Network-oblivious algorithms

Gianfranco Bilardi, Andrea Pietracaprina,
Geppino Pucci and **Francesco Silvestri**

DEPARTMENT OF INFORMATION ENGINEERING
UNIVERSITY OF PADOVA

Overview

- Motivation
- Framework for network-oblivious algorithms
- Case studies:
  - Network-oblivious optimal algorithms
  - An impossibility result
- Conclusions

Communication cost

- Communication heavily affects the efficiency of parallel algorithms
- Communication costs depend on interconnection topology and other machine-specific characteristics
- Models of computation for parallel algorithm design aim at striking some balance between portability and effectiveness

Models of parallel computation

PRAM

BSP, QSM
Decomposable-BSP
LogP …

Fat-Tree,
Pruned Butterfly,
Mesh

Parallel slackness **Bandwidth-latency** **Universality**

\+ Portability -

\- Effectiveness +

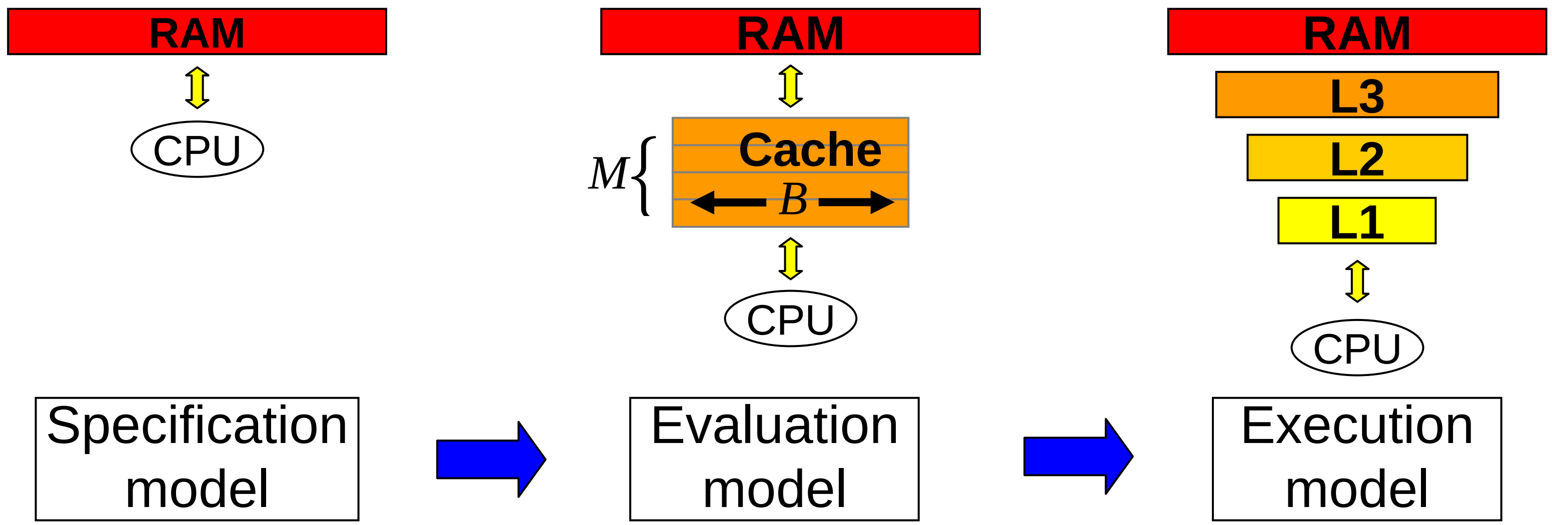
Obliviousness

- Broad consensus on bandwidth-latency models:
  - Parameters capture relevant machine characteristics
  - Logarithmic number of parameters sufficient to achieve high effectiveness (e.g., D-BSP) [Bilardi *et al.*, 99]

  - **Question:** Can we design efficient parallel algorithms oblivious to any machine/model parameters?

Cache-oblivious algorithms

- Cache-oblivious framework [Frigo *et al.*, 99]



- Parameters M, B not used for algorithm design
- Optimality in a cache-RAM hierarchy implies optimality in a multilevel cache hierarchy

Our results

- Notion of network-oblivious algorithm
- Framework for design, analysis, and execution of network-oblivious algorithms
- Network-oblivious algorithms for case study applications (matrix multiplication and transposition, FFT, sorting)
- Impossibility result for matrix transposition

Framework for network-oblivious algorithms

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

↓

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

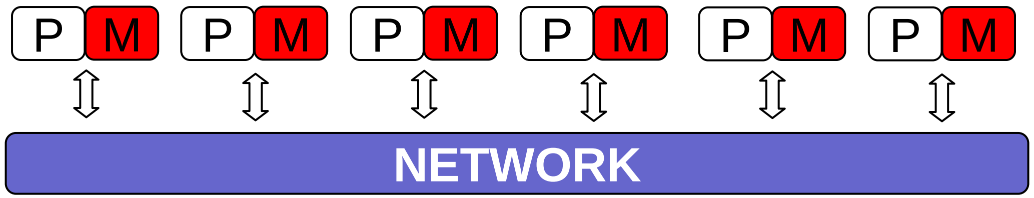
↓

Execution model: introduces hierarchical network structure

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

Specification model

- Specification model $\mathrm{M}(n)$:



