Cache Oblivious Algorithms

M. Frigo, C. E. Leiserson, H. Prokop, S. Ramachandran

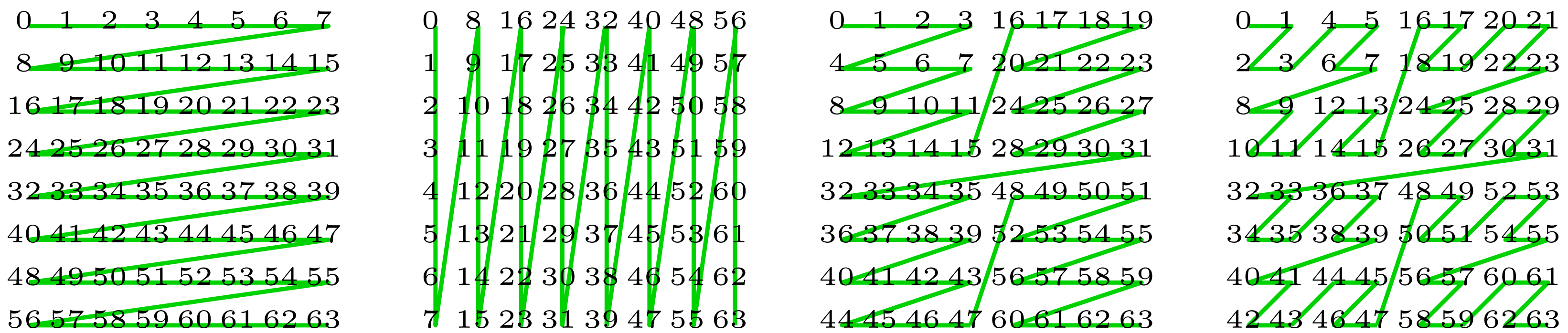
Proc. 40th Annual Symposium on Foundations of Computer Science (FOCS), 1999

Matrix Multiplication

Problem

$$C = A \cdot B \ , \quad c_{ij} = \sum_{k=1..N} a_{ik} \cdot b_{kj}$$

Layout of matrices



Row major Column major 4×4-blocked Bit interleaved

Matrix Multiplication

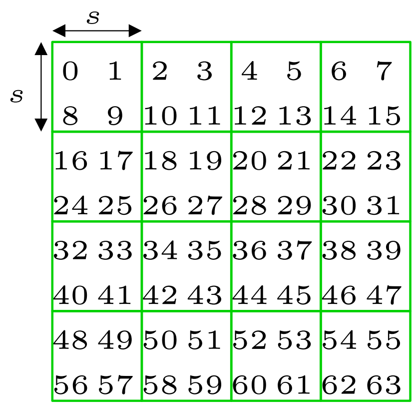
Algorithm 1: Nested loops

- Row major
- Reading a column of B uses N I/Os
- Total $O(N^3)$ I/Os

for $i = 1$ to N
  for $j = 1$ to N
    $c_{ij} = 0$
    for $k = 1$ to N
      $c_{ij} = c_{ij} + a_{ik} \cdot b_{kj}$

Algorithm 2: Blocked algorithm (cache-aware)

- Partition A and B into blocks of size $s \times s$ where $s = \Theta(\sqrt{M})$
- Apply Algorithm 1 to the $\frac{N}{s} \times \frac{N}{s}$ matrices where elements are $s \times s$ matrices
- $s \times s$-blocked or (row major and $M = \Omega(B^2)$)



$$O\left(\left(\frac{N}{s}\right)^3 \cdot \frac{s^2}{B}\right) = O\left(\frac{N^3}{s \cdot B}\right) = O\left(\frac{N^3}{B\sqrt{M}}\right) \text{ I/Os}$$

- Optimal

Hong & Kung, 1981

Matrix Multiplication

Algorithm 3: Recursive algorithm (cache-oblivious)

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix} = \begin{pmatrix} A_{11}B_{11} + A_{12}B_{21} & A_{11}B_{12} + A_{12}B_{22} \\ A_{21}B_{11} + A_{22}B_{21} & A_{21}B_{12} + A_{22}B_{22} \end{pmatrix}$$

- 8 recursive $\frac{N}{2} \times \frac{N}{2}$ matrix multiplications + 4 $\frac{N}{2} \times \frac{N}{2}$ matrix sums
- # I/Os if bit interleaved or (row major and $M = \Omega(B^2)$)

$$T(N) \leq \begin{cases} O(\frac{N^2}{B}) & \text{if } N \leq \varepsilon\sqrt{M} \\ 8 \cdot T\left(\frac{N}{2}\right) + O\left(\frac{N^2}{B}\right) & \text{otherwise} \end{cases}$$

$$T(N) \leq O\left(\frac{N^3}{B\sqrt{M}}\right)$$

- Optimal — Hong & Kung, 1981
- Non-square matrices — Frigo et al., 1999

Matrix Multiplication

Algorithm 4: Strassen's algorithm (cache-oblivious)

