RSIM x86: A cost-effective performance simulator

Ricardo Fernández Pascual

José M. García

{r.fernandez,jmgarcia}@ditec.um.es

Departamento de Ingeniería y Tecnología de Computadores

Universidad de Murcia

30071 Murcia (SPAIN)

GACOP

Index

- **Introduction.**
  - Performance simulators for multiprocessors.
  - RSIM.
  - Our problems with RSIM.
- **Portability problems.**
- **Evaluation of the new platform.**
- **Conclusions.**

Introduction. Performance simulators for multiprocessors.

- **We need models to test our ideas.**
- **Analytical models are not enough for predicting performance.**
- **We need experimental models for research in computer architecture ⇨ Simulators.**
- **Simulators are several orders of magnitude cheaper than the simulated system or a prototype.**
  - There is a tradeoff between speed and accurateness.
  - They are also several orders of magnitude slower.

Some available performance simulators.

- RSIM.
- Simics.
- GEMS.
- SimpleScalar.
- ML-RSIM.
- SimOS.
- MASE.
- Liberty.

Introduction. RSIM.

- Execution driven simulator targeted to study cc-NUMA architectures.
- Models a system comprised by several out-of-order processors with aggressive ILP. Enables accurate modeling of contention and synchronization in multiprocessors and of the effect of speculation.
- Detailed model of an aggressive memory system with scalable interconnection.
- Interprets Solaris/SPARC v9 executables.
- Available only in big-endian machines.

Introduction. Our problems with RSIM.

- **Our research group comprises several researchers who need to perform many simulations using RSIM.**
- **RSIM runs on Solaris/SPARC and IRIX/MIPS machines (and similar big-endian machines).**
  - Those machines are too expensive.
  - We have only two UltraSPARCs.
  - Every development and testing needs to be performed using remote access.
  - Long queues to perform simulations.
- **If RSIM ran on x86 machines, we would be able to have more machines available.**

Portability problems. (1)

- **Until now, RSIM has been ported only to a few similar 32 bit big-endian architectures.**
  - Solaris/SPARC.
  - IRIX/MIPS.
  - HP-UX 10/PA-RISC.
- **Portability problems:**
  - Build and library issues due to differences between Solaris and Linux.
  - Byte Ordering issues.
  - System call interface differences.
  - Floating point incompatibilities.

Portability problems. (2)

- **Byte ordering issues:**
  - RSIM simulates a big-endian architecture on a big-endian machine, which is straightforward.
  - Our port requires to simulate a big-endian machine on a little-endian architecture.
    - We need to change the order of bytes in some places.
    - We keep the simulated memory in big-endian and registers in little-endian.
    - Swap bytes when performing each memory access.
    - The predecoded executable data is swapped after loading.
    - Subtly affects other parts of the simulator, like partially overlapped memory accesses or initialization routines.
      - Requires careful debugging and extensive testing.

Portability problems. (3)

- **System call interface differences:**
  - RSIM uses the host OS to simulate system calls performed by the simulated system.
    - Simulated programs are actually statically linked Solaris programs.
  - Both Linux and Solaris are System V derivatives, but with slight differences.
  - In case of running on Solaris, no argument translation is necessary.
  - When running on Linux, the system call interface has some small differences which need to be translated.

Portability problems. (and 4)

- **Floating point differences:**
  - SPARC implements the IEEE 754 floating point standard, while Intel uses (by default) its own 80-bit format internally (it still uses IEEE 754 in memory).
  - This extra precision is harmless, but rounding differences cause small differences in some results.
  - Also, the representation of NaN values is different in each system, causing small differences.
  - These differences are harmless but complicate the verification of the correctness of the port and make comparisons between results obtained with RSIM and RSIM-x86 harder.
  - When we need identical results between RSIM and RSIM-x86, we use SSE2 to perform floating-point operations (uses IEEE 754, but requires Pentium IV processors) and normalize NaN representation.

Evaluation. (1)

- **The ported version (RSIM-x86) produces exactly the same results than the original version (RSIM).**
  - This allows us to meaningfully compare results of new experiments with results from older experiments performed using RSIM.
- **We want to check if using the ported version of RSIM on off-the-shelf x86 machines is a cost-effective solution to perform a great number of simulations.**
  - Better execution speed.
  - Better in terms of number of simulations per hour per euro.

