L82

Linux on zSeries performance update

Martin Kammerer

September 19 - 23, 2005 San Francisco, CA

Martin Kammerer <kammerer@de.ibm.com>,
IBM development lab Boeblingen, Germany

Trademarks

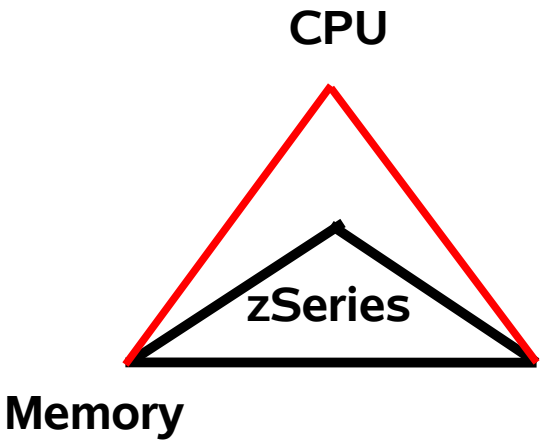
The following are trademarks of the International Business Machines Corporation in the United States and/or other countries.
Enterprise Storage Server
ESCON*
FICON
FICON Express
HiperSockets
IBM*
IBM logo*
IBM eServer
Netfinity*
S/390*
VM/ESA*
WebSphere*
z/VM
zSeries
* Registered trademarks of IBM Corporation
The following are trademarks or registered trademarks of other companies.
Intel is a trademark of the Intel Corporation in the United States and other countries.
Java and all Java-related trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc., in the United States and other countries.
Lotus, Notes, and Domino are trademarks or registered trademarks of Lotus Development Corporation.
Linux is a registered trademark of Linus Torvalds.
Microsoft, Windows and Windows NT are registered trademarks of Microsoft Corporation.
Penguin (Tux) compliments of Larry Ewing.
SET and Secure Electronic Transaction are trademarks owned by SET Secure Electronic Transaction LLC.
UNIX is a registered trademark of The Open Group in the United States and other countries.
All other products may be trademarks or registered trademarks of their respective companies.

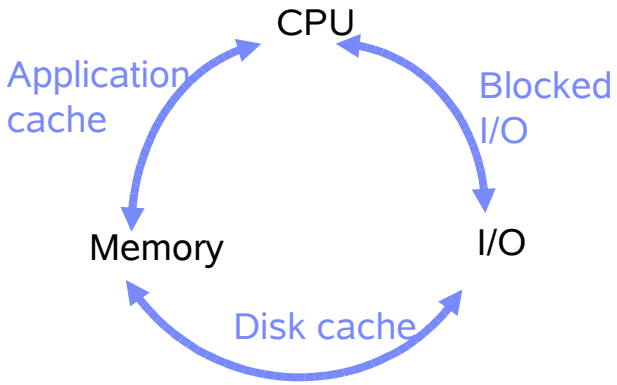
Agenda

- **System Capacity and zSeries hardware**
- **Kernel 2.6 based distros**
  - scalability
  - networking
  - compiler
  - Java
  - NPTL
  - I/O schedulers
  - sequential I/O scalability
  - direct I/O / async I/O
  - fixed I/O buffers

Relative System Capacity

- **A system provides different types of resources**
- **Capacity for each resource type may be different**
- **The ideal machine provides enough capacity of each type**
- **Don't forget additional Resources (Network, Skilled staff, Money, availability of software, reliability, time ...)**

