Communication Lower Bounds for Distributed-Memory Computations

Francesco Silvestri

Department of Information Engineering
University of Padova

Joint work with Michele Scquizzato (University of Pittsburgh)

STACS 2014
Lyon, France, March 8, 2014

Full version ArXiv: 1307.1805

Motivation

- Massive data sets almost always need to be processed on parallel machines
  - Revival of parallel computing in the big-data era
- Communication among processors is the major bottleneck
  - Time and energy for transferring data are significantly higher than that for performing arithmetic operations
- General quest for lower bounds for complexity of communications
  - Allow to evaluate the distance from optimality
  - In general, obtained under restrictive assumptions

Assumptions

- Good assumptions: without them, game rules completely change
  - E.g.: matrix multiplication with semiring: $\Theta\left(n^3\right)$ operations
  - E.g.: matrix multiplication with ring: $\Omega\left(n^2\right)$ operations
  - From an upper bound point of view: breaking hypotheses may allow to beat lower bounds
- Bad assumptions: the proof significantly simplifies
  - Input power of two
  - Property of the input (evenly distribution, ...)
  - From a lower bound point of view: hypotheses limit the applicability of the bound
- It is not easy to distinguish between good and bad assumptions!

The model

- We seek lower bounds to the communication complexity on the BSP model
- The BSP model [Valiant, Comm. ACM '90]:
  - p processing elements, each with unbounded local memory
  - Superstep-style program execution
  - Cost of communications
    - $h_s(n, p)$: max number of messages sent or received by any processor in superstep s
    - Communication complexity: $H(n, p) = \sum_s h_s(n, p)$
  - No latency cost

Our results

- We revisit assumptions of previous lower bounds to the communication complexity of several key computational problems
  - Matrix multiplication, stencil computations, sorting, FFT
- We prove new lower bounds with weaker assumptions
  - Lower bounds have the same functional form
  - but have a wider applicability

Are we Happy with Existing Lower Bounds?

Existing lower bounds are derived under some of the following hypotheses:

1. Inputs initially reside outside processors' local memories
   - More in the spirit of shared-memory models: in distributed-memory machines, inputs initially reside in local memories
   - "Hack" to obtain an easy $\Omega(n/p)$ lower bound

Are we Happy with Existing Lower Bounds?

Existing lower bounds are derived under some of the following hypotheses:

1. Inputs initially reside outside processors' local memories
   - More in the spirit of shared-memory models: in distributed-memory machines, inputs initially reside in local memories
   - "Hack" to obtain an easy $\Omega(n/p)$ lower bound
2. Inputs are initially evenly distributed among the p processors
   - Initial distribution of inputs is usually not fixed

Are we Happy with Existing Lower Bounds?

Existing lower bounds are derived under some of the following hypotheses:

1. Inputs initially reside outside processors' local memories
   - More in the spirit of shared-memory models: in distributed-memory machines, inputs initially reside in local memories
   - "Hack" to obtain an easy $\Omega(n/p)$ lower bound
2. Inputs are initially evenly distributed among the p processors
   - Initial distribution of inputs is usually not fixed
3. Computational load is evenly distributed among the p processors
   - Assumes (but does not prove) that optimal solutions balance computation

Are we Happy with Existing Lower Bounds?

Existing lower bounds are derived under some of the following hypotheses:

1. Inputs initially reside outside processors' local memories
   - More in the spirit of shared-memory models: in distributed-memory machines, inputs initially reside in local memories
   - "Hack" to obtain an easy $\Omega(n/p)$ lower bound
2. Inputs are initially evenly distributed among the p processors
   - Initial distribution of inputs is usually not fixed
3. Computational load is evenly distributed among the p processors
   - Assumes (but does not prove) that optimal solutions balance computation
4. Processors' local memories are bounded
   - The local memory can be very large (disks are chip)

Our Approach

- Our main hypothesis: no processor performs more than a constant fraction of the total required work
- Formally:
  - W_0 = total required work
  - W = maximum amount of work performed by any processor

$$W \leqslant \epsilon W_0, \text{ for some constant } \epsilon \in (0, 1)$$

Rationale:

Consider all possible parallel algorithms, excluding (nearly) sequential ones (in which case the bottleneck is computation rather than communication)

Our Approach (2)

- Some lower bounds also require
  - Limited input replication or no recomputation
  - These assumptions also required in previous lower bounds!