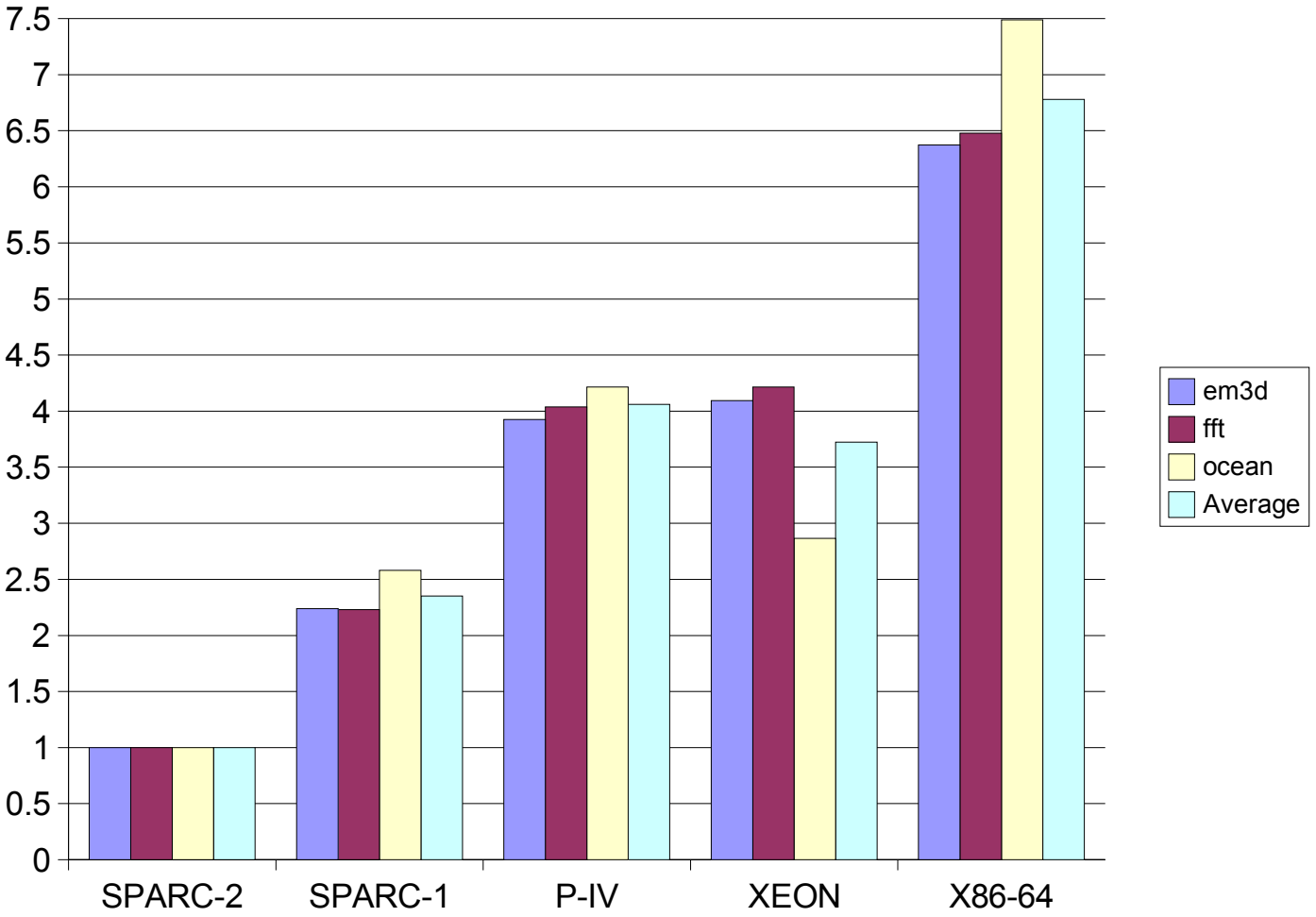
Evaluation. (2)

- **The systems used for the evaluation are:**
  - A high-end Solaris/SPARC Sunblade-2000: SPARC-1.
  - A low-end Solaris/SPARC Sunblade-100: SPARC-2.
  - A high-end Linux/Athlon64 SMP system (running in legacy IA-32 mode): X86-64.
  - A high-end Linux/Xeon SMP system: XEON.
  - A low-end Linux/Pentium-IV system: P-IV.