– 7 recursive $\frac{N}{2} \times \frac{N}{2}$ matrix multiplications + $O(1)$ matrix sums

$$\begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}$$

$$\begin{array}{lcl}
m_1 & := & (a_{21}+a_{22}-a_{11})(b_{22}-b_{12}+b_{11}) \\
m_2 & := & a_{11}b_{11} \\
m_3 & := & a_{12}b_{21} \\
m_4 & := & (a_{11}-a_{21})(b_{22}-b_{12}) \\
m_5 & := & (a_{21}+a_{22})(b_{12}-b_{11}) \\
m_6 & := & (a_{12}-a_{21}+a_{11}-a_{22})b_{22} \\
m_7 & := & a_{22}(b_{11}+b_{22}-b_{12}-b_{21})
\end{array}
\qquad
\begin{array}{lcl}
c_{11} & := & m_2+m_3 \\
c_{12} & := & m_1+m_2+m_5+m_6 \\
c_{21} & := & m_1+m_2+m_4-m_7 \\
c_{22} & := & m_1+m_2+m_4+m_5
\end{array}$$

– # I/Os if bit interleaved or (row major and $M = \Omega(B^2)$)

$$T(N) \leq \begin{cases} O(\frac{N^2}{B}) & \text{if } N \leq \varepsilon\sqrt{M} \\ 7 \cdot T\left(\frac{N}{2}\right) + O\left(\frac{N^2}{B}\right) & \text{otherwise} \end{cases}$$

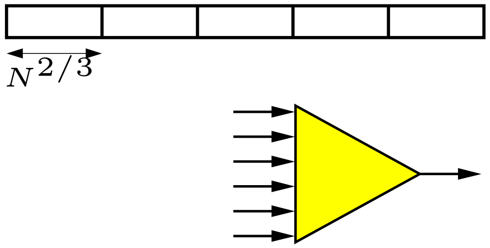
$$T(N) \leq O\left(\frac{N^{\log_2 7}}{B\sqrt{M}}\right) \qquad \log_2 7 \approx 2.81$$

Cache-Oblivious Sorting

Merge-Sorting : Funnel-sort

Funnel-sort

1. Partition input into $N^{1/3}$ pieces of size $N^{2/3}$
2. Sort each piece recursively
3. Merge sorted pieces using an $N^{1/3}$-merger



k-merger

A k-merger takes k sorted input sequences

Each invocation of a k-merger outputs the next k^3 elements

Theorem

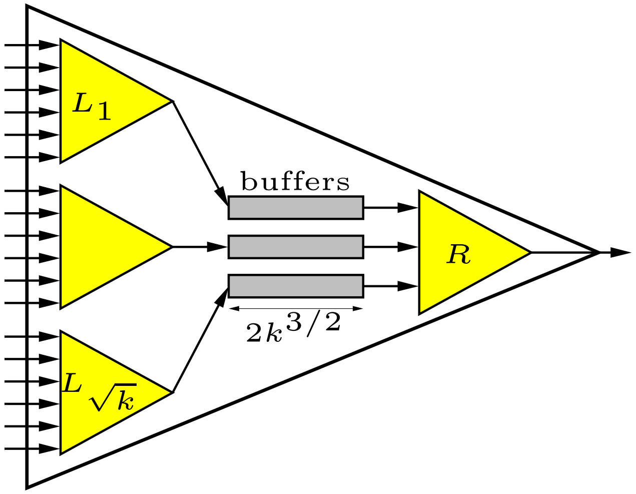
If an invocation of a k-merger uses $O(k + \frac{k^3}{B} \log_M k^3)$ I/Os then funnel-sort uses $O(\frac{N}{B} \log_M N)$ I/Os

Proof

$$T(N) = \begin{cases} \frac{N}{B} & \text{if } N \leq M \\ N^{1/3} \cdot T(N^{2/3}) + \frac{N}{B} \log_M N & \text{otherwise} \end{cases}$$

□

k-merger



- $\sqrt{k}+1$ recursive $\sqrt{k}$-mergers
- $\sqrt{k}$ buffers (FIFO) of size $2k^{3/2}$
  Implemented as cyclic queues in arrays
  (efficient scanning of input streams)
- 2-merger outputs 8 elements

Invocation of a k-merger

- $k^{3/2}$ times invoke the $\sqrt{k}$-merger R
- Before invoking R invoke all L_i where the buffer contains less than $k^{3/2}$ elements (and the input streams are not exhausted)