  - n Processing Elements (PEs)
  - An algorithm $\mathcal{A}$ is a sequence of supersteps
  - In an superstep, each PE can:
    - Perform operations on local data
    - Send/receive messages to/from PEs
  - Note that $\mathrm{M}(n)$ is a BSP [Valiant, 90] with no bandwidth and latency parameters

Network-oblivious algorithm

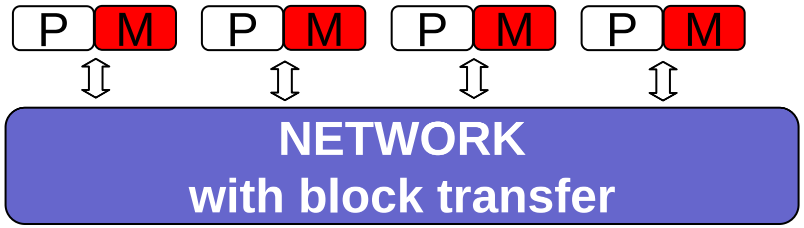
Definition: A network-oblivious algorithm for a problem Π is an $\mathrm{M}(n)$-algorithm where n is a function of the input size

Remarks: 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 a 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 $\mathcal{A}$: $\sum_{\forall s \text{ of } \mathcal{A}} h^s(p, B)$

Evaluation model (2)

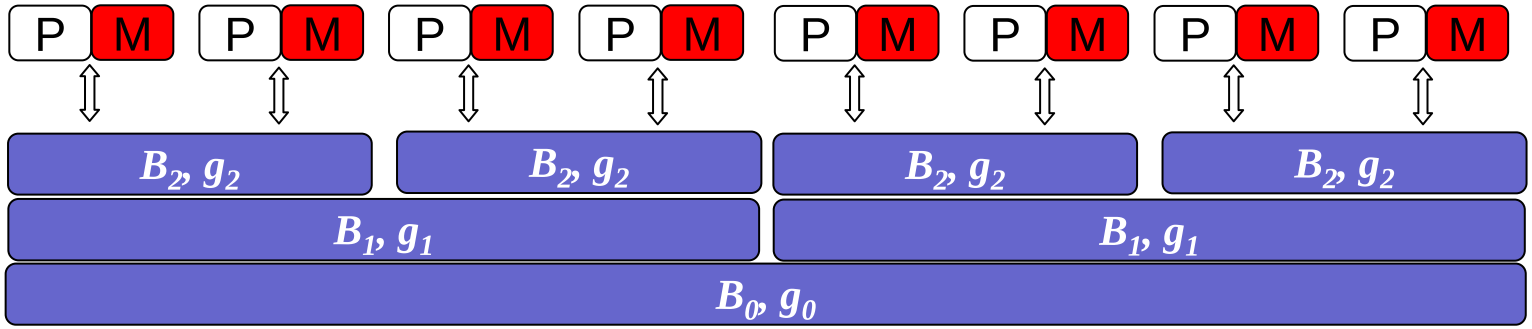
Execution of an $\mathrm{M}(n)$-algorithm on an $\mathrm{M}(p, B)$:

- Every PE of $\mathrm{M}(p, B)$ simulates a segment of n/p consecutive PEs of $\mathrm{M}(n)$
- Communications between PEs of $\mathrm{M}(n)$ in the same segment $\Rightarrow$ local computations in $\mathrm{M}(p, B)$.

Definition: A network-oblivious algorithm $\mathcal{A}$ for Π is optimal if, $\forall$ instance of size n and $\forall$ $p \leq n$ and $B \geq 1$, the execution of $\mathcal{A}$ on an $\mathrm{M}(p, B)$ yields an algorithm with asymptotically minimum communication complexity among all $\mathrm{M}(p, B)$-algorithms for Π

Execution model

- Execution model D-BSP(p, $\boldsymbol{g}$, $\boldsymbol{B}$) [De la Torre *et al.*, 96]:
  - p Processing Elements (PEs)
  - Recursive decomposition into i-clusters of $p/2^i$ PEs, $0 \leq i < \log p$
  - An algorithm $\mathcal{A}$ is a sequence of labeled supersteps
  - In an i-superstep, a PE can:
    - Perform operations on local data
    - Send/receive messages to/from PEs in its i-cluster



Execution model (2)

- A D-BSP(p, $\boldsymbol{g}$, $\boldsymbol{B}$) is an M(p, $\cdot$) with a hierarchical network structure
- $\boldsymbol{g}=(g_0, \ldots, g_{\log p-1})$, $\boldsymbol{B}=(B_0, \ldots, B_{\log p-1})$:
  - $g_i \Rightarrow$ reciprocal of the bandwidth in an i-cluster
  - $B_i \Rightarrow$ block size for communications in an i-cluster
- Communication time of an i-superstep: $h^s(p, B_i)g_i$
- Communication time of $\mathcal{A}$: $\sum_{\forall s \text{ of } \mathcal{A}} h^s(p, B_i)g_i$
- **Remark**: an M(p, $\cdot$)-algorithm can be naturally translated in a D-BSP(p, $\boldsymbol{g}$, $\boldsymbol{B}$)-algorithm by suitably labeling each superstep