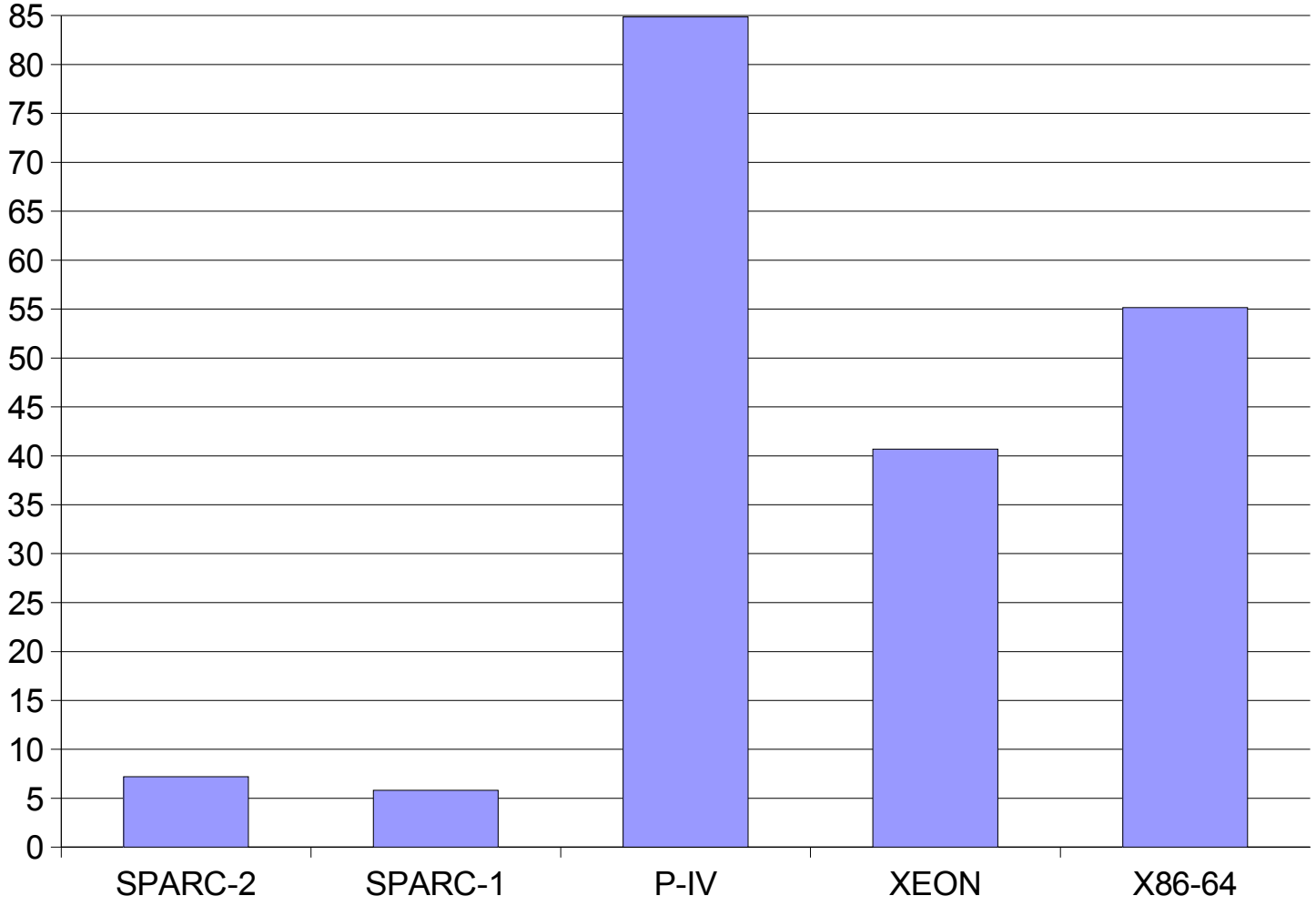
	SPARC-1	SPARC-2	X86-64	XEON	P-IV
Processor	UltraSPARC-III	UltraSPARC-IIi	Opteron	Xeon	Pentium-IV
No. processors	1	1	2	2	1
Frequency	1015 MHz	650 MHz	1971 MHz	2 GHz	3 GHz
RAM Memory	2 GB	256 MB	1 GB	1 GB	1 GB
L2 Cache	8 MB	512 KB	1 MB	512 KB	1 MB
Price	5000 €	1800 €	3000 €	2600 €	600 €

GACOP

Evaluation. (3)



Evaluation. (and 4)



Conclusions.

- Our port of RSIM to x86 allows us to use our research resources more efficiently.
- Using cheap off-the-shelf hardware allows us to provide each researcher with his own workstation to develop and test his experiments before submitting long simulations to our cluster.
- Clusters of inexpensive machines are the most cost-effective solution for our simulations.
- http://www.ditec.um.es/gacop/tools/RSIM-x86