The Input/Output Complexity of Sparse Matrix Multiplication

Rasmus Pagh[1], Morten Stöckel[2]

[1]IT University of Copenhagen, [2] University of Copenhagen

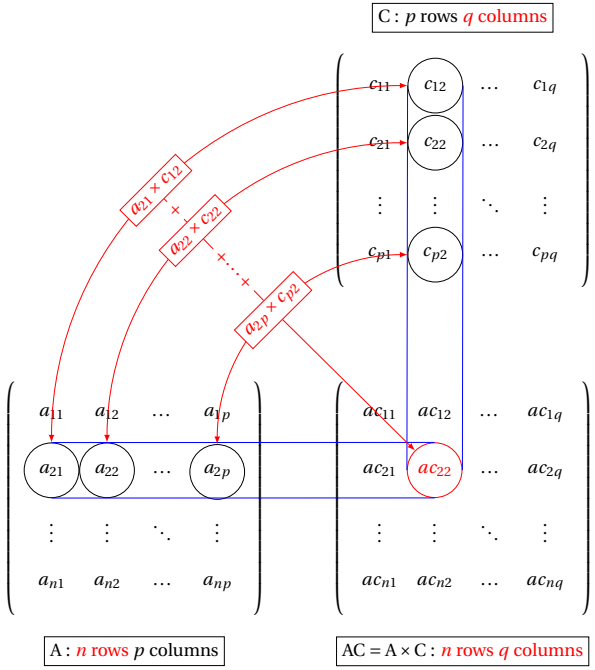
SIAM LA, October 26 2015

Overview

- Let A and C be matrices over a semiring $\mathbb{R}$ with N nonzero entries in total.
- The problem: Compute matrix product $[AC]_{i,j} = \sum_k A_{i,k}C_{k,j}$ with Z nonzero entries.
- Central result: Can be done in (for most of parameter space) optimal $\tilde{O}\left(\frac{N\sqrt{Z}}{B\sqrt{M}}\right)$ I/Os.

Cancellation of elementary products

We say that we have *cancellation* when two or more summands of $[AC]_{i,j} = \sum_k A_{i,k}C_{k,j}$ are nonzero but the sum is zero. Our algorithm handles such cases.



Motivation

Lots of applications. Some of them:

- Computing determinants and inverses of matrices.
- Bioinformatics.
- Graphs: counting cycles, computing matchings.

The semiring I/O model, 1

- A word is big enough to hold a matrix element plus its coordinates.
- Internal memory that holds M words and disk of infinite size.
- One I/O: Transfer B words from disk to internal memory.
- Cost of an algorithm: Number of I/Os used.
- Operations allowed: Semiring operations, copy and equality check.

The semiring I/O model, 2

- We make no assumptions about cancellation.
- To produce output: must invoke `emit`$(.)$ on every nonzero output entry once.
- Matrices are of size $U \times U$.
- $\tilde{O}$ suppresses $\mathrm{polylog}$ factors in U and N.

Our results, 1

- Let A and C be $U \times U$ matrices over semiring $\mathbb{R}$ with N nonzero input and Z nonzero output entries. There exist algorithms 1 and 2 such that:
  1. emits the set of nonzero entries of AC with probability at least $1 - 1/U$, using $\tilde{O}\left(N\sqrt{Z}/(B\sqrt{M})\right)$ I/Os.
  2. emits the set of nonzero entries of AC, and uses $O\left(N^2/(MB)\right)$ I/Os.
- Previous best [Amossen-Pagh, '09]: $\tilde{O}\left(N\sqrt{Z}/(BM^{1/8})\right)$ I/Os (boolean matrices $\implies$ no cancellation).

Our results, 2

- Let A and C be $U \times U$ matrices over semiring $\mathbb{R}$ with N nonzero input and Z nonzero output entries. There exist algorithms 1 and 2 such that:
  1. emits the set of nonzero entries of AC with probability at least $1 - 1/U$, using $\tilde{O}\left(N\sqrt{Z}/(B\sqrt{M})\right)$ I/Os.
  2. emits the set of nonzero entries of AC, and uses $O\left(N^2/(MB)\right)$ I/Os.
- There exist matrices that require $\Omega\left(\min\left(\frac{N^2}{MB}, \frac{N\sqrt{Z}}{B\sqrt{M}}\right)\right)$ I/Os to compute all nonzero entries of AC.