Optimality result

Theorem: an optimal network-oblivious algorithm $\mathcal{A}$ exhibits an asymptotically optimal communication time when executed on a $\text{D-BSP}(p, \boldsymbol{g}, \boldsymbol{B})$ with $p \leq n$ under the following conditions:

- Wiseness: for each superstep of $\mathcal{A}$, its communications are either ***almost all local*** or ***almost all non-local*** w.r.t. $\text{D-BSP}(p, \boldsymbol{g}, \boldsymbol{B})$ PEs
- Fullness: all communicated blocks are ***almost full***

Remark: The actual wiseness and fullness conditions specified in the paper are less restrictive

Matrix Multiplication

- **Problem:** multiplying two $\sqrt{n} \times \sqrt{n}$ matrices, A and B.
- Initial row-major distribution of A and B among the n PEs

$$A \quad \begin{array}{|c|c|} \hline A_{11} & A_{12} \\ \hline A_{21} & A_{22} \\ \hline \end{array} \qquad\qquad \begin{array}{|c|c|} \hline B_{11} & B_{12} \\ \hline B_{21} & B_{22} \\ \hline \end{array} \quad B$$

$$\boxed{A_{11}} X \boxed{B_{11}} \; \boxed{A_{12}} X \boxed{B_{21}} \; \boxed{A_{21}} X \boxed{B_{12}} \; \boxed{A_{22}} X \boxed{B_{22}} \; \boxed{A_{11}} X \boxed{B_{12}} \; \boxed{A_{12}} X \boxed{B_{22}} \; \boxed{A_{21}} X \boxed{B_{11}} \; \boxed{A_{22}} X \boxed{B_{21}}$$

- 8 subproblems
- Solve each subproblem in parallel within a distinct segment of $n/8$ PEs

Matrix Multiplication (2)

- When executed on an $M(p, B)$:
  - Optimal communication complexity $\Theta\left(\frac{n}{Bp^{2/3}}\right)$
- By the previous theorem, this algorithm is also optimal in a $\mathrm{D\text{-}BSP}(p, \boldsymbol{g}, \boldsymbol{B})$, as long as $B_i \leq n/p$
- The algorithm requires $\Theta\left(p^{1/3}\right)$ memory blow-up, unavoidable if minimal communication is sought
- A different recursive strategy yields
  - Constant memory blow-up
  - Communication complexity $\Theta\left(\frac{n}{Bp^{1/2}}\right)$: optimal under constant memory blow-up constraint [Irony *et al.*, 04]

Matrix Transposition

- The naïve one-step algorithm doesn't exploit the block feature.
- Two-step algorithm based on Z-Morton ordering.

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

$\rightarrow$

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

$\rightarrow$

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

Transform a Z-ordering in a row-major ordering

Transform a Z-ordering in a column-major ordering

- If $B \leq \sqrt{\frac{n}{p}}$, optimal communication complexity $\Theta\left(\frac{n}{pB}\right)$

Matrix Transposition: impossibility result

- Constraint $B \leq \sqrt{\frac{n}{p}}$ reminiscent of the tall-cache assumption in [Frigo *et al.*, 99] (necessary to achieve cache-oblivious optimality for the matrix transposition problem [Silvestri, 06]).

- Can we remove the assumption on the block size? No!

Theorem: There is no network-oblivious matrix transposition algorithm such that for each $p \leq n$ and $B \leq n/p$, its execution on $\mathrm{M}(p, B)$ achieves optimal communication complexity

FFT and Sorting

- Fast Fourier Transform of n elements (FFT(n)):
  - Network-oblivious algorithm exploits the recursive decomposition of the FFT(n) dag into $\sqrt{n}$ FFT($\sqrt{n}$) subdags
  - Optimal algorithm for $p \leq n$ and $B \leq \sqrt{\frac{n}{p}}$
- Sorting of n keys:
  - Network-oblivious algorithm based on a recursive version of *Columnsort*.
  - Optimal algorithm for $p \leq n^{1-\varepsilon}$ $\forall$ constant ε and $B \leq \sqrt{\frac{n}{p}}$

Conclusions

Our contribution:

- Notion of network-oblivious algorithms:
  - Independent of the actual number of processors.
  - Independent of interconnection network topology.
- Framework for design, analysis, and execution of network-oblivious algorithms.
- Optimality: general theorem and specific results for prominent case studies

Conclusions (2)

Further research:

- Network-oblivious algorithms for other key problems
- Broaden the spectrum of machines for which network-oblivious optimality translates into optimal time
- Lower bound techniques to limit the level of optimality of network-oblivious algorithms

Thank you!