NUMA Internals of SQL Server 2012

Gavin Payne
Session objectives

- Intermediate level internals overview
- Define what Non-Uniform Memory Architectures are
- Discover how NUMA is used by SQL Server
- Monitoring NUMA memory activity within SQL Server
Acknowledgements

- Guy Bowerman
  Senior Program Manager for SQLOS

- Jimmy May
  Principle Architect, Microsoft IT
  *Ex-SQLCAT*

- Thomas Kejser
  CTO Fusion-IO
  *Ex-SQLCAT*
Memory architecture overview
Defining NUMA
What is NUMA?

Non-Uniform Memory Architecture is a motherboard design

Improves scalability by removing memory controller bottlenecks

Physical memory becomes either **local** or **remote** to a specific physical processor

Accessing **remote** memory is slower than **local** memory
What NUMA isn’t

- **SMP**
  Symmetric Multi Processing

- All CPUs have **direct access** to all memory

- All CPUs **share** a Front Side Bus
What is NUMA?

- NUMA
- Each node (or CPU) has its own memory bank
- Memory access management is now distributed
Local vs. remote memory

Local Access
- Minimum latency (100-200 CPU cycles)

Remote Access
- Increased latency (1-4x local access)
Measuring inter-node latency

**SQLCat Lab 128 logical CPU server**

**HP DL980G7**

<table>
<thead>
<tr>
<th>NODE</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.0</td>
<td>1.5</td>
<td>2.3</td>
<td>2.7</td>
<td>2.7</td>
<td>2.7</td>
<td>2.7</td>
<td>2.7</td>
</tr>
<tr>
<td>1</td>
<td>1.5</td>
<td>1.0</td>
<td>2.7</td>
<td>2.7</td>
<td>3.3</td>
<td>3.3</td>
<td>3.1</td>
<td>1.3</td>
</tr>
<tr>
<td>2</td>
<td>2.3</td>
<td>2.3</td>
<td>1.0</td>
<td>2.2</td>
<td>3.9</td>
<td>3.9</td>
<td>3.8</td>
<td>2.7</td>
</tr>
<tr>
<td>3</td>
<td>2.7</td>
<td>2.7</td>
<td>1.5</td>
<td>1.0</td>
<td>3.3</td>
<td>3.3</td>
<td>3.1</td>
<td>2.7</td>
</tr>
<tr>
<td>4</td>
<td>3.6</td>
<td>3.6</td>
<td>2.7</td>
<td>2.7</td>
<td>1.0</td>
<td>2.5</td>
<td>3.3</td>
<td>2.4</td>
</tr>
<tr>
<td>5</td>
<td>3.3</td>
<td>3.3</td>
<td>3.3</td>
<td>3.3</td>
<td>1.5</td>
<td>1.0</td>
<td>2.6</td>
<td>2.3</td>
</tr>
<tr>
<td>6</td>
<td>4.0</td>
<td>3.9</td>
<td>4.0</td>
<td>3.9</td>
<td>2.3</td>
<td>2.3</td>
<td>1.0</td>
<td>1.5</td>
</tr>
<tr>
<td>7</td>
<td>3.4</td>
<td>3.4</td>
<td>3.3</td>
<td>3.3</td>
<td>2.7</td>
<td>2.7</td>
<td>1.5</td>
<td>1.0</td>
</tr>
</tbody>
</table>
# SQL Server and NUMA

## SQL Server’s memory usage:
- Heavy user of memory
- Requires tight integration with Windows
- Now also requires an awareness of the server’s hardware

## Product optimisations:
- Limited NUMA support in SQL Server 2000 SP4
- Central to SQLOS’s memory support in SQL Server 2005 *(NUMA by default)*
- What about SQL Server 2008 and SQL Server 2012?
SQL Server
NUMA internals
SQL Server’s internal nodes

NUMA NODE 0

NUMA NODE 1

SERVER HARDWARE

SQL SERVER SOFTWARE

SCHEDULER NODE 0

MEMORY NODE 0

SCHEDULER NODE 1

MEMORY NODE 1

MEMORY NODE 0

MEMORY NODE 1

MEMORY NODE 0

MEMORY NODE 1
### NUMA Nodes within SQLOS

#### sys.dm_os_schedulers

<table>
<thead>
<tr>
<th>scheduler_id</th>
<th>cpu_id</th>
<th>parent_node_id</th>
<th>status</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>VISIBLE ONLINE</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>VISIBLE ONLINE</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>VISIBLE ONLINE</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>1</td>
<td>VISIBLE ONLINE</td>
</tr>
<tr>
<td>1048576</td>
<td>0</td>
<td>64</td>
<td>VISIBLE ONLINE (DAC)</td>
</tr>
</tbody>
</table>

#### sys.dm_os_nodes

<table>
<thead>
<tr>
<th>node_id</th>
<th>dec-to-bin mask</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>00000000001111111</td>
</tr>
<tr>
<td>1</td>
<td>11111111000000000</td>
</tr>
</tbody>
</table>

Node configuration: node 1: CPU mask: 0xffffffff00000000. Active CPU mask: 0xffffffff00000000. This message provides a description of the NUMA configuration for this computer.

