**Have Your Cake In Parallel And Eat It Sequentially Too!**

Semantic Sequential, Parallel Execution of Multiprocessor Programs

Gagan Gupta

---

### Summary

Multiprocessors are ubiquitous, but programming them continues to be challenging. Our Goal: Simplify multiprocessor programming without compromising performance.

<table>
<thead>
<tr>
<th>Conventional Wisdom</th>
<th>Our Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Order in programs obstructs parallelism</td>
<td>Order can help to expose parallelism!</td>
</tr>
<tr>
<td>Use non-deterministic programs, or make dataflow in programs explicit</td>
<td>Use ordered programs; maintain precise program-order execution semantics</td>
</tr>
<tr>
<td>Programmer should expose parallelism</td>
<td>Use run-time dataflow and speculative techniques to expose parallelism</td>
</tr>
</tbody>
</table>

Benefits:
- Simplified programming; Simplified system design; Better reliability
- Performance at par or better (5% to 288%) than conventional methods

---

### Programmer’s Role

#### Conventional

- Develop Parallel Algorithm
- Schedule Execution of Tasks
- Reason About Inter-task Data Access Conflicts
- Ensure Independence between Parallel Tasks

#### Our Approach

- Develop Parallel Algorithm
- Reason About Task-local Data Accesses
- Program text => Order

---

### Exploiting Parallelism

#### Task Dependence Graph of Cholesky Decomposition

- Execution has to respect programmer-exposed parallelism
- Cannot schedule F5 in t2
- If dependencies are known, distant parallelism can be exposed
- Can schedule F5 in t2

---

### ParaKram

- Run-time parallel execution manager (C++ library)
- Performs out-of-order superscalar processor-like execution on multiprocessors

- Multiprocessor program
  - Precise-restart Engine
  - Dataflow Engine
  - Task Scheduler

---

### Precise-restart Engine

- Tracks tasks and their order in a Reorder List
- Checkpoints mod set in History Buffer
- Retires task in (total) program order

---

### Dataflow Engine

Uncovers parallelism past blocked tasks in the program
- Constructs dynamic data dependence graph using write and read sets
- Executes tasks out-of-order
- If task dependences/order are independent, speculates tasks are independent
- Detects and rectifies misspeculation

---

### Order-aware Load-balancing Task Scheduler

- Example speculative dataflow execution on 3 processors
- Conventional Checkpoint-and-Recovery (non-speculative)
- Dataflow => rolled back (non-speculative)
- Precisely restarting

---

### ParaKram speedup

- 39% speedup is up to 77% higher than non-deterministic OpenMP
- 75% over Cilk

---

### Applications

- ConjugateGradient
- WordCount
- RE
- Mergesort
- Swaptions
- Histogram
- Pbzip2
- BlkKmeans
- Blackscholes
- Tree
- PI
- ScaLAPACK
- CSparse
- Soundex
- Blas
- 1000BLAS
- 1000 Lanczos

---

### Conclusions

- ParaKram scales with system size; Non-deterministic method does not scale
- ParaKram speedup is up to 77% higher than non-deterministic Cilk and TLL STM