Scaling Linux to very high number of processors

Christoph Lameter

The question of how well Linux scales to higher processor counts was brought up on the list for the Kernel Summit. This talk will give an overview of the issues systems with a high number of processors are facing under Linux.

Threads and Nodes

- SMP is becoming common. Even laptops are now dual core.
- Typical servers have an ever increasing number of processors. 16p?
- Multiple memory busses. Memory at various distances (NUMA)
- Some introductory NUMA stuff at http://ftp.kernel.org/pub/linux/kernel/people/christoph/pmig/numamemory.pdf

Multiprocessor Scaling

- Distributing the computing load
- Independent execution context
- Everything common (shared) is bad for scalability (bus, memory, locks)
- But we need a shared bus and memory so that the system can work as a whole.
- Node to node interconnect
- Per cpu structures
- Per node structures

Scaling Limits

- 4-8p with just a single memory bus. Some 16p and 32p solutions.
- Larger system require NUMA interconnect. Scalability mostly determined by interconnect.
- Tight packing decreases latency (Opteron onboard NUMA)
- Longer distances allow larger systems (SGI Altix).
- The bigger the more distant memory

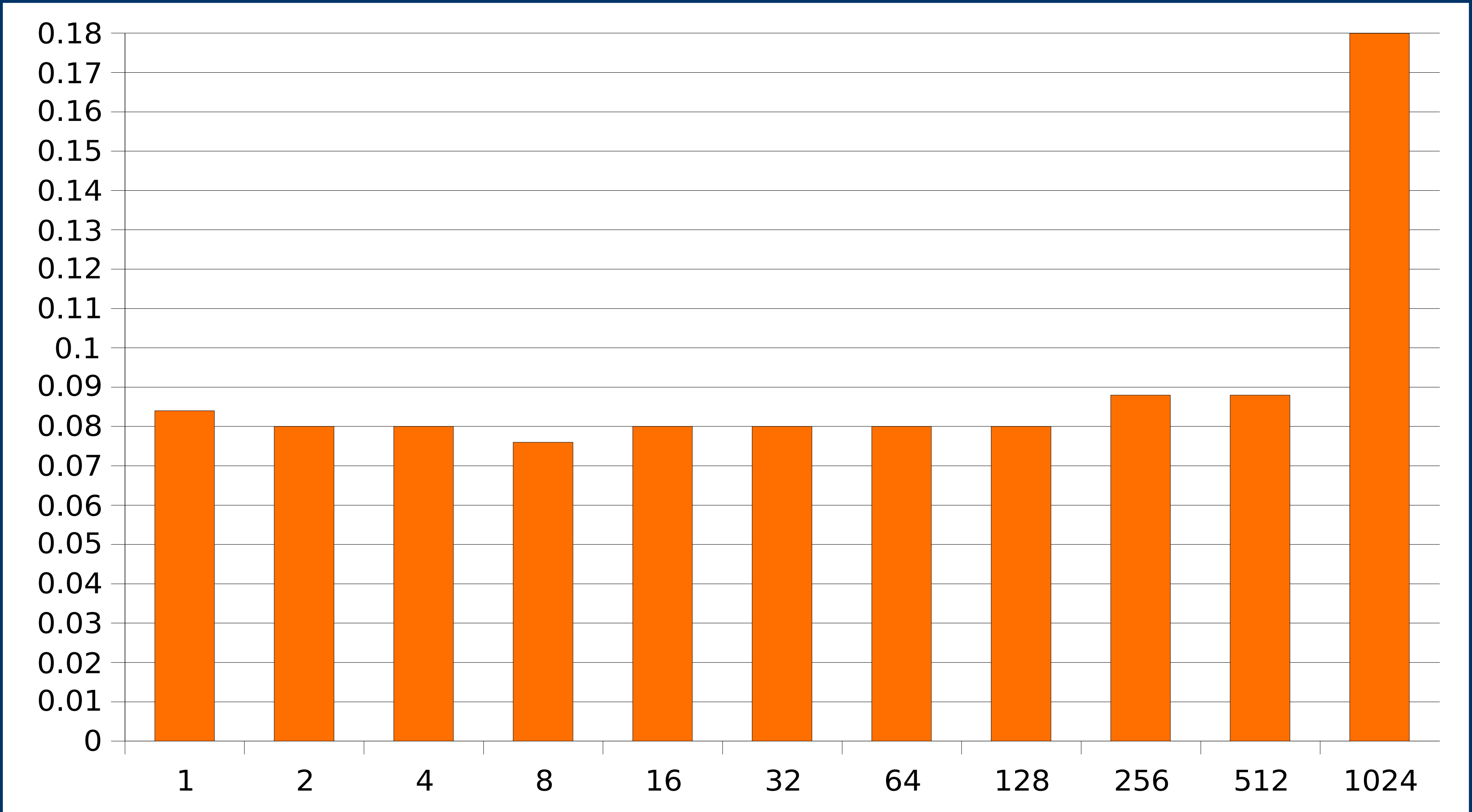
Altix NUMA Scaling

- 32p, 64p, 512p pretty common installations. 2p per node.
- Partitioning. Number of SSIs.
- Newest Itanium product line: 1024 node
- SUSE certified.
- Current design is for 1024 nodes / 4096 processors. Montecito delays result in only 1024 processors so far.