Output size estimation

Size estimation tool: Given matrices A and C with N nonzero entries, compute ε-estimate of number of nonzeroes of each column of AC using $\tilde{O}(\varepsilon^{-3}N/B)$ I/Os.

Fact (Bender et al, '07)

For dense $1 \times U$ vector y and sparse $U \times U$ matrix S we can compute yS in $\tilde{O}((\mathtt{nnz}(S)/B)$ I/Os.

Distinct elements and matrix size

- Distinct elements: Given frequency vector x of size n where x_i denotes the number of times element i occurs, then $F_0 = \sum_i |x_i|^0$.
- Fundamental problem in streaming: Estimate F_0 without materializing x.
- Observation: The distinct elements of AC is $\texttt{nnz}(AC)$.
- Good news: use existing machinery. Size $O(\varepsilon^{-3} \log n \log \delta^{-1}) \times n$ matrix F exists s.t Fx gives F_0 whp [Flajolet-Martin, '85].

Output estimation

F is $\varepsilon^{-3} \log \delta^{-1} \log U \times U$.
A and C are $U \times U$.
To get size estimate we must compute:

$$F \times A \times C$$

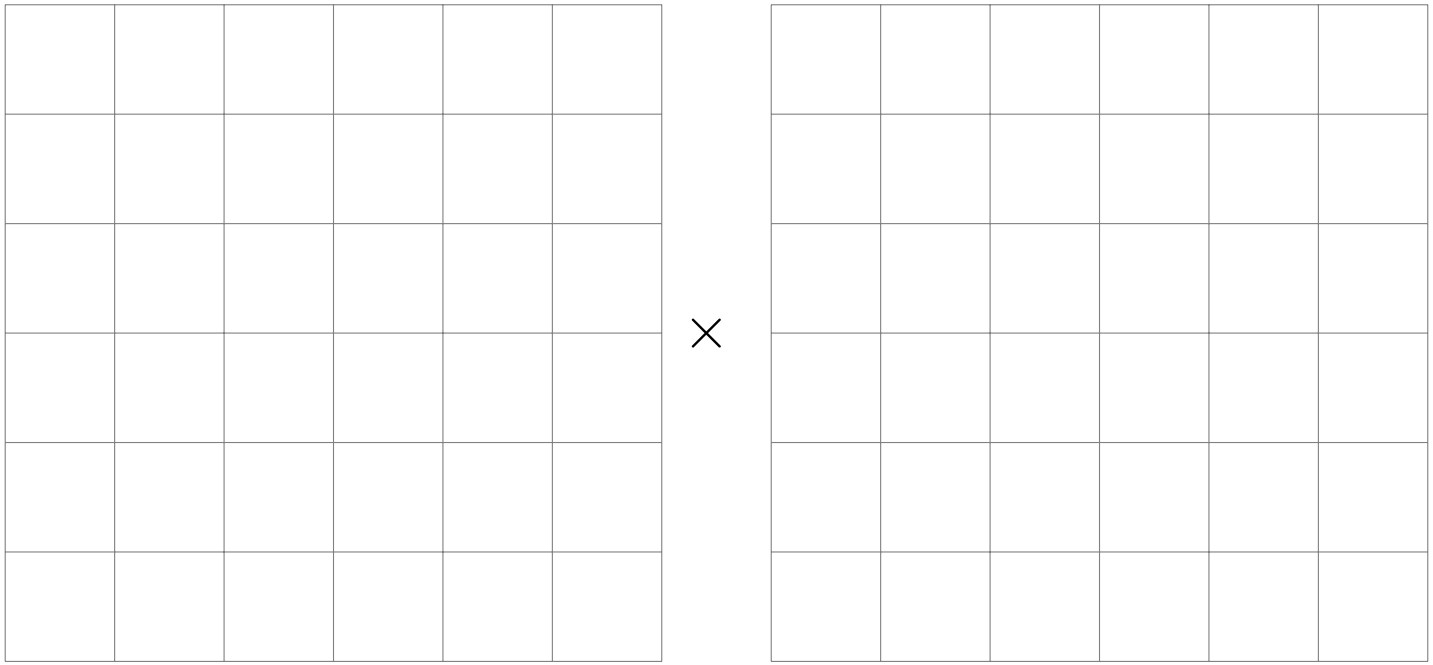
Output estimation

F is $\varepsilon^{-3} \log \delta^{-1} \log U \times U$.
A and C are $U \times U$.
To get size estimate we must compute:

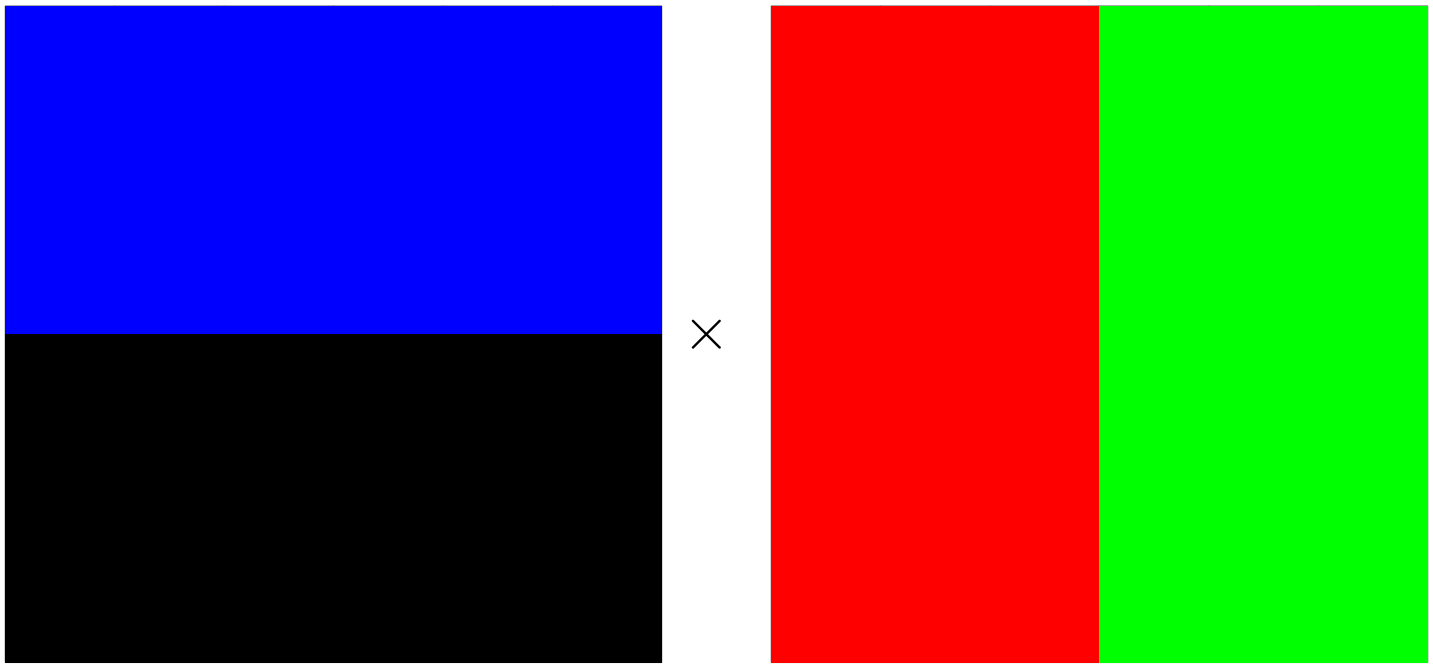
$$(F \times A) \times C$$

Due to associativity: Pick cheap order.
Analysis: $\varepsilon^{-3} \log \delta^{-1} \log U$ invocations of dense vector sparse matrix black box: $\tilde{O}(\varepsilon^{-3} N/B)$ I/Os.
Note: Works with cancellation, contrary to previous size estimation.

