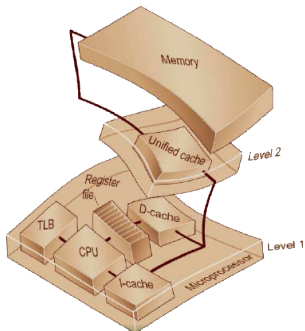


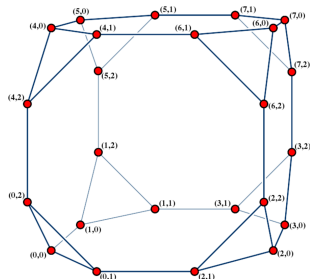
On The Limits Of Cache And Network-Oblivious Matrix Transposition



Francesco Silvestri

silvest1@dei.unipd.it

DEPARTMENT OF
INFORMATION
ENGINEERING
UNIVERSITY OF PADOVA



UTCS - November 16th, 2007

Results presented in this talk were published in
[Bilardi, Pietracaprina, Pucci, Silvestri, 2007] and *[Silvestri, 2006]*

Outline of the Talk

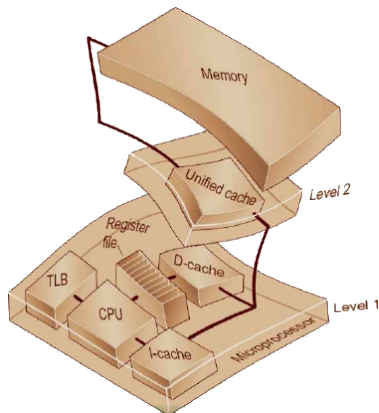
- 1 Overview of cache-oblivious algorithms
- 2 Limits of cache-oblivious matrix transposition
- 3 Overview of network-oblivious algorithms
- 4 Limits of network-oblivious matrix transposition

PART I

LIMITS OF CACHE-OBLIVIOUSNESS

The Memory Hierarchy

- Modern platforms feature a hierarchical cascade of memories
- The farther the memory from the CPU, the bigger the **capacity** and **access time**



Locality of Reference

- In good algorithms:
 - The same data are frequently reused within a short time interval

Temporal Locality of Reference

- Data stored at consecutive addresses are involved in consecutive operations

Spatial Locality of Reference

- Many computational models account for memory hierarchy:
 - External Memory (EM)
 - Ideal Cache (IC)

External Memory Model

External Memory, $EM(M, B)$ [Aggarwal, Vitter, 1988]:

- Represents DISK-RAM hierarchy
- Arbitrarily large disk, RAM of M words
- Data are transferred in blocks of B consecutive words
- Block transfers are **explicitly controlled** by the program
- **I/O Complexity**: number of accesses to disk

Ideal Cache Model

Ideal Cache, $IC(M, B)$ [Frigo et al., 1999]:

- Represents RAM-CACHE hierarchy
- Arbitrarily large RAM, cache of M words
- Cache:
 - Organized into M/B blocks of B words
 - Fully associative
- Data are transferred in blocks of B consecutive words
- Block transfers are **automatically controlled** by hardware:
 - Optimal offline strategy for block replacement
- **Cache Complexity**: number of misses

Cache-Oblivious Algorithms

Definitions

- An IC-algorithm is **Cache-Oblivious** (CO) if its specification is independent of M and B
- An algorithm is **Cache-Aware** (CA) otherwise
- **Optimal CO algorithm**: asymptotically optimal cache complexity on each $IC(M, B)$ model

Remark

CO algorithms adapt **automatically** to the platform in which they run

Framework for Cache-Oblivious Algorithms

Specification model: Random Access Machine (RAM)



Evaluation model: the IC model, described by M and B



Execution model: an arbitrary memory hierarchy

Remark

For a wide class of cache-oblivious algorithms, optimality in the evaluation model implies optimality in the execution model

Cache-Obliviousness vs Awareness

- Frigo *et. al* raised the question of the theoretical relationship between CO and CA algorithms
- For certain problems there exist CO algorithms which run asymptotically as fast as their CA optimal ones
- For other problems, CO algorithms achieve optimality only under the **Tall Cache Assumption**

Definition

An $IC(M, B)$ satisfies the **Tall Cache Assumption** (TCA) if:

$$B \leq \sqrt{M}$$

Previous Results

- *[Bilardi, Peserico, 2001]:*
 - $f(x)$ -HMM: access to the i -th memory location takes time $f(i)$
 - HMM does not feature block transfer
 - \exists a CDAG and two HMMs for which any fixed scheduling is sub-optimal on at least one of the models
- *[Brodal, Fagerberg, 2003]:*
 - There is no optimal CO comparison based algorithm for the sorting problem without the TCA
 - There is no optimal CO algorithm for general permutations, not even in the presence of TCA

