Network-Oblivious Algorithms (Reloaded)

Michele Scquizzato
Department of Information Engineering
University of Padova

Joint work with G. Bilardi, A. Pietracaprina, G. Pucci, and F. Silvestri

AlgoDEEP project, Kickoff meeting
Bertinoro, 16 April 2010
Motivations

- **communication** heavily affects the efficiency of parallel algorithms
- communication costs **depend** on interconnection topology and other machine-specific characteristics
- models of computation aim at striking some balance between *portability* and *effectiveness*
- broad consensus on bandwidth-latency models
  - parameters capture relevant machine characteristics
  - in general, efficient algorithms are parameter-aware
  - high effectiveness usually requires high number of parameters
Motivations (cont’d)

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

Preliminary work in [Bilardi et al., IPDPS ’07]
Framework for network-oblivious algorithms

**specification model**: parallelism as a function of input size, no machine parameters

↓

**evaluation model**: introduces the actual number of processors $p$ and communication latency $\sigma$

↓

**execution model**: introduces hierarchical network structure (⇒ high effectiveness)

**Request**: good performance on the evaluation model ⇒ good performance on the execution model
Framework for network-oblivious algorithms

**specification model:** parallelism as a function of input size, no machine parameters

**evaluation model:** introduces the actual number of processors $p$ and communication latency $\sigma$

**execution model:** introduces hierarchical network structure (⇒ high effectiveness)

**Request:** good performance on the evaluation model ⇒ good performance on the execution model

Analogy with the cache-oblivious framework [Frigo et al., '99]
Specification model $M(n)$

- $n$ virtual processors $\text{VP}_0, \ldots, \text{VP}_{n-1}$
- an algorithm $\mathcal{A}$ is a sequence of supersteps, separated by barriers
- in a superstep, each VP can
  - perform operations on local data
  - send/receive messages to/from VPs

Definition

A network-oblivious algorithm is an $M(n)$-algorithm, where $n$ is a function of the input size.

Remark: algorithm specification is independent of
- network topology
- actual number of processors
Evaluation model $M(p, \sigma)$

$M(p, \sigma)$ is an $M(p)$ where

- processing elements are called *processors* and denoted by $P_0, \ldots, P_{n-1}$
- a fixed additive cost $\sigma$ for each superstep $s$
- each processor simulates a segment of $n/p$ consecutive virtual processors
Definition

The communication complexity of $\mathcal{A}$ is

$$H_{\mathcal{A}}(n, p, \sigma) = \sum_{s \in \mathcal{A}} h^s_{\mathcal{A}}(n, p) + \sigma$$

where $h^s_{\mathcal{A}}(n, p)$ is the maximum number of messages sent or received by any processor in superstep $s$. 
Evaluation model $M(p, \sigma)$ (cont’d)

Definition

The communication complexity of $\mathcal{A}$ is

$$H_{\mathcal{A}}(n, p, \sigma) = \sum_{s \in \mathcal{A}} h^s_{\mathcal{A}}(n, p) + \sigma$$

where $h^s_{\mathcal{A}}(n, p)$ is the maximum number of messages sent or received by any processor in superstep $s$.

Definition

A network-oblivious algorithm $\mathcal{A}$ is $\beta$-optimal on an $M(p', \sigma')$ if $\exists \beta$ such that $\forall$ algorithm $\mathcal{B}$ and $\forall n$,

$$H_{\mathcal{B}}(n, p', \sigma') \geq \beta H_{\mathcal{A}}(n, p', \sigma').$$

Goal: algorithms optimal for wide ranges of $p$ and $\sigma$
Execution model $D - \text{BSP}(p, g, \ell)$

$D - \text{BSP}(p, g, \ell)$ [De la Torre et al., ’96]
- $p$ processors, $g = (g_0, \ldots, g_{\log p - 1})$ and $\ell = (\ell_0, \ldots, \ell_{\log p - 1})$,
- recursive decomposition into $i$-clusters of $p/2^i$ processors, with $0 \leq i < \log p$
- an algorithm $\mathcal{A}$ is a sequence of labeled supersteps
- in an $i$-superstep, each processor can
  - perform operations on local data
  - send/receive messages to/from processors in its $i$-cluster
Fundamental theorem

Theorem (optimality theorem)

Let $A$ be an $(\alpha, p^*)$-wise network-oblivious algorithm for a problem $\Pi$ specified for the $M(n)$ model, and $\sigma_0, \ldots, \sigma_{\log p^* - 1}$ be a vector of suitable non-negative values. If $A$ is $\beta$-optimal on $M(2i + 1, \sigma)$ for each $0 \leq i < \log p^*$ and $0 \leq \sigma \leq \sigma_i$, then $A$ exhibits $\alpha \beta / (1 + \alpha)$-optimal communication time when executed on any $D$-BSP $(p, g, \ell)$ where $1 < p \leq p^*$, $g_i \geq g_i + 1$, $\ell_i / g_i \geq \ell_i + 1 / g_i + 1$ for each $0 \leq i < \log p$, and $\ell_0 / g_0 \leq \rho := \min_{0 \leq r < \log p} \{2r + 1 \sigma_r / p\}$.

