DMP
Deterministic Shared Memory Multiprocessing

Joe Devietti, Brandon Lucia, Luis Ceze, Mark Oskin
DMP Guarantees

- DMP provides execution-level determinism for arbitrary programs
- Compared to language-level determinism, programmer has no control over which deterministic program he gets
- DMP is a hardware proposal
  - Tom Bergan’s CoreDet presentation tomorrow shows compiler version
The DMP Approach

- We only care about communicating instructions
- Deterministic serialization → same communication
  - ...but what about performance?
- Recover parallelism from non-communicating insns
Talk Outline

DMP: Deterministic Shared Memory Multiprocessing – WODET 2009
DMP-Serial Example
Can we do better?

- Communicating insns cause cache line state transitions
- Break each quantum into communication-free \textit{parallel prefix} and communicating \textit{serial suffix}
DMP-Ownership

- Need to know when communication happens, to transition from parallel to serial mode
  - Leverage existing cache coherence protocol
  - When a line changes state, communication is (potentially) happening!
  - The Memory Ownership Table (MOT) tracks information about ownership

- State of MOT must evolve deterministically
  - Only allow updates during serial suffix
DMP-Ownership Example
DMP-TM: Recovering Parallelism with Speculation

- DMP-Ownership conservatively assumes that all cache line state transitions are communication
  - ...but many transitions are not
- Use TM support to speculate that a quantum is not involved in communication
  - If communication happens, rollback + re-execute
  - Commit quanta in-order (need DT to commit)
DMP-TM Example
DMP-TM-Forward:
Speculative Value Forwarding

- DMP-TM eliminates WAW and WAR dependencies
  - but cannot speculate past true (RAW) dependences
- Idea: speculatively forward values to “future” quanta
  - ordered transactions make it easy to decide when and where to forward
  - rollback if a quantum’s speculatively read data is updated before the quantum commits
Rollbacks in DMP-TM-Forward
Better Quantum Building

- Any deterministic policy will work
- We want quanta that are free of communication
  - no communication $\rightarrow$ no serialization, no rollbacks
Experimental Methodology

- PIN-based simulator
  - Models serialization, quantum building, address conflicts and transaction rollbacks
  - Assumes constant IPC with free commits
- SPLASH2 and PARSEC benchmark suites
## Results

### Splash2 Benchmark Suite

<table>
<thead>
<tr>
<th>Threads</th>
<th>DMP-Serial</th>
<th>DMP-Ownership</th>
<th>DMP-TM</th>
<th>DMP-TMForward</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>4.3</td>
<td>6.4</td>
<td>4.6</td>
<td>6.7</td>
</tr>
<tr>
<td>8</td>
<td>4.3</td>
<td>6.4</td>
<td>4.6</td>
<td>6.7</td>
</tr>
<tr>
<td>16</td>
<td>4.3</td>
<td>6.4</td>
<td>4.6</td>
<td>6.7</td>
</tr>
</tbody>
</table>

### Parsec Benchmark Suite

- **1,000-instruction quanta**
- **Synchronization+burst quantum builder**
Conclusions

- DMP is a new multiprocessor architecture that provides execution-level determinism for arbitrary programs
  - Leverages existing architectural techniques
  - Performance very close to nondeterministic execution
- Determinism is a worthwhile and achievable goal
Also in the paper...

- Support for debugging
  - Adding instrumentation without affecting communication
- Making execution deterministic across machines
- Dealing with nondeterminism from I/O and the OS