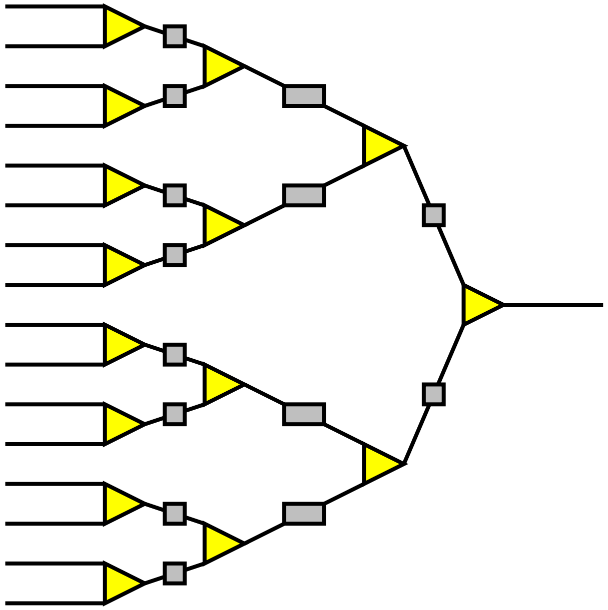
Lemma Can be laid out in $O(k^2)$ contiguous memory locations

Proof

$$S(k) = (\sqrt{k}+1)\cdot S(\sqrt{k}) + O(2k^{3/2}\cdot\sqrt{k}) \qquad \square$$

k-merger

Example : 16-merger



k-merger $\equiv$ a scheduling of 2-mergers

Memory accesses

2-merger : Reads/writes sequentially input and output

k-merger : $k^{3/2} \cdot \sqrt{k} = k^2$ reads to check if input buffers $\geq k^{3/2}$

k-merger

Lemma A k-merger makes $O(k + \frac{k^3}{B} \log_M k^3)$ I/Os per invocation

Case I: $k \leq \varepsilon\sqrt{M}$

Space $\underbrace{k \cdot B}_{\text{input}} + \underbrace{B}_{\text{output}} + \underbrace{O(k^2)}_{\text{buffers}} = O(M)$ assuming $M = \Omega(B^2)$

i.e. computation can be done in internal memory

I/Os $O(\underbrace{k + k^3/B}_{\text{input/output}} + \underbrace{k^2/B}_{\text{touching buffers}}) = O(k + k^3/B)$

Case II: $k > \varepsilon\sqrt{M}$

At some level in the recursion $\sqrt{\varepsilon\sqrt{M}} < \bar{k} \leq \varepsilon\sqrt{M}$

$$\bar{k}^2 \geq \varepsilon\sqrt{M} = \Omega(B) \quad \Rightarrow \quad \bar{k}^3/B = \Omega(\bar{k})$$

By Case I a $\bar{k}$-merger uses $O(\bar{k}^3/B) = O(\frac{\text{output}}{B})$ I/Os

k-merger

Case II: $k > \varepsilon\sqrt{M}$ (continued)
I/Os used for merging k^3 elements in a k-merger

– In $\bar{k}$-mergers : $O(\frac{k^3}{B} \cdot \log_{\bar{k}} k)$ I/Os

– Checking buffer sizes between two $\bar{k}$-mergers

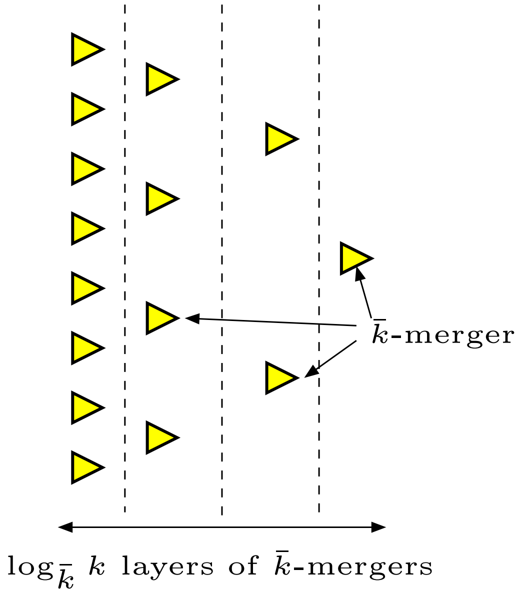
Invoking a $\hat{k}$-merger, $\hat{k} > \bar{k}$

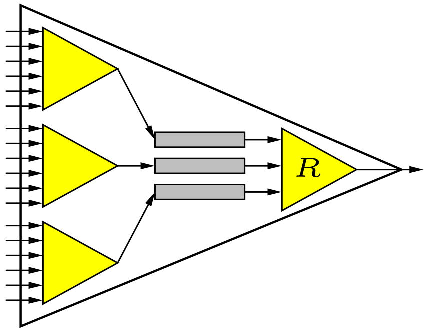
- $\hat{k}^2$ reads to check buffer sizes
- $\hat{k}^3$ elements propagated from the buffers
- I/Os per element propagated

$$\frac{\hat{k}^2}{\hat{k}^3} = \frac{1}{\hat{k}} = O(\frac{1}{\sqrt{M}}) = O(\frac{1}{B})$$

Total #I/Os for buffer checks is

$$O(\frac{1}{B} \cdot k^3 \log_{\bar{k}} k) = O(\frac{k^3}{B} \log_M k)$$





□

Sorting

Theorem

Funnel-sort requires $O(\frac{N}{B}\log_M N) = O(\frac{N}{B}\log_{M/B}\frac{N}{B})$ I/Os for $M = \Omega(B^2)$