Dynamic Fine-Grained Scheduling for Energy-Efficient Main-Memory Queries

We need to make DBMS power-aware
Power management features

- Dynamic voltage and frequency scaling (DVFS)
- Turbo boost
- Idle states (C-states)
- Power-related H/W counters

We can exploit these to improve energy efficiency
Current approaches

• Black box
  – e.g. dynamic concurrency throttling $^{[TPDS13]}$

• Query optimizer $^{[ICDE10]}$

We need fine-grained energy-awareness in the database
Fine-grained energy-aware scheduling

How do you schedule this query plan?

- parameters:
  - parallelism
  - thread placement
  - data placement
  - dynamic voltage and frequency scaling (DVFS)

Calibration of operators under different parameters
Concurrent partitioned scans

- Each thread scans 128MB of integers for 5 secs
- Maximize
  
  \[
  \text{performance per power} = \frac{\text{throughput}}{\text{power}}
  \]
  - under different parallelism, scheduling, and frequency settings
- Machine
  - Two 8-core Intel Xeon E5-2690, HT enabled, 64GB RAM, frequencies from 1.2GHz to 2.9GHz
- Power measurements
  - Hardware performance counters RAPL (CPU & DRAM)
  - External equipment
# Socket-fill scheduling

<table>
<thead>
<tr>
<th>Socket 1</th>
<th>Core 1 &amp; HT</th>
<th>Core 2 &amp; HT</th>
<th>Core 8 &amp; HT</th>
<th>Core 9 &amp; HT</th>
<th>Core 10 &amp; HT</th>
<th>Core 16 &amp; HT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>8</td>
<td>17</td>
<td>18</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>10</td>
<td>16</td>
<td>25</td>
<td>26</td>
<td>32</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Socket 2</th>
<th>Core 1 &amp; HT</th>
<th>Core 2 &amp; HT</th>
<th>Core 8 &amp; HT</th>
<th>Core 9 &amp; HT</th>
<th>Core 10 &amp; HT</th>
<th>Core 16 &amp; HT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**bandwidth saturation**

![Graph showing throughput per Watt vs. number of threads](image)

- **Throughput per Watt**
- **# Threads**

- **Auto (RAPL)**
## Socket-fill scheduling

### Throughput per Watt

<table>
<thead>
<tr>
<th>Socket 1</th>
<th>Socket 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core 1 &amp; HT</td>
<td>Core 9 &amp; HT</td>
</tr>
<tr>
<td>Core 2 &amp; HT</td>
<td>Core 10 &amp; HT</td>
</tr>
<tr>
<td>Core 8 &amp; HT</td>
<td>Core 16 &amp; HT</td>
</tr>
<tr>
<td>1 &amp; HT 9</td>
<td>17 &amp; HT 25</td>
</tr>
<tr>
<td>2 &amp; HT 10</td>
<td>18 &amp; HT 26</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>8 &amp; HT 16</td>
<td>24 &amp; HT 32</td>
</tr>
</tbody>
</table>

### Graph

- **Auto (RAVL):** Triangles up
- **Auto (external equipment):** Line

**Constant difference**

![Graph showing throughput per watt vs. # threads](image)
# Socket-fill scheduling

**Throughput per Watt**

<table>
<thead>
<tr>
<th># Threads</th>
<th>Core 1 &amp; HT</th>
<th>Core 2 &amp; HT</th>
<th>Core 8 &amp; HT</th>
<th>Core 9 &amp; HT</th>
<th>Core 10 &amp; HT</th>
<th>Core 16 &amp; HT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>9</td>
<td>...</td>
<td>17</td>
<td>25</td>
<td>...</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>10</td>
<td>8</td>
<td>18</td>
<td>26</td>
<td>24</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>12</td>
<td>16</td>
<td>25</td>
<td>26</td>
<td>32</td>
</tr>
</tbody>
</table>

**Best frequency**

**Different saturation points**

- Core 1 & HT
- Core 2 & HT
- Core 8 & HT
- Core 9 & HT
- Core 10 & HT
- Core 16 & HT

**Thread Frequencies**

- 1.2GHz
- 2.0GHz
- 2.9GHz
- Auto

<table>
<thead>
<tr>
<th># Threads</th>
<th>0</th>
<th>4</th>
<th>8</th>
<th>12</th>
<th>16</th>
<th>20</th>
<th>24</th>
<th>28</th>
<th>32</th>
</tr>
</thead>
<tbody>
<tr>
<td>Throughput per Watt</td>
<td>0.0</td>
<td>0.5</td>
<td>1.0</td>
<td>1.5</td>
<td>2.0</td>
<td>2.5</td>
<td>3.0</td>
<td>3.5</td>
<td>4.0</td>
</tr>
</tbody>
</table>

