(m\, n^{0.5}) = O(n^{2.5})$.

Parallel Algorithms

A Boolean circuit is a DAG each of whose nodes is annotated with an operation:

1. Read i^{th} bit of input ($\emptyset$ incoming edges)
2. NOT (must have exactly 1 incoming edge).
3. AND (≥ 1 incoming edge)
4. OR " "



$(x_1 \wedge x_2) \vee x_3$

Two properties that relate to complexity:

1. "Work": total # of nodes in the DAG

2. "depth": (corresponds to parallel running time)

= minimum # of layers $L_0, L_1, \ldots, L_d$ such that nodes can be partitioned into layers with every edge pointing from L_i to L_j with $i<j$.

= # edges in the longest path of the DAG.

Adding 2 binary numbers in parallel.

$$
\begin{array}{lcccc}
\text{"carries"} & c_1 & c_2 & \cdots & c_n = 0 \\
 & a_1 & a_2 & \cdots & a_n \\
 & b_1 & b_2 & \cdots & b_n \\
\hline
s_0 & s_1 & & \cdots & s_n
\end{array}
\qquad a_i, b_i \in \{0,1\}
$$

$$s_i = a_i \oplus b_i \oplus c_i$$

In parallel in $O(1)$, you can conclude:

$$c_1 = c_2 = \underset{\substack{\| \\ 0}}{c_3} \quad c_4 = c_5 = c_6 = \underset{\substack{\| \\ 1}}{c_7}$$

After $O(\log n)$ rounds of "looking ahead" all carry bits are known.

$O(n \log n)$ work and $O(\log n)$ depth.