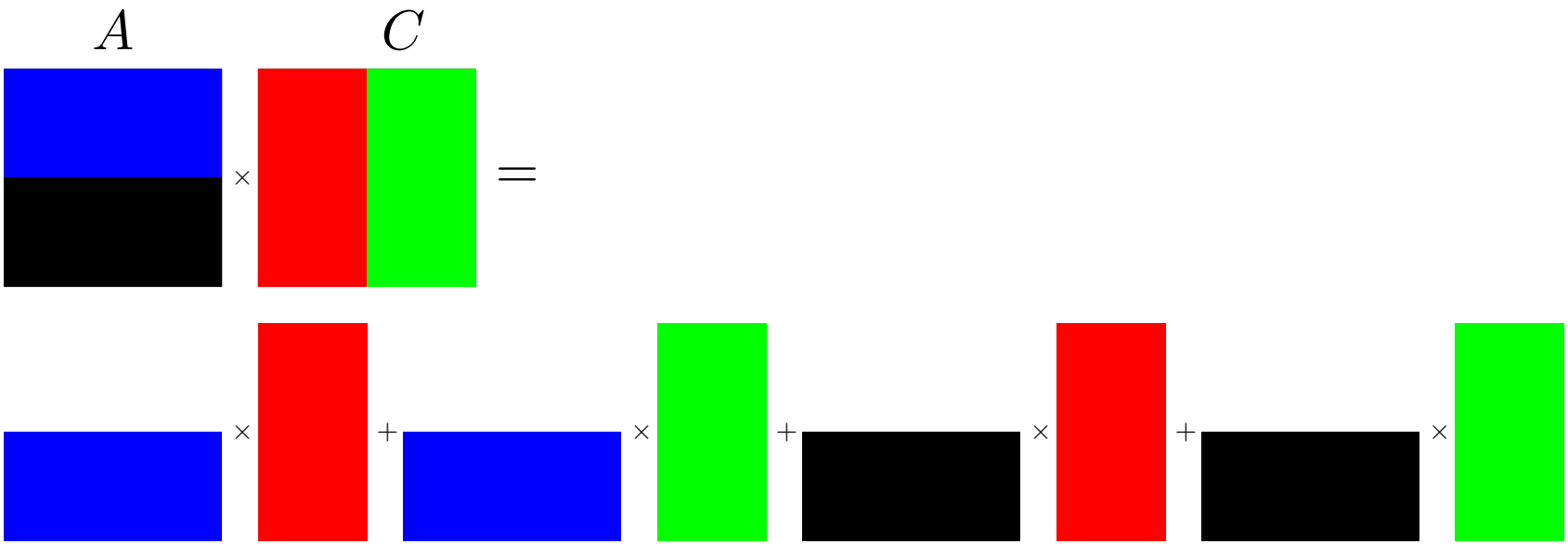
Matrix mult partitioning, 1



Matrix mult partitioning, 1

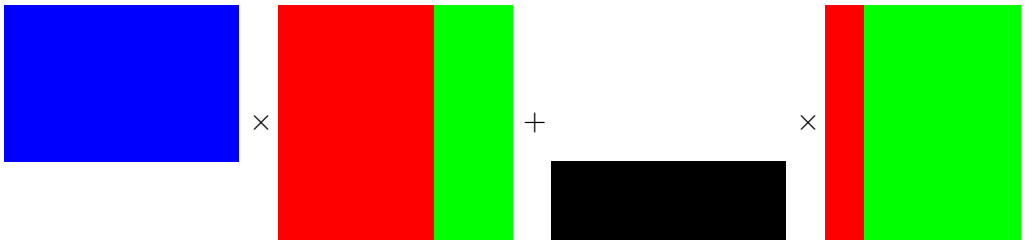


Matrix mult partitioning, 2



Partitioning the matrices

- What we want: Split matrices into disjoint colored groups s.t. every color combination has at most M nonzero output entries.
- Problem: Can't be done.
- Instead: Color rows of A using c colors. For each c groups of rows, do an independent coloring with c colors of columns of C.