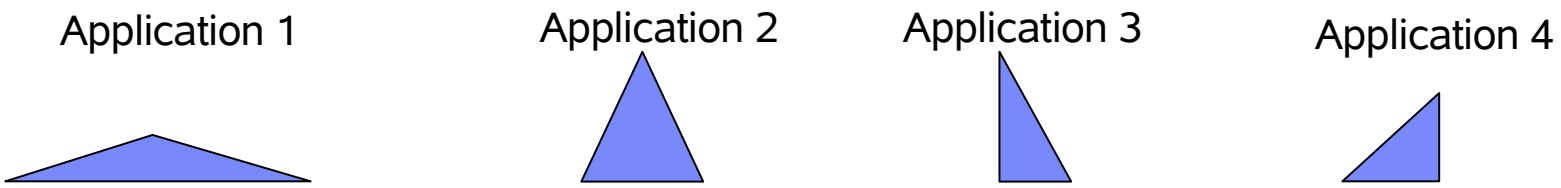




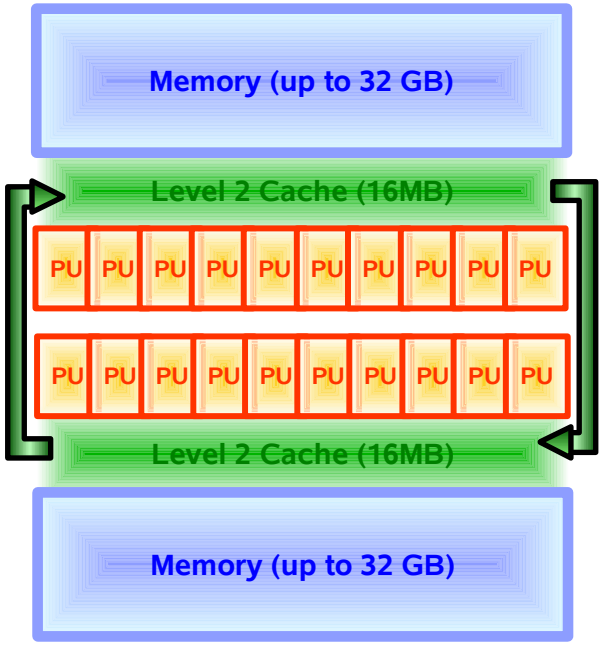
The ideal platform requires a mix of resources in right quantity

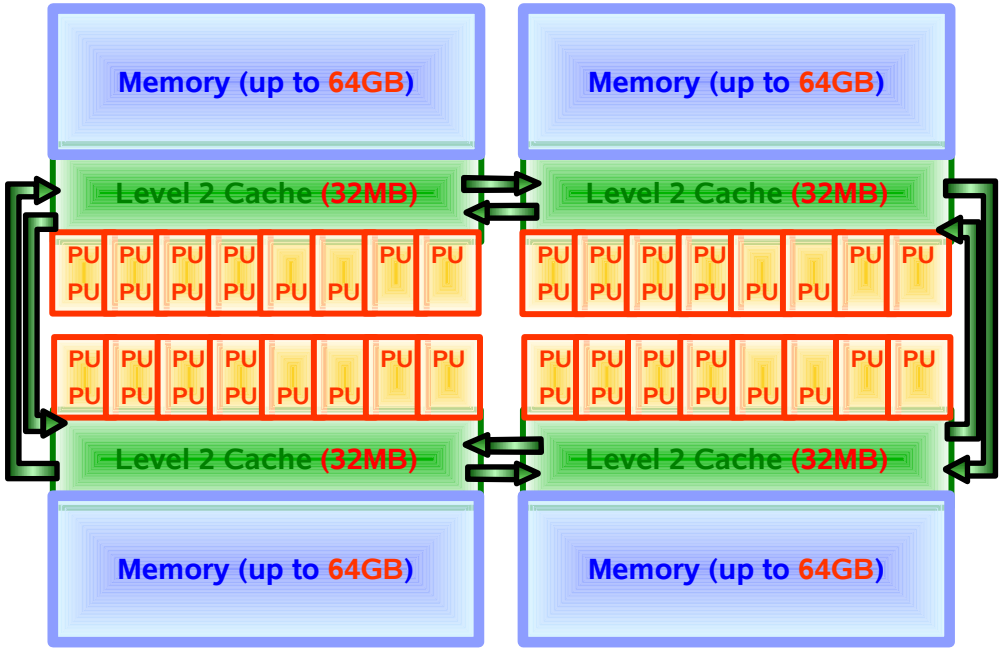
Resource Profiles

- Each application has its specific requirements
  - CPU intensive
  - I/O intensive
  - Memory intensive
- Applications can often be tuned to change the resource profile
  - Exchange one resource for the other
  - Requires knowledge about available resources
- Some platforms can be extended better than others
  - Not every platform runs every application well
  - It's not easy to determine the resource profile of an application

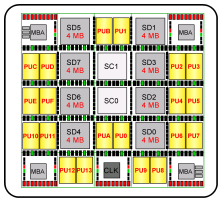


zSeries extended multi book structure



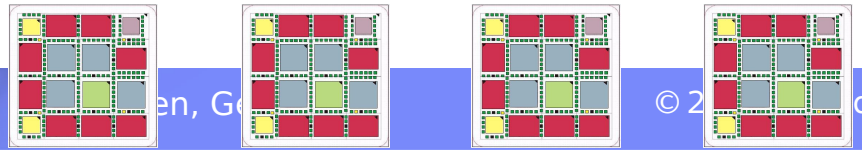


From z900/z800 ...