Definition

Let $0 < \alpha \leq 1$ and $1 < p \leq n$. A network-oblivious algorithm $A$ specified on $M(n)$ is said to be $(\alpha, p)$-wise if, for each $0 < i \leq \log p$,

$$H_A(n, 2^i, 0) \geq \alpha \frac{p}{2^i} \sum_{j=0}^{i-1} F_A^j(p).$$
Fundamental theorem

Theorem (optimality theorem)

Let \( \mathcal{A} \) be an \((\alpha, p^*)\)-wise network-oblivious algorithm for a problem \( \Pi \) specified for the \( M(n) \) model, and \( \{\sigma_0, \ldots, \sigma_{\log p^* - 1}\} \) be a vector of suitable non-negative values. If \( \mathcal{A} \) is \( \beta \)-optimal on \( M(2^{i + 1}, \sigma) \) for each \( 0 \leq i < \log p^* \) and \( 0 \leq \sigma \leq \sigma_i \), then \( \mathcal{A} \) exhibits \( \alpha \beta/(1 + \alpha) \)-optimal communication time when executed on any \( D - \text{BSP}(p, g, \ell) \) where \( 1 < p \leq p^* \), \( g_i \geq g_{i+1} \), \( \ell_i/g_i \geq \ell_{i+1}/g_{i+1} \) for each \( 0 \leq i < \log p \), and \( \ell_0/g_0 \leq \rho := \min_{0 \leq r < \log p} \{2^{r+1}\sigma_r/p\} \).

Definition

Let \( 0 < \alpha \leq 1 \) and \( 1 < p \leq n \). A network-oblivious algorithm \( \mathcal{A} \) specified on \( M(n) \) is said to be \((\alpha, p)\)-wise if, for each \( 0 < i \leq \log p \),

\[
H_\mathcal{A}(n, 2^i, 0) \geq \alpha \frac{p}{2^i} \sum_{j=0}^{i-1} F^j_\mathcal{A}(p).
\]
Fundamental theorem

Theorem (optimality theorem)

Let $A$ be an $(\alpha, p^*)$-wise network-oblivious algorithm for a problem $\Pi$ specified for the $M(n)$ model, and $\{\sigma_0, \ldots, \sigma_{\log p^* - 1}\}$ be a vector of suitable non-negative values. If $A$ is $\beta$-optimal on $M(2^i + 1, \sigma)$ for each $0 \leq i < \log p^*$ and $0 \leq \sigma \leq \sigma_i$, then $A$ exhibits $\alpha\beta/(1 + \alpha)$-optimal communication time when executed on any D-BSP($p, g, \ell$) where $1 < p \leq p^*$, $g_i \geq g_{i+1}$, $\ell_i/g_i \geq \ell_{i+1}/g_{i+1}$ for each $0 \leq i < \log p$, and $\ell_0/g_0 \leq \rho := \min_{0 \leq r < \log p} \{2^{r+1}\sigma_r/p\}$.

Definition

Let $0 < \alpha \leq 1$ and $1 < p \leq n$. A network-oblivious algorithm $A$ specified on $M(n)$ is said to be $(\alpha, p)$-wise if, for each $0 < i \leq \log p$,

$$H_A(n, 2^i, 0) \geq \alpha \frac{p}{2^i} \sum_{j=0}^{i-1} F_{\lambda_A}^j(p).$$

Remark: optimality on D-BSP $\implies$ optimality for several common topologies, including $d$-dimensional arrays [Bilardi et al., ’99]
Matrix Transposition

**Problem:** transposing an $\sqrt{n} \times \sqrt{n}$ matrix, with entries evenly distributed among the VPs according to a row-major ordering

Accomplished in a single 0-superstep by the trivial algorithm

The algorithm exhibits optimal $\Theta(n/p + \sigma)$ communication complexity on $M(p, \sigma)$ for each $1 < p \leq n$ and $\sigma \geq 0$
Matrix Multiplication

Problem: multiplying two $\sqrt{n} \times \sqrt{n}$ matrices

Algorithm: solve each of the above 8 subproblems in parallel within a distinct segment of $n/8$ processors

Communication complexity on an $M(p, \sigma)$

$$H_{MM}(n, p, \sigma) = O\left(\frac{n}{p^{2/3}} + \sigma \log p\right),$$

optimal for each $1 < p \leq n$ and $\sigma = O\left(n/(p^{2/3} \log p)\right)$. 
Problem: Fast Fourier Transform of $n$ elements ($\text{FFT}(n)$)

Algorithm: exploit the recursive decomposition of the $\text{FFT}(n)$ DAG into $\sqrt{n}$ $\text{FFT}(\sqrt{n})$ subDAGs

