Adaptive SMT Control for More Responsive Web Applications

Hiroshi Inoue†‡ and Toshio Nakatani†
† IBM Research – Tokyo
‡ University of Tokyo
Response time matters!

- **Peak throughput** has been the common metric for the Web server performance

- Even sub-second improvements in response times are essential for better user experiences†
  - Amazon: +100 msec → 1% drop in sales
  - Yahoo: +400 msec → 5-9% drop in traffic
  - Google: +500 msec → 20% drop in searches

- We focus on improving the response time of Web application servers

Key Question: How SMT affects response time?

- SMT (Simultaneous Multi Threading, a.k.a. Hyper Threading) allows multiple hardware threads to run on one core

- SMT typically
  - 😊 improves aggregated throughput
  - 😞 degrades single-thread performance

→ Question: How SMT affects response times of Web application server?
Outline

1. How SMT affects response time
2. Adaptive SMT control with queuing model
Evaluations

- Processors:
  - Xeon (SandyBridge-EP): 2-way SMT, 2.9 GHz, 16 cores
  - POWER7: 4-way SMT, 3.55 GHz, 16 cores

- Workloads:
  - PHP (MediaWiki)
  - Ruby (Ruby-on-rails)
  - Java (Cognos BI)

- OS: Redhat Enterprise Linux 6.4 (Kernel-2.6.32)
Response time of the PHP application on 16 cores of Xeon

- SMT1 was better at lower CPU utilization
- SMT2 was better at higher CPU utilization

SMT degraded response time when CPUs were not fully loaded
SMT improved response time at high CPU utilization

lower CPU utilization: lower is faster
higher CPU utilization: lower is faster
Response time of the PHP application on 16 cores of POWER7

- SMT1 was best
- SMT2 was best
- SMT4 was best

SMT degraded response time when CPUs were not fully loaded

SMT1 (disabled SMT)  SMT2 (2-way SMT)  SMT4 (4-way SMT)
Response time of the PHP application on 1 core of Xeon

SMT2 was better

lower is faster

SMT improved response time even when CPUs were not fully loaded

lower CPU utilization

higher CPU utilization

Response time of the PHP application on 1 core of Xeon

SMT2 was better

lower is faster

SMT improved response time even when CPUs were not fully loaded

lower CPU utilization

higher CPU utilization
How SMT affects response time?

<table>
<thead>
<tr>
<th></th>
<th>Low CPU utilization</th>
<th>High CPU utilization</th>
</tr>
</thead>
<tbody>
<tr>
<td>on 1 core</td>
<td>improve</td>
<td>improve</td>
</tr>
<tr>
<td>on multiple cores</td>
<td><strong>degrade</strong></td>
<td>improve</td>
</tr>
</tbody>
</table>

- SMT hurts the response time on multicore systems with low CPU utilization level, which is the common case in today’s server.
- The crossover point depends on the number of cores.
Histogram of response time at low (~25%) CPU utilization

- SMT degraded single-thread performance and shifted the peak of the histogram towards slower response times

- SMT reduced long-latency transactions on 1 core

![Histogram of response time](image)

- on 1 core of Xeon
  - SMT degraded single-thread performance and shifted the peak of the histogram towards slower response times
  - SMT reduced long-latency transactions on 1 core

- on 16 cores of Xeon
  - SMT2 (2-way SMT)
  - SMT1 (disabled SMT)
  - almost no long-latency transactions on 16 cores
  - many long-latency transactions on 1 core
  - SMT reduced the long-latency transactions

Lower means faster response
Breaking down response time

response time $T_r = \text{service time } T_s + \text{waiting time } T_w$

- SMT typically
  - ☹ increases service time (CPU time) by lowering single-thread performance
  - 😊 reduces waiting time (in task scheduling queue) by providing more hardware threads

→ SMT degrades the response time on multicore systems with low CPU utilization level because waiting time is not significant in such case
→ For other cases (single core or high utilization) waiting time affect the total response time
Outline

1. How SMT affects response time
2. Adaptive SMT control with queuing model
Adaptive SMT Control

- We periodically (once per 5 sec)
  - obtain the CPU utilization from /proc/stat,
  - calculate the response time for each SMT level using a new queuing model, and
  - select the best SMT level

- Implemented as a user-space daemon without modification in OS kernel
Challenges in queuing model for SMT processors

- How to model single-thread performance on SMT processor
  - affected by resource contention among the SMT threads

- How to model task migration behavior of the OS task scheduler
  - aggressively balances the load among the SMT threads within one core while minimizing migrations among different cores
Hierarchical queuing model

1. In-core modeling: model the single SMT core
   - To calculate service time (i.e. single-thread performance) and waiting time without considering task migration

2. Out-of-core modeling: model the task migration among cores
   - To modify the waiting time considering the task migration

- Both phases are based on the standard M/M/s model
- Model takes CPU utilization as input w/o task characteristics
- See the paper for the model details
Response time predicted by our model on 16-cores of Xeon

**Measured** response time of MediaWiki (PHP)

**Predicted** response time of MediaWiki (PHP)

- SMT1 was better
- SMT2 was better

**Normalized average response time**

**Normalized CPU utilization (1.0 means fully utilized without SMT)**

**Response time (SMT1)**

**Response time (SMT2)**

lower is faster
Response time predicted by our model on 16-cores of POWER7

**measured** response time of MediaWiki (PHP)

*predicted* response time of MediaWiki (PHP)
Response time predicted by our model on 1-core of Xeon

measured response time of MediaWiki (PHP)

predicted response time of MediaWiki (PHP)

SMT2 was better

lower CPU utilization

higher CPU utilization

lower is faster

higher is faster

lower CPU utilization

higher CPU utilization

lower is faster

higher is faster

normalized average response time

normalized CPU utilization (1.0 means fully utilized without SMT)
Response time with adaptive SMT control on 16 cores of Xeon

SMT1 is automatically selected with low CPU utilization to improve response time.

about 10% reduction in response time compared to the default (SMT2).

SMT2 is automatically selected with high CPU utilization to achieve higher peak throughput.
Response time with adaptive SMT control on 16 cores of POWER7

- SMT1 (disabled SMT)
- SMT2 (2-way SMT)
- SMT4 (4-way SMT)
- with Adaptive Control

SMT1 is selected
SMT2 is selected
SMT4 is selected

about 10% reduction in response time compared to the default (SMT4)

lower is faster
higher CPU utilization

Response time with adaptive SMT control on 16 cores of POWER7
Response time with adaptive SMT control on 1 core of Xeon

SMT2 is automatically selected regardless of the CPU utilization.
Summary

- We showed that SMT may degrade the response time on *multicore* processors with *low CPU utilization*.

- We developed a new queuing model to predict the response time on multicore SMT processors.

- Our adaptive SMT control based on the new model automatically selected the best SMT level at runtime.

See the paper for more detail:
- evaluation with Ruby and Java workloads
- results on moderate number of cores
- detail of the queuing model