Predicting Coherence Communication by Tracking Synchronization Points at Run Time

Socrates Demetriades
and Sangyeun Cho

University of Pittsburgh

45th International Symposium in Microarchitecture, December 2012
Communication Overheads

Directory-based Coherence Protocol

Miss

Indirect Miss to the Directory
=> Increase Miss Latency

Broadcast to all
=> Increase traffic

Snoop-based Coherence Protocol
Communication Prediction

![Diagram with nodes and arrows indicating communication and prediction]

- Miss
- Predict

Trade-Off
Accuracy vs Extra traffic
Contribution

Synchronization Point based Prediction (**SP-prediction**)

*Inter-thread communication caused by coherence transactions is tightly related with the synchronization points in parallel execution*

- **Main Idea:** Associate the communication behavior with synchronization points and utilize this association to predict the destination of misses.

- **Main Advantage:** Has very low storage cost, yet delivers relatively high performance.
Why Synchronization Points?

- BARRIER
- LOCK
- UNLOCK
- BARRIER
- SIGNAL
- WAIT
- shared data
- communication direction

[Pthread notation]
Synchronization Epochs

Communication Distribution of Core 0 (full interval)

Communication Distribution of Core 0 (different sync-epochs)

[Benchmark: Bodytrack / 16-threads]
Sync-Epoch Dynamic Instances

Communication Distribution of Core 0
(same sync-epoch in different dynamic instances)

[Benchmark: Bodytrack / 16-threads]
SP-prediction – Overview

- Monitor destinations of each miss on each core.
- Extract communication signatures for each sync-epoch.
- Store and later reuse those signatures to predict misses in future sync-epoch instances.
- When initial predictions do not exist or are inaccurate, reconstruct the signatures within the sync-epochs.

- Sync Points must be exposed to the hardware so it can sense the beginning and end of sync-epochs.
  - A dedicated instruction must be inserted at the calling location of the synchronization point.
  - PC, lock variable and type must be extracted and pass to a history table.
SP-prediction: History-based Example

SYNC-POINT A  SYNC-POINT B

CORE 0

Track Communication

A

Extract Hot Commun.Set

[hot comm. set]

C0 C1 C2 C3

Store to SP-table

SYNC-POINT A  SYNC-POINT B

Miss

[hot comm. set]

Retrieve hot core set

SP-TABLE

<table>
<thead>
<tr>
<th>Sync-Point PC</th>
<th>PREDICTOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>[hot comm. set]</td>
</tr>
</tbody>
</table>
Results: Prediction Accuracy

- **Accuracy:** 76%

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Destination Set Size (actual)</th>
<th>SP-prediction Set Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average</td>
<td>1.2</td>
<td>2.6</td>
</tr>
</tbody>
</table>
Results: Latency & Bandwidth

13%

18% (5%)
Comparison with other Predictors

![Graph showing comparison with other predictors. The x-axis represents % Additional Bandwidth per Miss, the y-axis represents % Incuring Indirection. The graph includes data points for Last 2 misses, ADDR-based, INSTR-based, SP-prediction, and DIRECTORY.]
Conclusions

• SP-prediction is a new, run-time approach on coherence communication predictability.

• In contrast to traditional hardware based temporal predictors, exploits localities based on application-defined intervals.

• Promotes very low storage requirements, an important property for emerging CMP implementations.

• Scales independent of core count and cache sizes.

• Takes advantage of the existing shared memory programming paradigm.