Communication complexity on an $M(p, \sigma)$

$$H_{\text{FFT}}(n, p, \sigma) = O\left(\left(\frac{n}{p} + \sigma\right) \frac{\log n}{\log(n/p)}\right),$$

optimal for each $\sigma \geq 0$ if $p = O\left(n^{1-\epsilon}\right)$, with $0 < \epsilon < 1$ being an arbitrary constant; for each $\sigma \in O\left(n/p\right)$ otherwise.
Problem: sorting of $n$ keys

Algorithm: recursive version of Columnsort

Communication complexity on an $M(p, \sigma)$

$$H_{\text{Sort}}(n, p, \sigma) = O \left( \left( \frac{n}{p} + \sigma \right) \left( \frac{\log n}{\log(n/p)} \right)^{\log_3 4} \right),$$

optimal for each $\sigma \geq 0$ if $p = O \left( n^{1-\epsilon} \right)$, with $0 < \epsilon < 1$ being an arbitrary constant.
Network simulations

(Meta)Problem: simulation of a network topology; let us focus first on a linear array
Network simulations

(Meta)Problem: simulation of a network topology; let us focus first on a linear array

Algorithm 1: 2-way recursion [Prokop ’99]
Network simulations

(Meta)Problem: simulation of a network topology; let us focus first on a linear array

Algorithm 1: 2-way recursion [Prokop ’99]
Network simulations

(Meta)Problem: simulation of a network topology; let us focus first on a linear array

Algorithm 1: 2-way recursion [Prokop ’99]
Network simulations

(Meta)Problem: simulation of a network topology; let us focus first on a linear array

Algorithm 1: 2-way recursion [Prokop '99]
Network simulations

(Meta)Problem: simulation of a network topology; let us focus first on a linear array

Algorithm 1: 2-way recursion [Prokop ’99]

Communication complexity on an $M(p, \sigma)$

$$H_{2\text{-way}}(n, p, \sigma) = O \left( n p^{\log 3/2} + p^{\log 3} \sigma \right),$$

optimal only when $p = O(1)$. 
Network simulations (cont’d)

Algorithm 2: $\sqrt{n}$-way recursion

Communication complexity on an $M(p, \sigma)$

$$H_{\sqrt{n}-\text{way}}(n, p, \sigma) = O \left( (n + (p + \sqrt{n})\sigma) \frac{\log n}{\log(n/p)} \right),$$

optimal for each $\sigma = O \left( \min\{\sqrt{n}, n/p\} \right)$ if $p = O \left( n^{1-\epsilon} \right)$, with $0 < \epsilon < 1$ being an arbitrary constant.
Network simulations (cont’d)

Algorithm 2: $\sqrt{n}$-way recursion

Communication complexity on an $M(p, \sigma)$

$$H_{\sqrt{n}-\text{way}}(n, p, \sigma) = O\left(\left(n + (p + \sqrt{n})\sigma\right) \frac{\log n}{\log(n/p)}\right),$$

optimal for each $\sigma = O\left(\min\{\sqrt{n}, n/p\}\right)$ if $p = O\left(n^{1-\epsilon}\right)$, with $0 < \epsilon < 1$ being an arbitrary constant.

Key observation: values of $p$ for which optimality is attained $\sim$ degree of recursion

Algorithm 3: degree of recursion increases as the recursion unfolds

Communication complexity on an $M(p, \sigma)$

$$H(n, p, \sigma) = O\left(n + p\sigma\right),$$

optimal for each $1 < p \leq n$ and $\sigma = O\left(n/p\right)$.

Same idea successfully applies to the simulation of a mesh
An impossibility result

Problem: broadcast of a datum

Observation: let $p$ be fixed; the number of supersteps of an oblivious algorithm cannot vary with $\sigma$

Theorem

*There cannot exist a network-oblivious algorithm for $n$-broadcast which is optimal on every $M(p, \sigma)$ with fixed $p$ and $\sigma \in [\sigma_1, \sigma_2]$, unless $\log \sigma_2 = \Theta(\log \sigma_1)$.***
Conclusions

Our contribution

▶ (revised) notion of network-obliviousness
▶ (revised) framework for design, analysis, and execution of network-oblivious algorithms
▶ analysis of network-oblivious algorithms for prominent case studies

Further research

▶ network-oblivious algorithms for other key problems
▶ broaden the spectrum of machines for which network-oblivious optimality translates into optimal time
▶ explore the role of wiseness
▶ lower bound techniques to limit the level of optimality of network-oblivious algorithms

network- + cache-obliviousness = machine-obliviousness
Conclusions

Our contribution

▶ (revised) notion of network-obliviousness
▶ (revised) framework for design, analysis, and execution of network-oblivious algorithms
▶ analysis of network-oblivious algorithms for prominent case studies

Further research

▶ network-oblivious algorithms for other key problems
▶ broaden the spectrum of machines for which network-oblivious optimality translates into optimal time
▶ explore the role of wiseness
▶ lower bound techniques to limit the level of optimality of network-oblivious algorithms
▶ network- + cache-obliviousness = machine-obliviousness