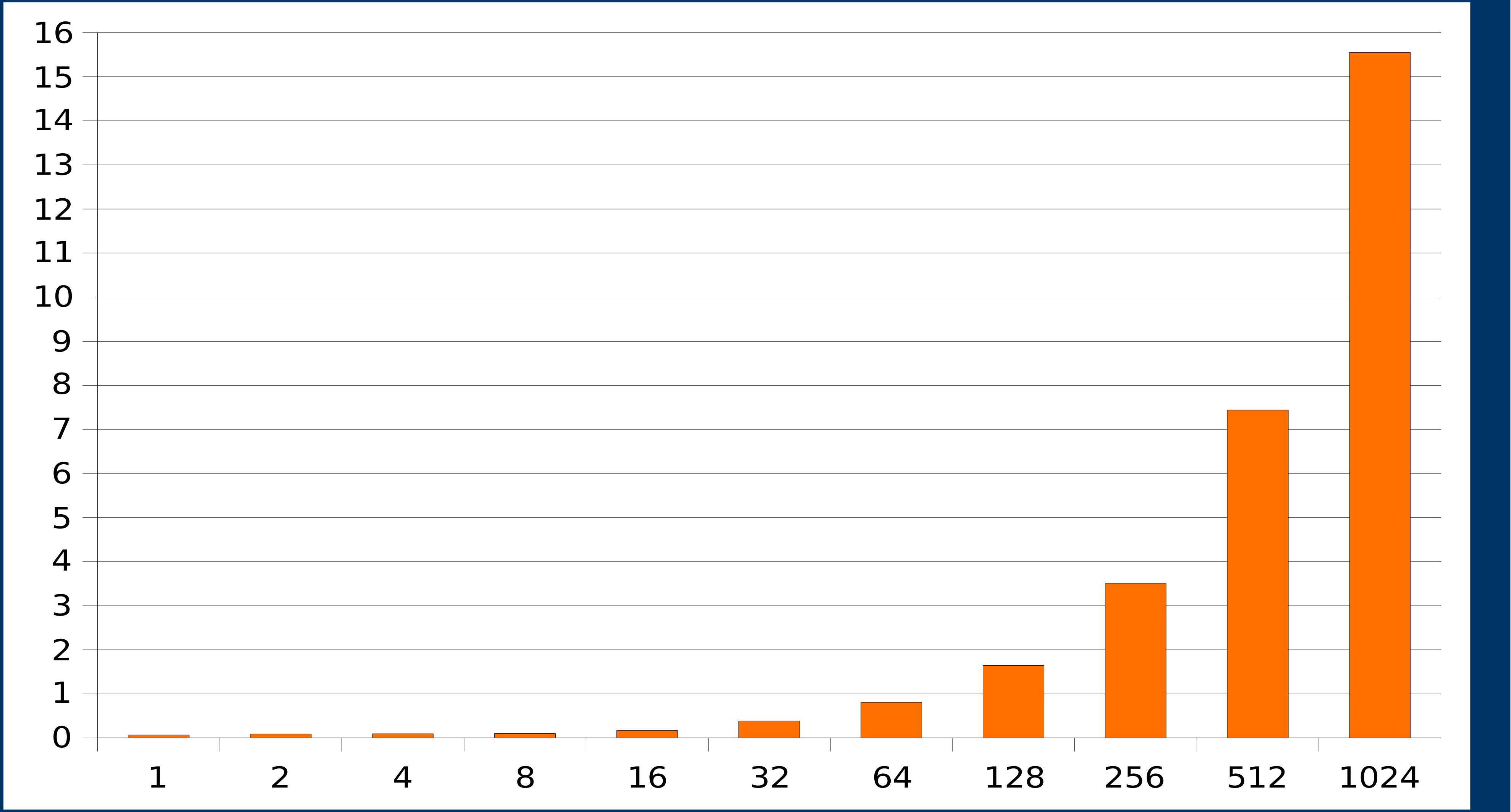
Some performance numbers

- New SGI blade architecture
- 1024 node, one processor per node
- Itanium Madison 9M processors.
- 8 Terabyte of main memory (8 GB per node).
- 6.4Gbyte/sec switched fabric interconnect.
- Bootup 30 minutes to about an hour.
- Time of allocating 100MB in seconds for a given processor count.

Memalloc: Parallel processes



Memalloc: Threaded app



Issues

- Larger system (>64p) may have issues with lock contention for some workloads (radix tree, dcache, inode locks).
- Long boot time: Memory initialization and device initialization.
- Memory balancing, control of memory.
- MTBF is reduced. It would be good to have the ability for a node or memory to fail.

Future Problems

- Sparsely populated per cpu and per node arrays.
- The number of per node/cpu objects in some kernel subsystems grows excessively.
- Need to replicate memory to avoid off node access? (40% for some apps!)
- 4K pagesize for i386 and x86_64 will result in significant TLB pressure for large memory sizes.

Conclusion

- The real challenge at 1024-4096 processors for now
- Hardware issues are significant.
- The OS seems to be mostly okay below 1024p.
- A list of remaining kinks (policies, better scheduling, control over memory etc).
- NUMA scheduler (get too complex, user space?)