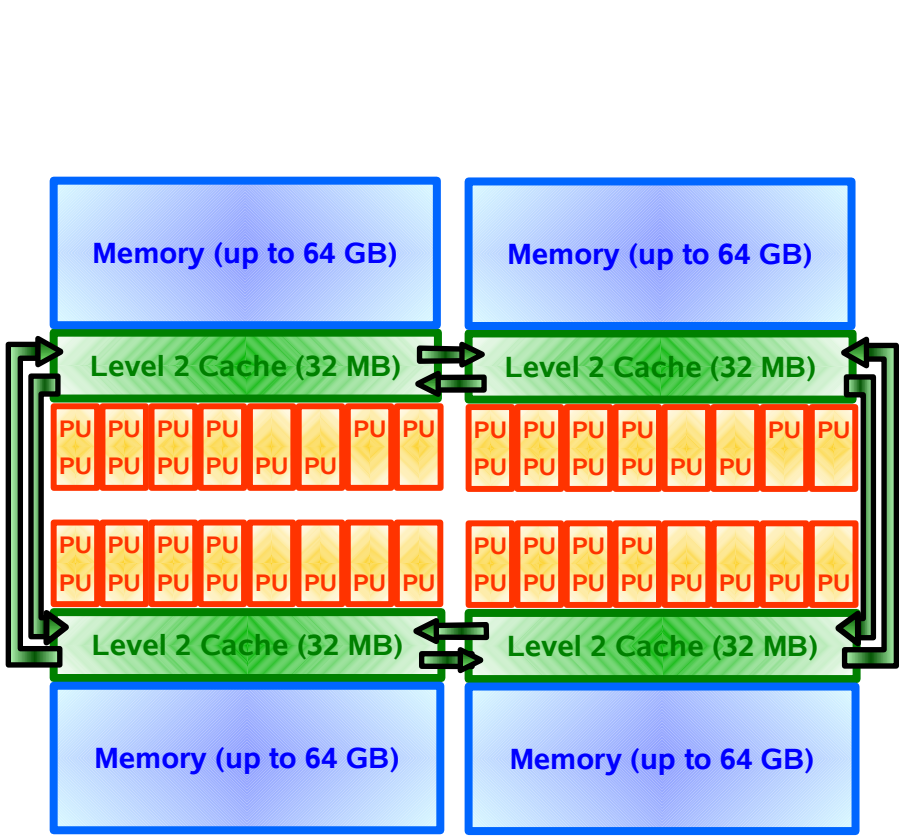


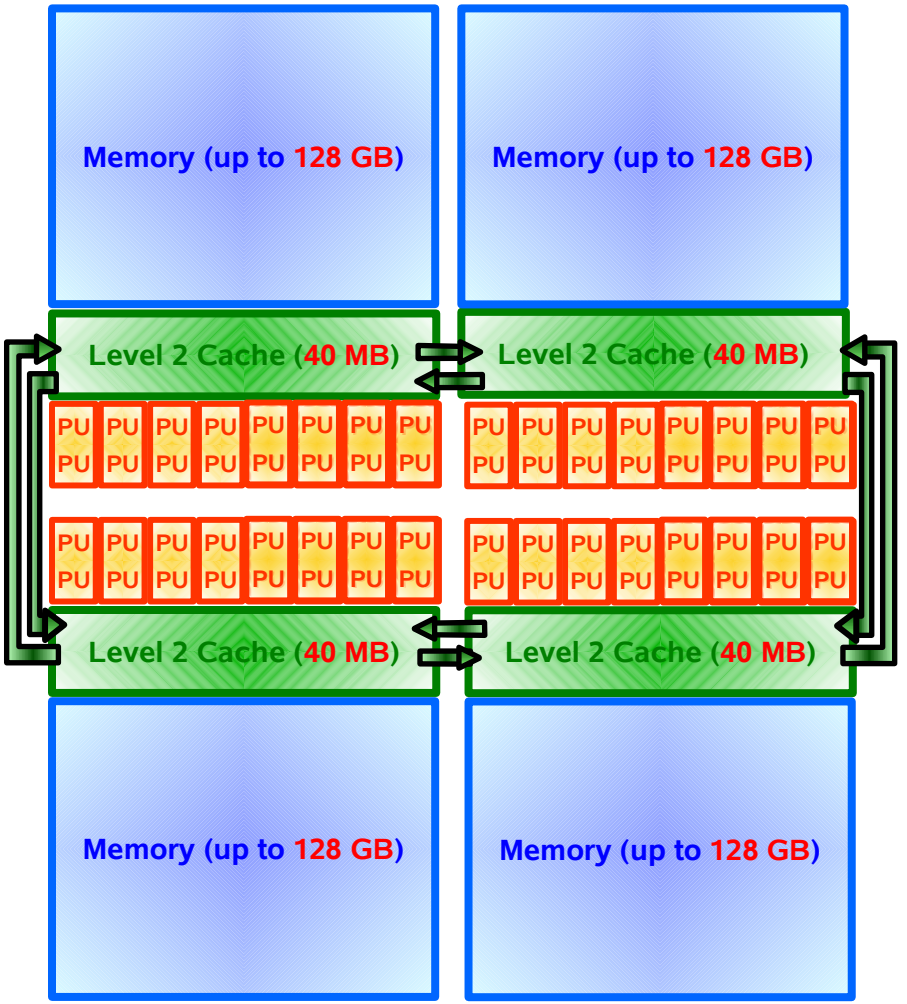
... to modular z990 systems with up to 3-fold capacity