Node configuration: node 0: CPU mask: 0xffffffff00000000. Active CPU mask: 0xffffffff00000000. This message provides a description of the NUMA configuration for this computer.
SQL Server’s NUMA objectives

- **Align** the motherboard and SQLOS architectures
- **Reduce** remote memory access
- **Create** memory objects in a task’s NUMA node
- E.g. Buffer pool pages, locks and schedulers etc
Max Server Memory and NUMA

Max Server Memory = 12000MB

SQL SERVER

- Node Target 3000MB
- SQL SERVER MEMORY NODE 0
- SQL SERVER MEMORY NODE 1
- SQL SERVER MEMORY NODE 2
- SQL SERVER MEMORY NODE 3

HARDWARE

- 4000MB NODE 0
- 4000MB NODE 1
- 6000MB NODE 2
- 2000MB NODE 3
SQL Server objects and NUMA

What gets stored in which memory node?

\[ \text{node_id \ in \ sys.dm_os_memory_clerks} \]
Query execution

Where do buffer pool pages come from?

Session A on Node 1

- select * from tblTest
- insert into tblTest

Session B on Node 2

- select * from tblTest
- insert into tblTest

NUMA Internals of SQL Server 2012
Query execution

- What happens with parallel queries?

- Objective is to run all threads in the same node
- Reducing remote memory access is less important

- Controlling task, or data page nodes, not considered
- Often a different node to what you expect is used

- Parallel queries do not respect TCP/IP port boundaries
Monitoring NUMA activity

**sys.dm_os_memory_nodes**

<table>
<thead>
<tr>
<th>memory_node_id</th>
<th>virtual_address_space_committed_kb</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>958336</td>
</tr>
<tr>
<td>1</td>
<td>86852</td>
</tr>
<tr>
<td>64 (DAC)</td>
<td>20</td>
</tr>
</tbody>
</table>

*What about other DMVs?*

**Perfmon**

<table>
<thead>
<tr>
<th>Color</th>
<th>Scale</th>
<th>Counter</th>
<th>Instance</th>
<th>Parent</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00001</td>
<td></td>
<td>Database pages</td>
<td>000</td>
<td>---</td>
<td>MSSQL$INST1:Buffer Node</td>
</tr>
<tr>
<td>0.00001</td>
<td></td>
<td>Database pages</td>
<td>001</td>
<td>---</td>
<td>MSSQL$INST1:Buffer Node</td>
</tr>
<tr>
<td>0.1</td>
<td></td>
<td>Page life expectancy</td>
<td>000</td>
<td>---</td>
<td>MSSQL$INST1:Buffer Node</td>
</tr>
<tr>
<td>0.1</td>
<td></td>
<td>Page life expectancy</td>
<td>001</td>
<td>---</td>
<td>MSSQL$INST1:Buffer Node</td>
</tr>
<tr>
<td>0.000001</td>
<td></td>
<td>Target Node Memory (KB)</td>
<td>000</td>
<td>---</td>
<td>MSSQL$INST1:Memory Node</td>
</tr>
<tr>
<td>0.000001</td>
<td></td>
<td>Target Node Memory (KB)</td>
<td>001</td>
<td>---</td>
<td>MSSQL$INST1:Memory Node</td>
</tr>
<tr>
<td>0.000001</td>
<td></td>
<td>Total Node Memory (KB)</td>
<td>000</td>
<td>---</td>
<td>MSSQL$INST1:Memory Node</td>
</tr>
<tr>
<td>0.000001</td>
<td></td>
<td>Total Node Memory (KB)</td>
<td>001</td>
<td>---</td>
<td>MSSQL$INST1:Memory Node</td>
</tr>
</tbody>
</table>
Systems tasks

- Each hardware NUMA node has its own:
  - IO Completion Port
  - Schedule Monitor
  - Resource Monitor
  - Lazywriter, controlled by global Checkpoint

- But, there is only ever one Log Manager
Mapping connections to Nodes

- User connections are “bound” to a node on creation
- The node is assigned on a round-robin basis

- TCP/IP ports can be used to direct connections to specific NUMA nodes using SQL Server Configuration Manager
Soft NUMA

Logical partitioning within a hardware NUMA node:

- Unique to SQL Server
- Why?

Partition workloads without NUMA hardware support:

- More TCP/IP to scheduler mappings
- Has no effect on memory locality
- Does not create more system tasks
Conclusion

- NUMA is fundamental to SQLOS’s memory management (2005+)

- You cannot change how it works
- But, you can know how it works

- Consider adding \texttt{node\_id} to your memory DMV queries