The Problem: Matrix Transposition (MT)

Problem Specification

Input: an $\sqrt{n} \times \sqrt{n}$ matrix A in row-major

Output: A^T in row major

- Cache or I/O complexity (if $n > M^2$) [Aggarwal, Vitter, 1988]:

$$\Omega \left(\frac{n}{B} \left(\frac{\log B}{\log \left(\frac{M}{B} + 1 \right)} + 1 \right) \right)$$

- There is an **optimal** CA algorithm for each $IC(M, B)$, $M \geq 1, 1 \leq B \leq M$ [Aggarwal, Vitter, 1988]
- There is an **optimal** CO algorithm for every $IC(M, B)$ that satisfies the TCA [Frigo et al., 1999]

Lower Bound on I/O Complexity

- Consider the disk subdivided into blocks of B words (B -blocks)
- i -th target group: A 's entries that will ultimately be in the i -th B -block of A^T
- n/B different target groups, B entries per group
- **Begin**: each B -block of A holds B entries of B target groups
- **End**: each B -block of A^T holds B entries of one target group

Lower Bound on I/O Complexity (2)

- **Potential function** after x I/Os ($POT(x)$):
 - Measures the progress of a MT EM-algorithm
 - The bigger the potential, the bigger the number of entries in a B -block belonging to the same target group
 - Let Q be the algorithm's I/O complexity:

$$POT(0) = 0, POT(Q) = n \log B$$

- Each input increases the potential of at most:

$$O\left(B \log \frac{M}{B}\right)$$

- Each output does not decrease the potential
- Lower bound follows □

Cache-Aware Algorithm for MT

Cache-aware algorithm:

- Based on a merging procedure
- In each pass, the number of entries in a B -block belonging to the same target group increases of a multiplicative factor M/B
- Each pass requires $\Theta(n/B)$ misses
- Number of passes: $\log B / \log(M/B)$
- Optimal cache complexity:

$$\Theta \left(\frac{n}{B} \left(\frac{\log B}{\log \left(\frac{M}{B} + 1 \right)} + 1 \right) \right)$$

Cache-Oblivious Algorithm for MT

Cache-oblivious algorithm:

- Divide the input matrix into four submatrices of size $\sqrt{n}/2 \times \sqrt{n}/2$
- Transpose each submatrix recursively
- Under the TCA, optimal cache complexity:

$$\Theta\left(\frac{n}{B}\right)$$

What Are We Going to Prove?

Theorem

*There cannot exist an **optimal** CO algorithm for matrix transposition for every value of $M \geq 1$ and $1 \leq B \leq M$*

How?

Exploiting EM lower bound arguments through a simulation technique

The Simulation Technique

- \mathcal{A} : optimal CO algorithm for MT **without TCA**
- \mathcal{C}_1 : $IC(M, B_1)$; \mathcal{C}_2 : $IC(M, B_2)$, $B_1 < B_2$
- Q_1, Q_2 : (optimal) cache complexities of \mathcal{A} on \mathcal{C}_1 and \mathcal{C}_2
- Sequence of operations does **not change**
- Sequence of misses could **be different**

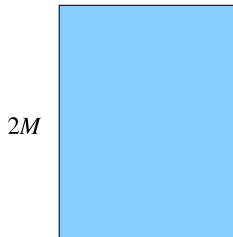
Remark

When an operation requires the word x , the B_1 -block and B_2 -block containing x must be in \mathcal{C}_1 and \mathcal{C}_2

The Simulation Technique (2)

- \mathcal{A}' : new MT algorithm for $EM(2M, B_2)$
- \mathcal{A}' simulates the executions of \mathcal{A} on both \mathcal{C}_1 and \mathcal{C}_2 at the same time
 - sequence of operations
 - both sequences of misses

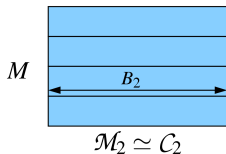
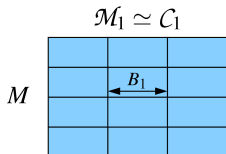
The Simulation Technique (3)



Simulation

Divide the memory into two segments \mathcal{M}_1 and \mathcal{M}_2

The Simulation Technique (3)