- **0.83 nsec CPU-Cycle (1.2 GHz)**
- **Superscalar design**
- **50 - 60% more UP-Performance z900**

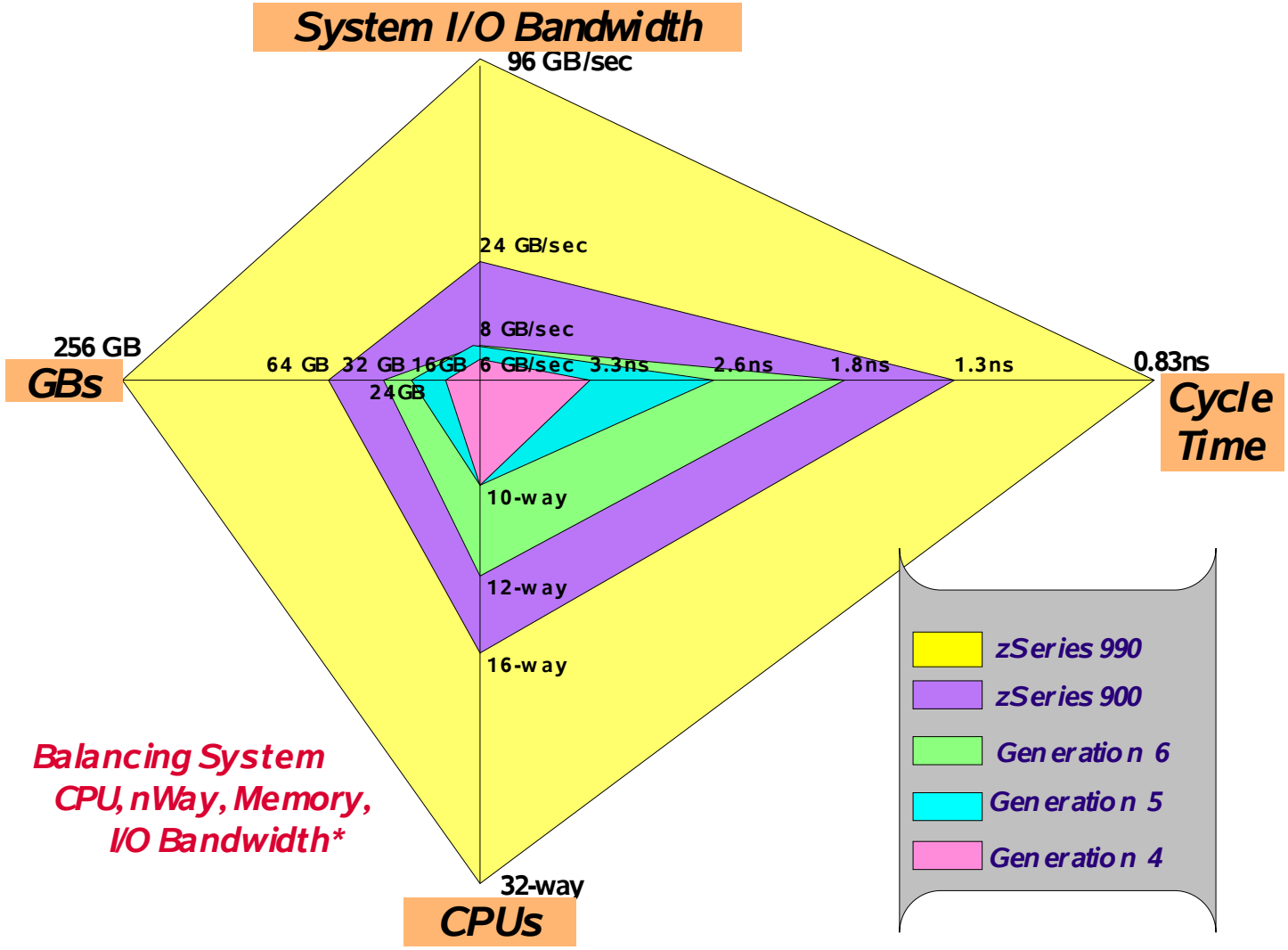


From z990 to System z9





IBM S390 and zSeries Servers - Balanced Scaling



* External I/O or STI bandwidth only (Internal Coupling Channels and HiperSockets not included) zSeries MCM internal bandwidth is 500 GB/s. Memory bandwidth not included (not a system constraint)

Our Hardware for Measurements

2084-B16 (z990)

- 0.83ns (1.2 GHz)
- 2 Books each with 8 CPUs
- 2 * 32 MB L2 Cache
- 96 GB
- FICON Express
- HiperSockets
- OSA Express GbE

2105-F20 (Shark)

- 16 GB Cache
- 384 MB NVS
- 128 * 36 GB disks
- 10.000 RPM
- FCP (1 Gbps)
- FICON (1 Gbps)

2105-800 (Shark)

- 32 GB Cache
- 1 GB NVS
- 128 * 72 GB disks
- 15.000 RPM
- FCP (2 Gbps)
- FICON (2 Gbps)