1.2GHz
2.0GHz
2.9GHz
Auto
Socket-fill HT scheduling

<table>
<thead>
<tr>
<th></th>
<th>Socket 1</th>
<th></th>
<th></th>
<th>Socket 2</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Core 1 &amp; HT</td>
<td>Core 2 &amp; HT</td>
<td>Core 8 &amp; HT</td>
<td>Core 9 &amp; HT</td>
<td>Core 10 &amp; HT</td>
</tr>
<tr>
<td>Core 1 &amp; HT</td>
<td>1</td>
<td>2</td>
<td>...</td>
<td>Core 16 &amp; HT</td>
<td>17</td>
</tr>
<tr>
<td>Core 2 &amp; HT</td>
<td>3</td>
<td>4</td>
<td>15</td>
<td>Core 10 &amp; HT</td>
<td>19</td>
</tr>
<tr>
<td>Core 8 &amp; HT</td>
<td>...</td>
<td>16</td>
<td></td>
<td>...</td>
<td>18</td>
</tr>
</tbody>
</table>

HT draws negligible power

Throughput per Watt vs. # Threads

- 1.2GHz
- 2.0GHz
- 2.9GHz
- Auto
# Socket-wise scheduling

<table>
<thead>
<tr>
<th>Socket 1</th>
<th>Socket 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core 1 &amp; HT</td>
<td>Core 9 &amp; HT</td>
</tr>
<tr>
<td>Core 2 &amp; HT</td>
<td>Core 10 &amp; HT</td>
</tr>
<tr>
<td>Core 8 &amp; HT</td>
<td>Core 16 &amp; HT</td>
</tr>
<tr>
<td>1 17</td>
<td>2 18</td>
</tr>
<tr>
<td>3 19</td>
<td>4 20</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>15 31</td>
<td>16 32</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th># Threads</th>
<th>Socket 1 Throughput per Watt</th>
<th>Socket 2 Throughput per Watt</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.0</td>
<td>0.5</td>
</tr>
<tr>
<td>4</td>
<td>1.2GHz</td>
<td>2.0GHz</td>
</tr>
<tr>
<td>8</td>
<td>2.9GHz</td>
<td>Auto</td>
</tr>
<tr>
<td>12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

avoids socket-specific bandwidth saturation
Socket-wise HT scheduling

<table>
<thead>
<tr>
<th>Socket 1</th>
<th>Socket 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core 1 &amp; HT</td>
<td>Core 2 &amp; HT</td>
</tr>
<tr>
<td>1 2</td>
<td>5 6</td>
</tr>
</tbody>
</table>

Throughput per Watt vs. # Threads

- 1.2GHz
- 2.0GHz
- 2.9GHz
- Auto

best energy efficiency

1.3x increase
Parallel aggregation

- $a = \sum (b(i) + c(i))$, 4GB arrays

- Minimize
  
  $\text{energy delay product (EDP)} = \text{response time (sec)} \times \text{energy}(J)$
  
  - under different parallelism, scheduling, and memory placement

- Machine
  
  - Two 8-core Intel Xeon E5-2640, HT disabled, 256GB of RAM

- Memory placement
  
  - On first socket
  
  - Interleaved
Parallel aggregation

Memory on first socket

EDP (kJ x sec)

- Socket-fill
- Socket-wise

bandwidth constrained

Memory interleaved

EDP (kJ x sec)

- Socket-fill
- Socket-wise

socket-wise better
Main-memory memory-bound operations

• Intermediate frequency has best efficiency
  – Different saturation points

• Avoid memory bandwidth saturation
  – by data and thread placement

• Up to 4x energy efficiency
Fine-grained energy awareness

Calibration analysis
- of operators and parameters

Measurements
- hardware counters and/or external equipment

Runtime decisions
- scheduling, resource allocation, power management

Energy efficiency

Power

# Threads

Time

parallelism
data & thread placement

DVFS

THIS PAPER

THIS PAPER

Thank you!
References

• [J. R. Hamilton] Internet-Scale Datacenter Economics: Where the Costs And Opportunities Lie. HPTS, 2011.