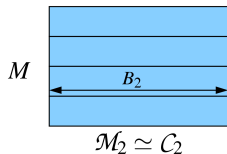
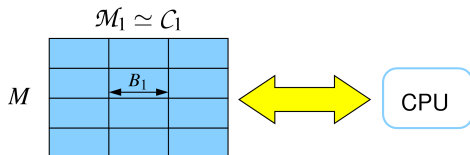


Simulation

\mathcal{M}_1 represents \mathcal{C}_1

\mathcal{M}_2 represents \mathcal{C}_2

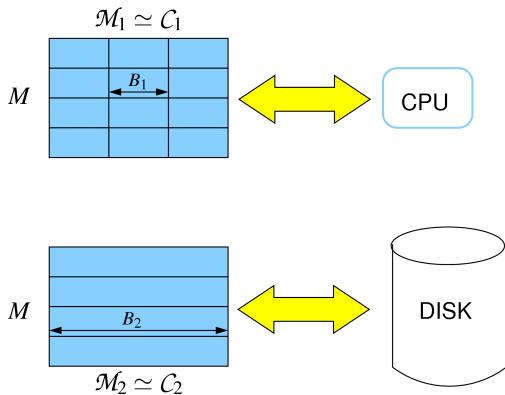
The Simulation Technique (3)



Simulation

Operations only on \mathcal{M}_1

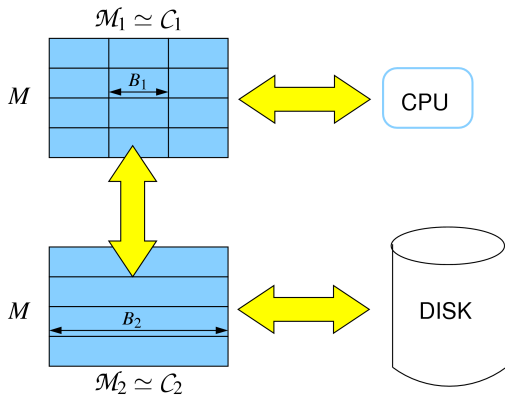
The Simulation Technique (3)



Simulation

I/Os only on \mathcal{M}_2

The Simulation Technique (3)



Simulation

- Miss in $\mathcal{C}_2 \Rightarrow$ I/Os between \mathcal{M}_2 and the disk
- Miss in $\mathcal{C}_1 \Rightarrow$ words exchanged between \mathcal{M}_2 and \mathcal{M}_1

The Simulation Technique (4)

- The I/O complexity of \mathcal{A}' : $Q \in \Theta(Q_2)$
- Let

$$K = \frac{Q_1 B_1}{B_2}$$

Crux

Change \mathcal{A}' in such a way that $O(K)$ words are exchanged between \mathcal{M}_1 and a B_2 -block in \mathcal{M}_2 , before this block is removed from \mathcal{M}_2

Potential Function

- Adaptation of the argument used to lower bound the I/O complexity
- Potential function after x I/Os ($POT(x)$)
 - Measures the progress of a MT algorithm for $EM(2M, B_2)$
 - $POT(0) = 0$
 - $POT(Q) = n \log B_2$
- The rate of $POT(x)$ is limited by the amount of words exchanged between \mathcal{M}_1 and \mathcal{M}_2 :

$$\Delta POT \in O\left(K \log \frac{M}{K}\right)$$

Limits of CO Matrix Transposition

- Suppose \mathcal{C}_1 satisfies the TCA, but \mathcal{C}_2 does not
- \exists a constant $0 < \gamma < 1$ such that:

$$Q \cdot \Delta POT \geq POT(Q) - POT(0) \Rightarrow Q_2 \in \Omega \left(n \frac{B_2^\gamma}{M} \right)$$

- If $B_2 = \Theta(M)$

$$Q_2 \in \Omega \left(\frac{n}{M^{1-\gamma}} \right) \in \omega \left(n \frac{\log M}{M} \right) \Leftarrow \text{Lower Bound}$$

Contradiction!!!

There cannot exist an **optimal** CO algorithm for MT for every value of M and B

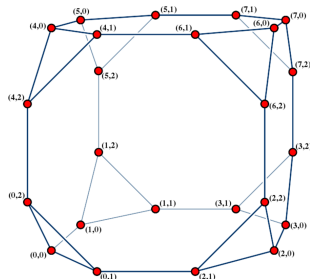


PART II

LIMITS OF NETWORK-OBLIVIOUSNESS

Communication cost

- Communication **heavily** affects the efficiency of parallel algorithms
- Communication costs depend on interconnection topology and other machine-specific characteristics
- Some regions of the network have high bandwidths and low latencies
- **Broad consensus on bandwidth-latency models:**
 - Parameters capture relevant machine characteristics
 - BSP, Decomposable-BSP, QSM, LogP, ...