Linux on zSeries – Kernel 2.6 new features

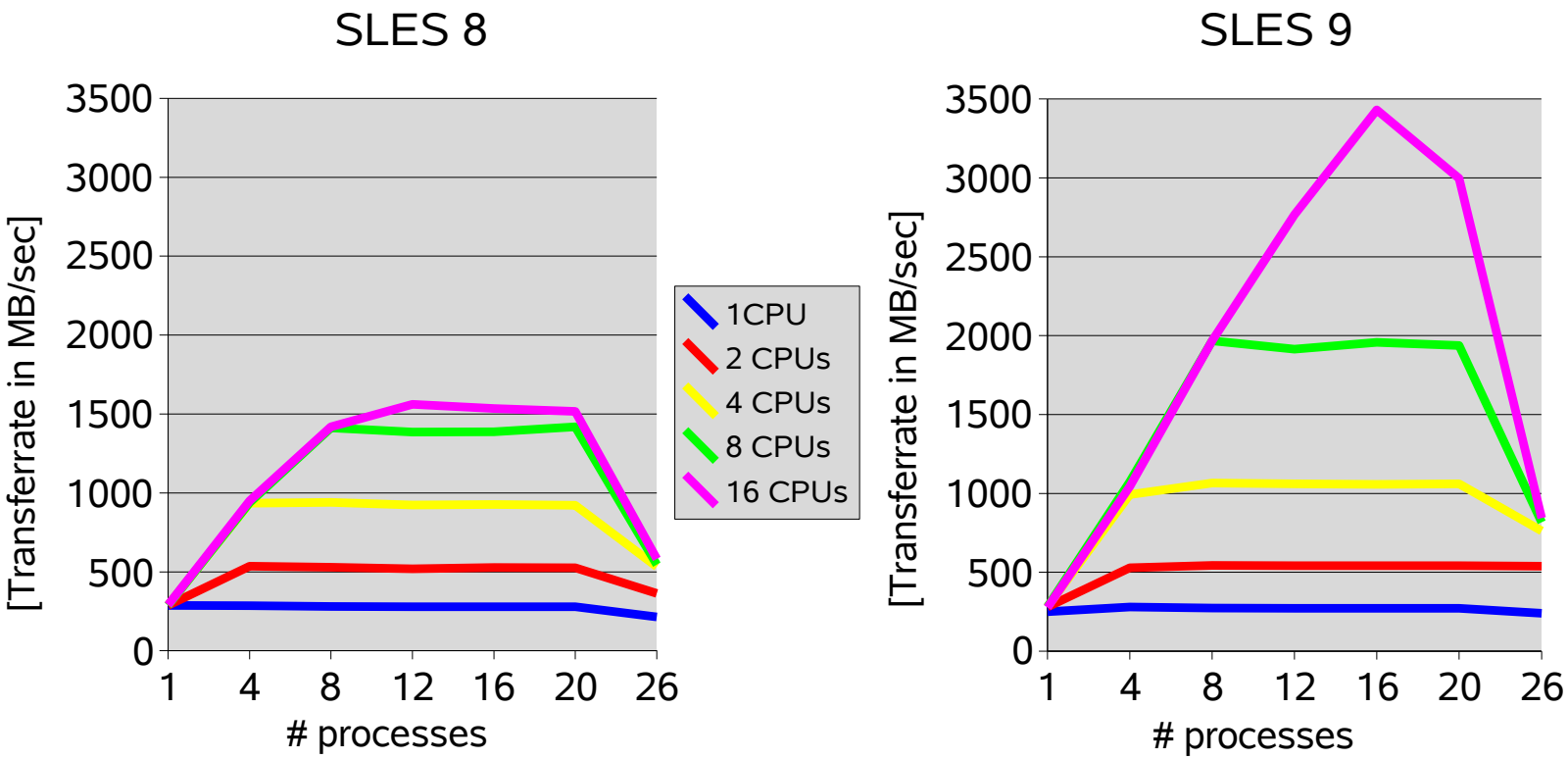
This is only a subset from a long list

	V2.4.23	V2.6
Maximum CPUs (IA32)	16	64
Maximum CPUs (zSeries)	32	64 (hardware limit)
Maximum RAM (IA32)	16GB	64GB
Maximum RAM (zSeries)	256GB (hardware limit)	256GB (hardware limit)
Maximum major devices	255	4095
Maximum minor devices	255	1M
Maximum fs size (IA32)	2TB	16TB
Maximum fs size (zSeries)	2TB	8EB
Max. Process / Threads	64K	2G
Threading Library	Linux Threads	Linux Threads & NPTL