Partitioning the matrices, 2

Overview of how to partition matrices A and C:

1. Pick number of colors $c = \sqrt{\frac{\mathtt{nnz}(AC)\log U}{M}} + O(1)$
2. Recurse: Split A into A_1 and A_2 where it holds: $\mathtt{nnz}(A_1C) \approx \mathtt{nnz}(AC)/2$ and $\mathtt{nnz}(A_2C) \approx \mathtt{nnz}(AC)$.
3. After $\log c + O(1)$ recursive levels we have $O(c)$ disjoint colored groups of rows of A.
4. For each of those groups: Repeat procedure for columns of C.
5. The key point: $O(c^2)$ problems of size $\mathtt{nnz}(AC)/c^2 = O(M/\log U)$.

Getting the correct subproblem size

Say we can do splits of A into A_1, A_2 s.t.

1. $\mathtt{nnz}(A_1 C) \in \left[(1 - \log^{-1} U)\,\mathtt{nnz}(AC)/2; (1 + \log^{-1} U)\,\mathtt{nnz}(AC)/2\right]$.
2. $\mathtt{nnz}(A_2 C) \in \left[(1 - \log^{-1} U)\,\mathtt{nnz}(AC)/2; (1 + \log^{-1} U)\,\mathtt{nnz}(AC)/2\right]$.

Assume biggest possible positive error: after q recursions have problem output size $\mathtt{nnz}(AC)(1/2 + 1/(2\log U))^q$. Then after $\log c^2 + O(1)$ recursions:

$$\begin{aligned}\mathtt{nnz}(AC)\left(\frac{1}{2} + \frac{1}{2\log U}\right)^{\log c^2} &\leq \mathtt{nnz}(AC) 2^{-\log c^2} e^{\frac{\log c^2}{\log U}} \\ &\leq \mathtt{nnz}(AC) O(1)/c^2 = O(M/\log U)\end{aligned}$$

How to compute the split

How to do relative error $1/\log U$ splits: Use size estimation tool:
For any set r of rows we have access to $\hat{z}_i$'s s.t.

$$(1-\log^{-1} U)\,\mathtt{nnz}\left(\sum_{i\in r}[AC]_{i*}\right) \leq \sum_{i\in r}\hat{z}_i \leq (1+\log^{-1} U)\,\mathtt{nnz}\left(\sum_{i\in r}[AC]_{i*}\right).$$

Splitting A into A_1 and A_2:

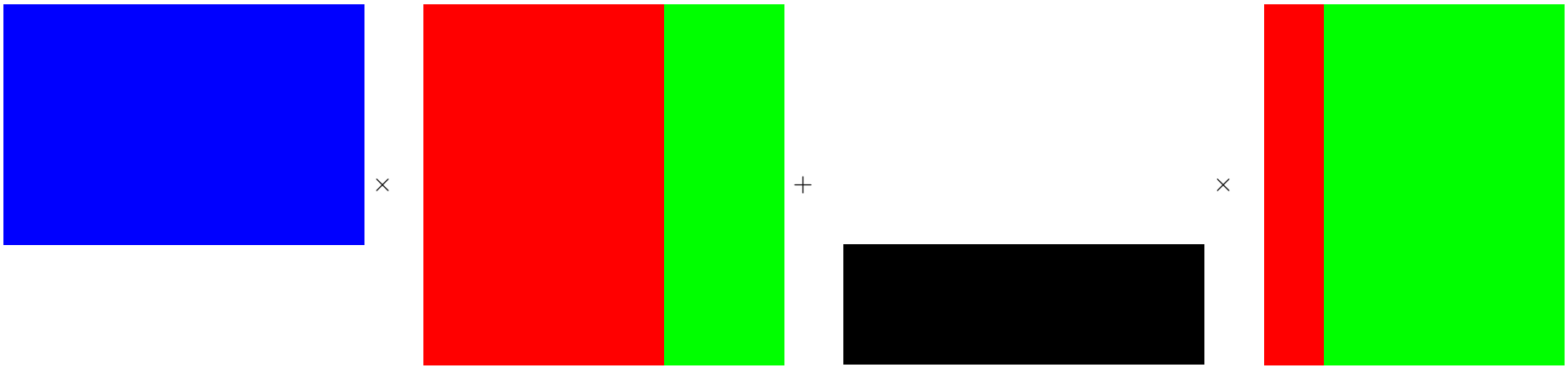
1. Let $\hat{Z} = \sum_i \hat{z}_i$.
2. Add rows from A to A_1 until $\sum_{i\in A_1}\hat{z}_i \geq \hat{Z}/2$.
3. The row that y overflows A_1: Compute $y \times C$ directly.
4. Add remaining rows to A_2