Question

Can we design efficient parallel algorithms oblivious to any machine/model parameters?

Network-Oblivious Algorithms

- Independent of any machine parameter
- Network-oblivious algorithms adapt **automatically** to the parallel platform in which they run

Framework for Network-Oblivious Algorithms

Specification model: parallelism function of input size,
no machine parameters



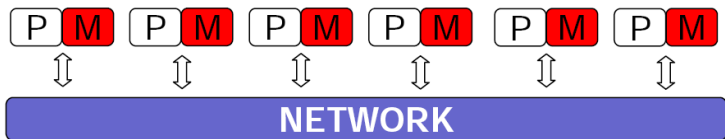
Evaluation model: introduces number of processors p
and communication block size B



Execution model: introduces hierarchical network
structure

Specification Model

Specification model $M(n)$



- n Processing Elements (PEs)
- An algorithm \mathcal{A} is a sequence of supersteps
- In a superstep, each PE can:
 - Perform operations on local data
 - Send/receive messages to/from PEs
- $M(n)$ is a BSP [Valiant, 1990] with no bandwidth and latency parameters

Network-Oblivious Algorithms

Definition

- A **network-oblivious** (NO) algorithm is an $M(n)$ -algorithm, where n is a function of input size
- An algorithm is **network-aware** (NA) otherwise

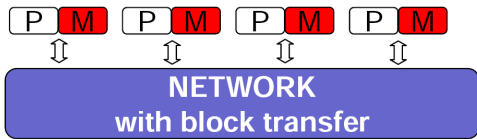
Remarks

NO algorithm specification is

- **independent** of network topology
- **independent** of the actual number of processors

Evaluation Model

Evaluation model $M(p, B)$



- $M(p, B)$ is an $M(p)$ where:
 - Data exchanged between two PEs travel within blocks of B words
- **Block-degree** $h_s(p, B)$: maximum number of blocks sent/received by a PE in a superstep s
- **Communication complexity** of \mathcal{A} :

$$\sum_{\forall s \in \mathcal{A}} h_s(p, B)$$

Evaluation Model (2)

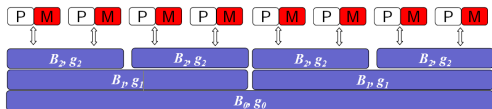
- Execution of an $M(n)$ -algorithm on an $M(p, B)$:
 - Every PE of $M(p, B)$ simulates a segment of n/p consecutive PEs of $M(n)$
 - Communications between PEs of $M(n)$ in the same segment \Rightarrow local computations in $M(p, B)$

Definition

A NO algorithm is **optimal** if, $\forall n, 1 \leq p \leq n$, and $B \geq 1$, its execution on an $M(p, B)$ yields asymptotically optimal communication complexity

Execution Model

Execution Model $D\text{-BSP}(p, \mathbf{g}, \mathbf{B})$ [De la Torre et al., 1996]



- Hierarchical structure of the network
- Recursive decomposition into i -clusters of $p/2^i$ PEs, $0 \leq i < \log p$
- Communications within small clusters are faster than within big clusters

Remark

For a wide class of network-oblivious algorithms, optimality in the evaluation model implies optimality in the execution model

Network-Obliviousness vs Awareness

- For certain problems there exist NO algorithms which run asymptotically as fast as their NA optimal ones
- For other problems, NO algorithms achieve optimality only under the *n-Tall Block Assumption*

Definition

The $M(p, B)$ satisfies the *n-Tall Block Assumption* (*n-TBA*) if:

$$B \leq \sqrt{\frac{n}{p}}$$

Parallel Matrix Transposition

Problem Specification

Input: $\sqrt{n} \times \sqrt{n}$ matrix A in row-major among the processors

Output: A^T in row-major among the processors

- Communication complexity (adaptation of [Aggarwal, Vitter, 1988])

$$\Omega \left(\frac{n}{pB} \left(\frac{\log(\min\{(n/p), p\})}{\log(1 + n/(pB))} + 1 \right) \right)$$

- There is an optimal NA algorithm for MT for each $M(p, B)$: parallel version of the CA counterpart [Aggarwal, Vitter, 1988]
- There is an optimal NO algorithm for MT for each $M(p, B)$ that satisfies the n -TBA [Bilardi et al.]