I/O Mode	classic	classic & async I/O
Schedulers	Default schedulers	O(1) process scheduler, different I/O schedulers

Scalability Benchmark - Dbench

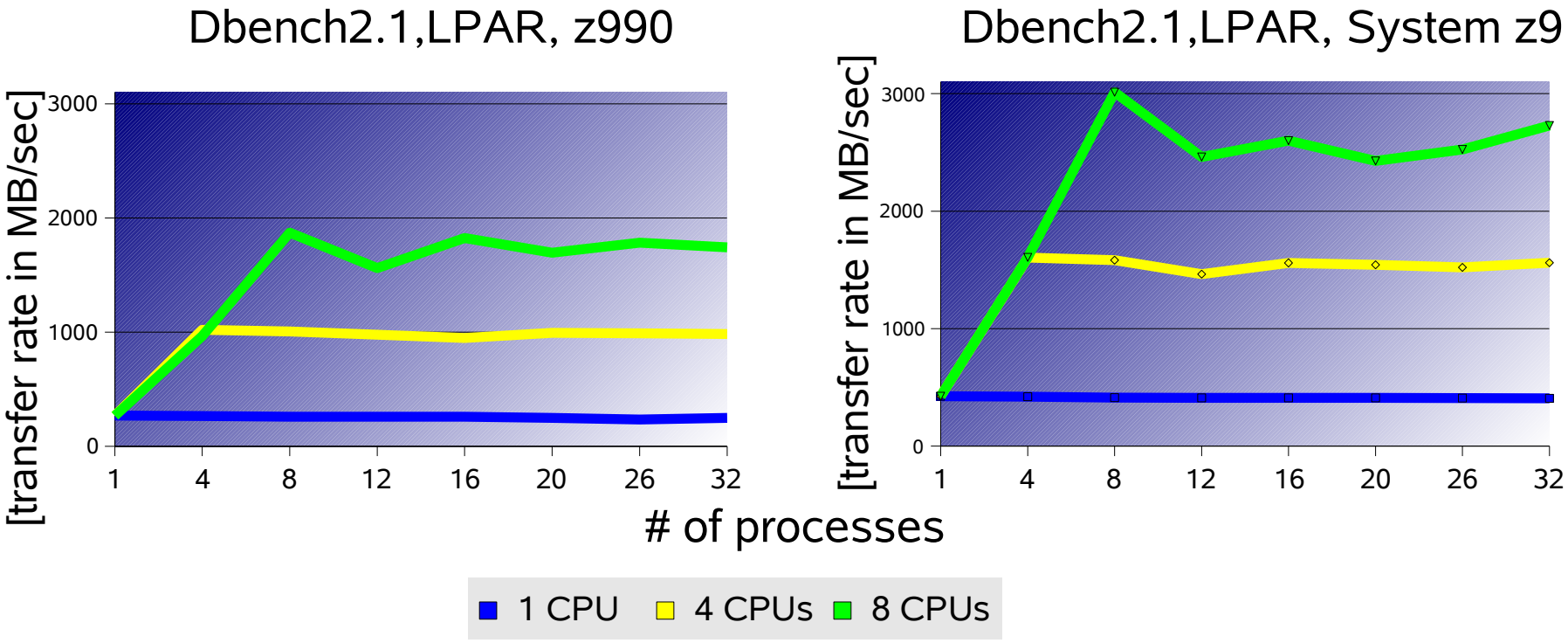
- File system benchmark
- Generates load patterns similar to Netbench
- It does no networking calls
- Does not require a lab of load generators to run
- De-facto standard for generating load on the Linux VFS
- Author: Andrew Tridgell
- Released under the GNU Public License