I/O cost of splitting

I/O cost:

- Initial size est: $\tilde{O}(N/B)$.
- Partition A: c dense-vector-sparse-matrix: $\tilde{O}(cN/B)$.
- For the c A-partitions: one size est of total $\tilde{O}(N/B)$ and c DVSM of total $\tilde{O}(cN/B)$.
- Total: $\tilde{O}(cN/B) = \tilde{O}\left(\frac{N\sqrt{\mathtt{nnz}(AC)}}{B\sqrt{M}}\right)$ since $c = \sqrt{\frac{\mathtt{nnz}(AC)\log U}{M}}$.

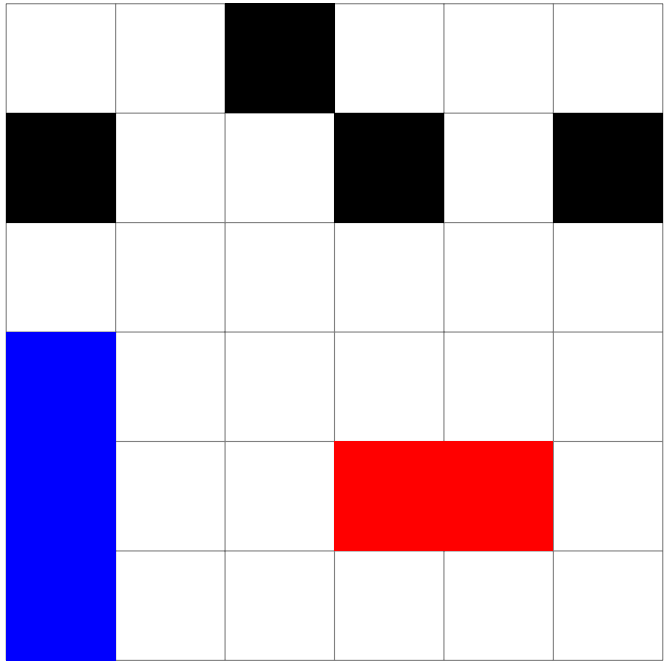
Are we done?



Status

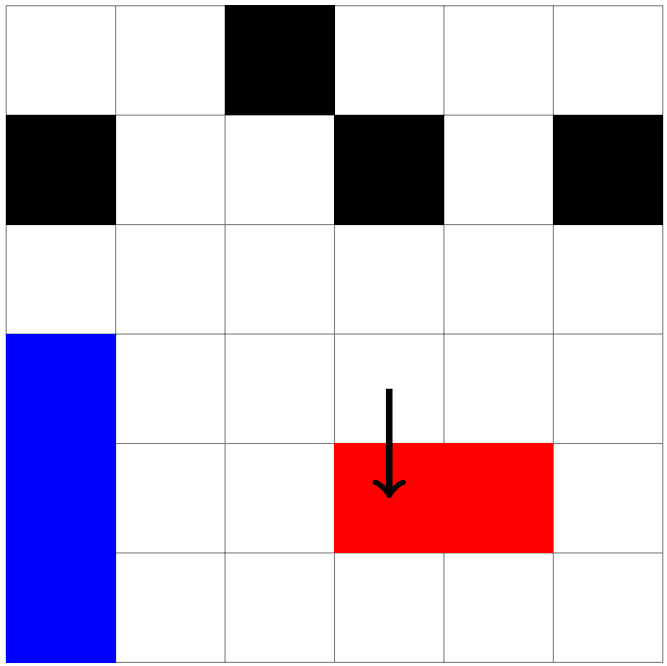
- Where we are: have $c^2 = \frac{\mathtt{nnz}(AC)\log U}{M}$ subproblems with output $\leq M/\log U$.
- Central cancellation difficulty: Intermediate results can be much larger than M.
- Our I/O aim: $\tilde{O}(cN/B)$, hence we can't pay for those cancelling inner products.
- Solution: Compute a particular polynomial and allow polynomially small error probability.

