Cache-Oblivious Simulation of Parallel Programs

Andrea Pietracaprina    Geppino Pucci    Francesco Silvestri

Bertinoro, February 17-18th 2006
Outline of the talk

1. Problem definition
2. Computational Models
3. Technical results
4. Conclusions
Architectural trends

- Modern parallel architectures:
  - Network hierarchies: communication costs depend on the processors involved;
  - Memory hierarchies: access costs depend on the level of memory involved.

- Examples:
  - IBM BlueGene/L;
  - IBM SP5.
Locality

- **Temporal locality of reference:** the same data are frequently reused within a short time interval;

- **Spatial locality of reference:** data stored at consecutive addresses are involved in consecutive operations;

- **Submachine locality:** communications are confined within small submachines featuring high bandwidth and small latency.
Objective

- Study the relation between submachine locality (SL) in network hierarchies and locality of reference (LR) in memory hierarchies.
Previous work

Vishkin 94, 02: Flat Parallelism PRAM ⇒ Cache prefetching strategies;
Dehne et al. 97, 99: Bulk Parallelism (BSP, CGM) ⇒ Efficient External Memory (EM) algorithms;
FPP 02÷05 Structured parallelism (D-BSP) ⇒ Efficient HMM and BT algorithms.

Remarks

- Works based on flat parallelism (PRAM, BSP) can’t be extended to memory hierarchies with more than two levels.
- EM, HMM, BT models have explicit control over the hierarchy.
Our results

- **Parallel model:** Decomposable Bulk Synchronous Parallel Model (D-BSP) [De la Torre, Kruskal, 96];
  - Rewards Submachine Locality.

- **Sequential model:** Ideal Cache Model (ICM) [Frigo et al., 99];
  - Rewards Temporal Locality;
  - Rewards Spatial Locality;
  - Cache is hardware controlled.

Results

- Automatic and cache oblivious simulation of D-BSP algorithms on ICM;
- Efficient ICM (cache oblivious) algorithms obtained from efficient D-BSP algorithms.
Ideal Cache Model

- Only two levels of memory: cache, RAM memory;
- Parameters:
  - \( Z \): cache size;
  - \( L \): cache line size;
- Cache features:
  - Fully associative;
  - Optimal offline strategy for cache line replacement;
  - Tall cache hypothesis \( Z = \Omega(L^2) \).
An algorithm is characterized by:
- **Work complexity** $W(N, Z, L)$: number of CPU operations;
- **Cache complexity** $Q(N, Z, L)$: number of cache miss.

**Definition**
An algorithm is *cache oblivious* if its specification is independent of the two parameters $Z$ and $L$. An algorithm is *cache aware* otherwise.
D-BSP

- \( N \) processor-RAM memory pairs.
- Recursive decomposition into \( i \)-clusters of \( N/2^i \) processor-RAM memory pairs, \( 0 \leq i < \log N \).
- \( i \)-clusters work independent in \( i \)-supersteps.
- Parameters of D-BSP:
  - \( g_i \): inverse measure of \( i \)-cluster bandwidth; usually \( g_i \geq g_{i+1} \).
  - \( h \): upper bound to number of messages sent/received by a processor.
  - \( \mu \): upper bound to the local memory used by a processor during the execution of a D-BSP program.

\[
\begin{align*}
g_0 &= g(8\mu) \\
g_1 &= g(4\mu) \\
g_2 &= g(2\mu) \\
g_2 &= g(2\mu) \\
g_2 &= g(2\mu) \\
g_2 &= g(2\mu)
\end{align*}
\]
How can an $i$-superstep for an $i$-cluster $C$ be simulated?

Execution of $C$’s local contexts:
- if $i = \log N$: the unique context of $C$ is simulated;
- If $i < \log N$: the contexts of the two $i + 1$-clusters of $C$ are recursively simulated.

Communications between $C$’s processors:
- the contexts of $C$ are partitioned into constant size elements;
- each element is tagged with a suitable key;
- the elements are sorted by a \textit{cache-oblivious} algorithm.
Remarks

- [FPP 02÷05]: explicit movements of data through the hierarchy.
- This work: indirect movement of data through the hierarchy regulates by order of accesses to clusters/contexts.

Which cluster will be simulated in next round?
Let $C$ be the $i$-cluster that has been just simulated and $j$ be the next superstep index to be executed by processors in $C$;

If $i \leq j$, simulate the first $j$-subcluster of $C$;
Next round (Cont’d)

- If $i > j$ and $C$ is the last $i$-subcluster contained in the $j$-cluster $C'$, then simulate $C'$,

- otherwise simulate the next sibling of $C$. 
Complexity

Theorem

Consider a D-BSP program, with contexts of size $\mu$ and aggregate time for local computations $\tau$. Let $k_i$ be the number of $i$-supersteps, $0 \leq i < \log N$. The work and cache complexities of the simulation are:

$$W(N, Z, L) = O \left( \tau + \mu N \sum_{i=0}^{\log N-1} k_i \log \frac{\mu N}{2^i} \right),$$

$$Q(N, Z, L) = O \left( \sum_{i=0}^{\lambda-1} k_i \frac{\log \frac{\mu N}{2^i}}{\log Z} \right).$$

for a suitable $\lambda$. 
Complexity (Cont’d)

- $\lambda$: index of the largest cluster whose contexts fit in cache.
- All misses due to simulation of $i$-clusters with $i \geq \lambda$ are negligible!
- Applications:
  - Matrix Multiplication:
    - $W(N, Z, L) = O(N^{3/2})$
    - $Q(N, Z, L) = O\left(\frac{N^{3/2}}{L\sqrt{Z}}\right)$
    \[\Rightarrow \text{OPTIMAL}\]
  - DFT:
    - $W(N, Z, L) = O(N \log N \log \log N)$
    - $Q(N; Z, L) = O\left(\frac{N \log N \log \log Z}{L \log Z}\right)$
    \[\Rightarrow \text{QUASI-OPTIMAL}\]
Discussion

- The simulation is cache oblivious.
- The simulation automatically gives efficient cache oblivious algorithms from D-BSP ones.
- The complexities remain asymptotically unchanged under LRU.
- Extends to multilevel hierarchies [Frigo et al., 99].
- The double logarithmic slowdown for DFT is due to the generality of the simulation algorithm (arbitrary communication patterns).
Better approaches can be employed with regular communication patterns (*ad-hoc simulations*).

Optimal cache-oblivious ad-hoc DFT algorithm.

Complexity implication: ICM optimality of the simulated algorithm implies optimality of the D-BSP source algorithm.
Thank you!!!