Scalability – kernel 2.4 vs kernel 2.6



- SLES9 scales better with 8 and 16 CPUs
- Disk I/O starts at 20+ processes
- Dbench V2.1, SLES 8 Submarine, SLES 9 SP1

Scalability – z990 versus System z9

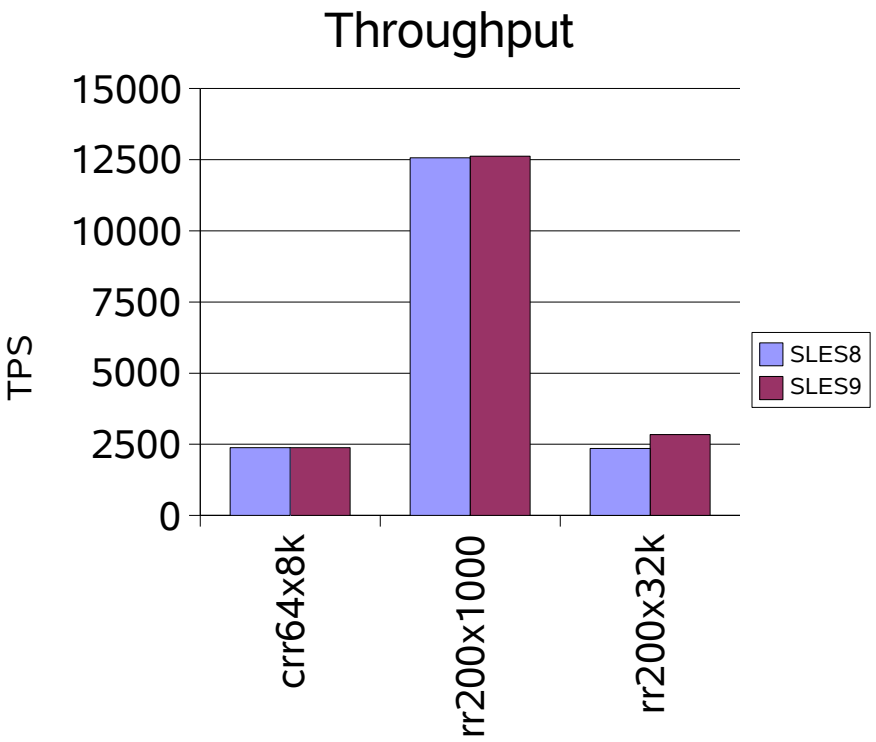


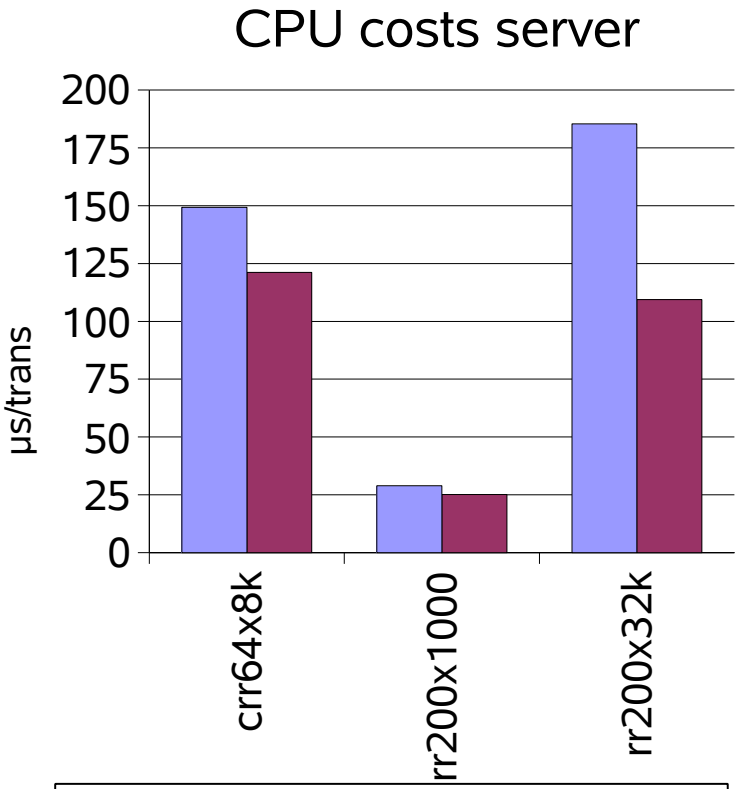
- System z9 takes advantage of higher memory bandwidth
- Throughput increase by 50% for 1, 4 and 8 CPUs
- **IBM internal driver, pre-GA hardware → preliminary results**