Compressed matrix mult intuition



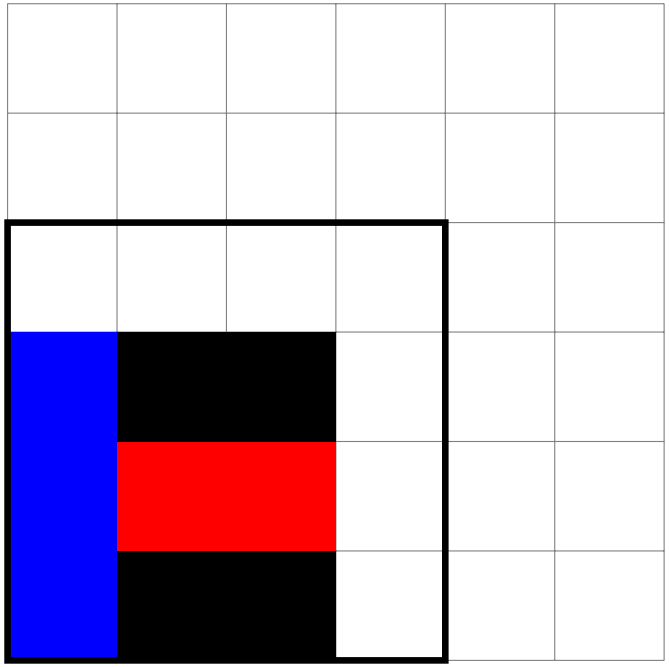
$A_i C_j$

Compressed matrix mult intuition



$A_i C_j$

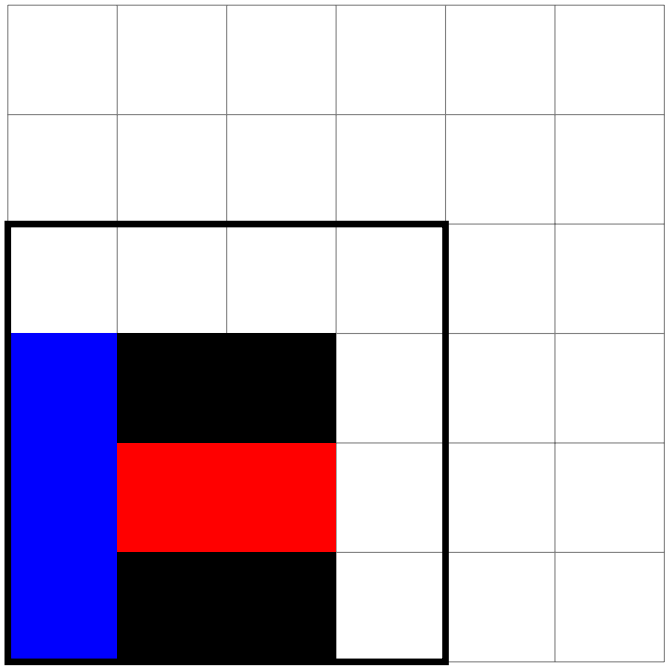
Compressed matrix mult intution



$A_i C_j$

Compressed matrix mult

- Let $r = M/\log U$ be the number of output entries in a subproblem.
- We can perform compressed matrix mult in $4r$ space by computing a $O(r)$-degree polynomial [Pagh, '12].
- Need $O(\log U)$ repetitions to get high probability.