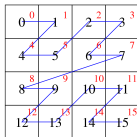
Network-Oblivious Algorithm for MT

- The naïve one-step algorithm does not exploit block transfer
- Two-step algorithm based on Z-Morton order, designed for $M(n)$
- **Red:** PE's id; **Black:** A 's entries

0 ⁰	1 ¹	2 ²	3 ³
4 ⁴	5 ⁵	6 ⁶	7 ⁷
8 ⁸	9 ⁹	10 ¹⁰	11 ¹¹
12 ¹²	13 ¹³	14 ¹⁴	15 ¹⁵

Network-Oblivious Algorithm for MT

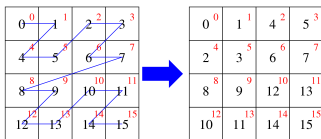
- The naïve one-step algorithm does not exploit block transfer
- Two-step algorithm based on Z-Morton order, designed for $M(n)$
- **Red:** PE's id; **Black:** A 's entries



Consider the Z-order of the matrix

Network-Oblivious Algorithm for MT

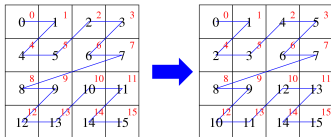
- The naïve one-step algorithm does not exploit block transfer
- Two-step algorithm based on Z-Morton order, designed for $M(n)$
- **Red:** PE's id; **Black:** A 's entries



Transform the Z-order in a
row-major order

Network-Oblivious Algorithm for MT

- The naïve one-step algorithm does not exploit block transfer
- Two-step algorithm based on Z-Morton order, designed for $M(n)$
- **Red:** PE's id; **Black:** A 's entries

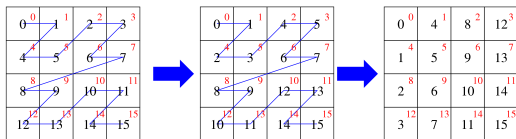


Transform the Z-order in a row-major order

Consider the Z-order of the new matrix

Network-Oblivious Algorithm for MT

- The naïve one-step algorithm does not exploit block transfer
- Two-step algorithm based on Z-Morton order, designed for $M(n)$
- **Red:** PE's id; **Black:** A 's entries



Transform the Z-order in a row-major order

Transform the Z-order in a column-major order

- Under n -TBA, optimal communication complexity

$$\Theta\left(\frac{n}{Bp}\right)$$

What Are We Going to Prove?

Theorem

*There cannot exist an **optimal** full NO algorithm for matrix transposition for every value of $1 \leq p \leq n$ and $B \geq 1$*

Definition

Full $M(p, B)$ -algorithm: in each superstep each PE sends/receives the same number of blocks, with each block containing $\Theta(B)$ words

Limits of NO Matrix Transposition

- \mathcal{A} : optimal NO algorithm for MT **without n -TBA**
- H_1 : communication complexity on $M(p_1, B_1)$
- H_2 : communication complexity on $M(p_2, B_2)$, $p_1 > p_2$
- Every data communicated in $M(p_2, B_2)$ is communicated in $M(p_1, B_1)$, too

$$B_1 p_1 H_1 \in \Omega(B_2 p_2 H_2) \Rightarrow \frac{B_1 p_1 H_1}{B_2 p_2 H_2} \in \Omega(1)$$

Limits of NO Matrix Transposition (2)

- Suppose $M(p_1, B_1)$ satisfies the n -TBA, but $M(p_2, B_2)$ does not
- If $B_2 \in \Theta\left(\frac{n}{p_2}\right)$ and $p_2 \in \Theta(n^\epsilon)$, for a constant $0 < \epsilon < 1$:

$$\frac{B_1 p_1 H_1}{B_2 p_2 H_2} \in \Theta\left(\frac{1}{\log n}\right) \in o(1)$$

Contradiction!!!

There cannot exist an **optimal** full NO algorithm for MT for every value of p and B



Conclusions

- There is a separation between aware and oblivious algorithms
- Optimal oblivious algorithms require sometime some assumptions
 - Tall Cache Assumption for CO algorithms
 - n -Tall Block Assumption for NO algorithms
- We showed that for MT:
 - **There exists** an optimal CA/NA algorithm for each value of model's parameters
 - **There cannot exist** an optimal CO/NO algorithm for each value of model's parameters

Are the Assumptions Strong or Weak?

- Tall cache assumption:
 - Real caches are usually tall; *i.e.* $M = 512$ Kbyte, $B = 32$ byte
 - Translation Lookaside Buffers (TLB) are usually not; *i.e.* if a TLB is considered as a cache, $M/B = 512$, $B = 4$ Kbyte
- n -Tall block assumption:
 - Generally satisfied by real platforms