- We do not require
  - Load balance
  - Specific distribution of inputs or outputs
  - Bounded memories

Matrix Multiplication

- Standard (i.e., $O\left(n^3\right)$) multiplication of two $n \times n$ matrices
- Several $\Omega\left(n^2/p^{2/3}\right)$ bounds under hypothesis 1), 2), 3), or 4)

Theorem

If $W \leqslant \max\{n^3/p, n^3/11^3\}$, and the input matrices are not initially replicated, then

$$H(n,p) = \Omega\left(W^{2/3}\right).$$

- Good news:
  - Apply for $W \leqslant \epsilon n^3$, with $\epsilon \in (0,1)$
  - Support small input replication
- Minimum bound when $W = n^3/p$

Proof for Matrix Multiplication

- Consider the processor performing work W.
- If this processor initially holds few input values, then it must receive many input from other processors since it computes at least n^3/p multiplicative terms

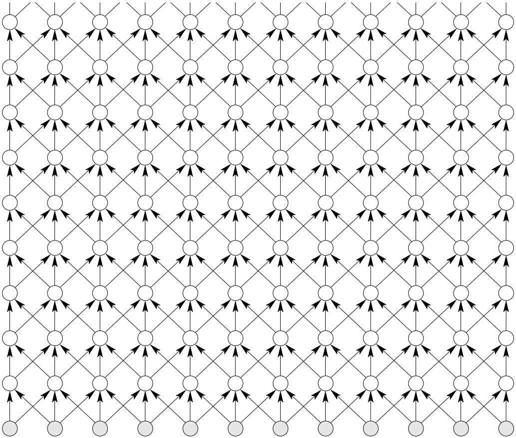
  Holds few input: $I \leqslant W^{2/3}/5$
  Receive many input: $H \geqslant W^{2/3} - I = \Omega\left(W^{2/3}\right)$

- Otherwise, if it initially holds many inputs, then it has to send many of them to the other processors since it cannot perform too much work on its own

  Holds many input: $I > W^{2/3}/5$
  Send many input: $H \geqslant I - W/n \geqslant \Omega\left(W^{2/3}\right)$

Stencil Computations

- Computation of d-dimensional grid-like structures



Stencil Computations

- Tight $\Omega(n)$ lower bound already known for $d = 2$; for $d \geqslant 3$, tight bound known only under hypothesis 3)

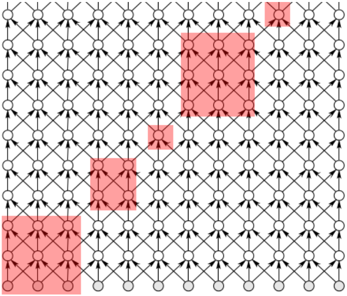
Theorem

If $W \leqslant \epsilon n^d$, for an arbitrary constant $\epsilon \in (0, 1)$, and recomputation is disallowed, then

$$H_d(n, p) = \Omega\left(\frac{n^{d-1}}{p^{(d-2)/(d-1)}}\right).$$

Proof for Stencil Computations ($d = 2$)

- Highlight a sequence of squares
- Each square communicates messages proportional to the perimeter
- Sum of square sizes is almost the length of the main diagonal
- Communication minimized when all squares have size n/p
- Main issues:
  - d dimensions
  - Squares may have very different sizes if work is unbalanced



Sorting

- Comparison-based sorting of n elements
- Tight $\Omega\left(n \log n/(p \log(n/p)\right)$ lower bounds under hypothesis 1) or 2)

Theorem

If $W \leqslant \epsilon(n \log n)$ for an arbitrary constant $\epsilon \in (0,1)$, the inputs are not initially replicated, and the p processors store only a constant number of copies of any key at any time instant, then

$$H(n,p) = \Omega\left(\frac{n \log n}{p \log(n/p)}\right).$$

Proof for Sorting

- Based on counting arguments on the number of permutations distinguished by the algorithm in superstep
- Assume each processor contains S inputs
- Communication complexity:

$$H(n, p) = \Omega\left(\frac{n \log(n/S)}{p \log(n/p)} + S\right).$$

- Communication complexity minimized when $S = N/p$

Fast Fourier Transform

- Computation of the $n \log n$-nodes FFT DAG
- Tight $\Omega\left(n \log n / (p \log(n/p)\right)$ lower bounds under hypothesis 1) or 2)

Theorem

If $W \leqslant \epsilon(n \log n)$ for an arbitrary constant $\epsilon \in (0, 1)$, recomputation is disallowed, and the inputs are not initially replicated, then

$$H(n, p) = \Omega\left(\frac{n \log n}{p \log(n/p)}\right).$$

Proof for Fast Fourier Transform

- W maximum number of FFT nodes evaluated by a processor
- When $W \geqslant (n \log n)/p$, we prove a $\Omega\left(\frac{W}{\log W}\right)$ lower bound (bandwidth argument)
- Otherwise, we exploit the lower bound $\Omega\left(\frac{n \log(n/U)}{p log(n/p)}\right)$ where $U \leqslant W$ is the maximum number of output nodes evaluated by a processor.

Our Contribution & Open Problems

Our Contribution

- Proposed new approach in communication lower bounds for distributed-memory computations
- New tight lower bounds of wider applicability

Our Contribution & Open Problems

Our Contribution

- Proposed new approach in communication lower bounds for distributed-memory computations
- New tight lower bounds of wider applicability

Open Problems

- Further relax hypotheses under which lower bounds are proved (replication/recomputation?)
- Application to other models of computation
- Unified theory of lower bound techniques for communication complexity