Algorithm summary

I/O cost of steps taken:

- Initial size est: $\tilde{O}(N/B)$.
- Partition into c^2 problems with output $M/\log U$: $\tilde{O}(cN/B)$.
- Compute and `emit`(.) all subproblems: $\tilde{O}(cN/B)$.
- Total: $\tilde{O}(cN/B) = \tilde{O}\left(\frac{N\sqrt{\mathtt{nnz}(AC)}}{B\sqrt{M}}\right)$ since $c = \sqrt{\frac{\mathtt{nnz}(AC)\log U}{M}}$.

Lower bound, technique used

- We will show: $\Omega\left(\frac{N}{B}\min\left(\sqrt{\frac{Z}{M}}, \frac{N}{M}\right)\right)$ I/Os needed.
- Argument type follows "phase argument" [Hong and Kung, '81] – divide execution in phases of M/B I/Os.
- Double memory to be $2M$: There now exists equivalent execution where reads and writes are ordered.
- This allows us to argue: For a specific computation, how good is the best possible execution.

Technique, continued

- Our hard instance: Dense matrices A is $\sqrt{Z} \times \frac{N}{\sqrt{Z}}$ and C is $\frac{N}{\sqrt{Z}} \times \sqrt{Z}$.
- Notice: $\texttt{nnz}(A) + \texttt{nnz}(C) = \Theta(N)$ and $\texttt{nnz}(AC) = \Theta(Z)$.
- Crucial due to semiring operations: Every stored element is always either:
  1. an input entry
  2. entry from a partial sum
- We are now ready to argue about number of phases needed to create two types of output.

Bounding direct outputs

- **Direct outputs**: All needed entries are stored – requires two $\frac{N}{\sqrt{Z}}$-size vectors to be stored.
- At most $\frac{2M\sqrt{Z}}{N}$ vectors fit in memory, thus at most $\frac{M^2Z}{N^2}$ direct outputs possible.
- To output $Z/2$ of this type: $(Z/2)/\frac{M^2Z}{N^2} = (N/M)^2$ phases needed, hence $\Omega\left(\frac{N^2}{BM}\right)$ I/Os.

Bounding indirect outputs

- **Indirect outputs**: Output entries for which an elementary product is written in some phase.
- In space $2M$, the number of elementary products stored and computed is at most $(2M)^{3/2}$ [Irony et al, '04].
- To output $Z/2$ of this type: $Z/2 \cdot N/\sqrt{Z} = N\sqrt{Z}/2$ elementary products to be computed.
- Number of phases needed: $\frac{N\sqrt{Z}/2}{(2M)^{3/2}}$, thus $\Omega\left(\frac{N\sqrt{Z}}{B\sqrt{M}}\right)$ I/Os.

Lower bound summary

- To do $Z/2$ direct: $\Omega\left(\frac{N^2}{BM}\right)$ I/Os.
- To do $Z/2$ indirect: $\Omega\left(\frac{N\sqrt{Z}}{B\sqrt{M}}\right)$ I/Os.
- Since at least $Z/2$ of either is needed, lower bound becomes minimum of the two.

Concluding remarks

- Size estimation: Supports cancellation and uses $\tilde{O}(\varepsilon^{-3}N/B)$ I/Os.
- Algorithm 1: $\tilde{O}\left(N\sqrt{Z}/(B\sqrt{M})\right)$ I/Os.
- Algorithm 2: $O\left(N^2/(MB)\right)$ I/Os.
- Lower bound: $\Omega\left(\min\left(\frac{N^2}{MB}, \frac{N\sqrt{Z}}{\sqrt{MB}}\right)\right)$ I/Os.

Open: Remove monte carlo (and log factors).

The Input/Output Complexity of Sparse Matrix Multiplication

Rasmus Pagh[1], Morten Stöckel[2]

[1]IT University of Copenhagen, [2] University of Copenhagen

SIAM LA, October 26 2015