Networking Benchmark

- **AWM**
  - several workload models
    - transactional workload
    - streaming workload
    - mixed workload
  - measured with GbE (QDIO, LCS), Hipersockets, and virtual connections in z/VM
  - throughput and cost (CPU) measurements

Networking Gigabit Ethernet, MTU 1500





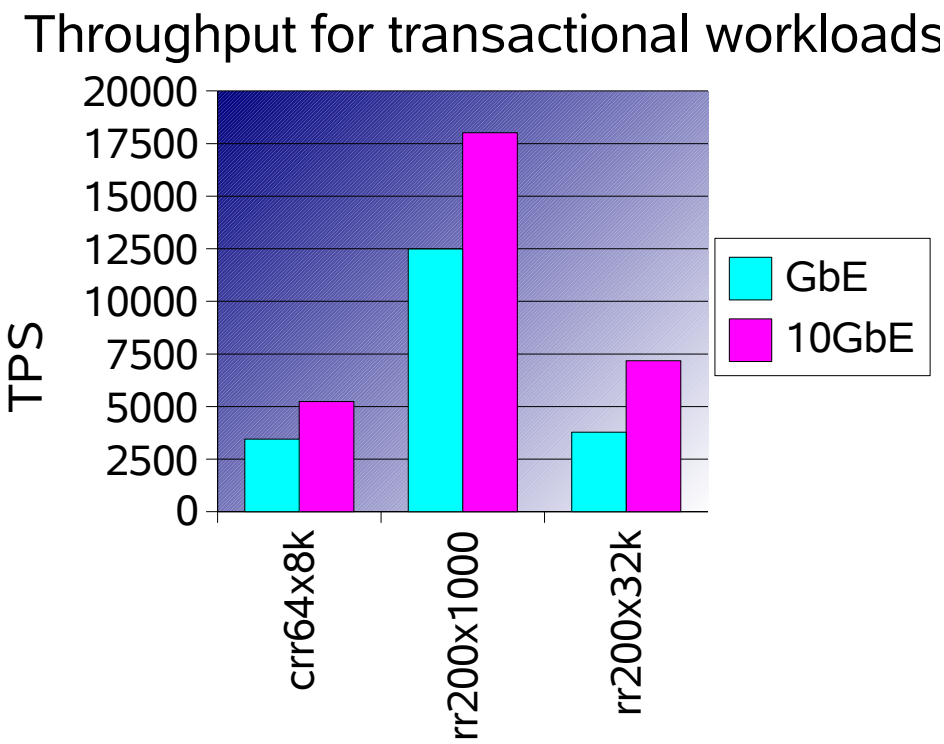
- **rr200x32k improved by 20%**
- **reduced CPU costs**

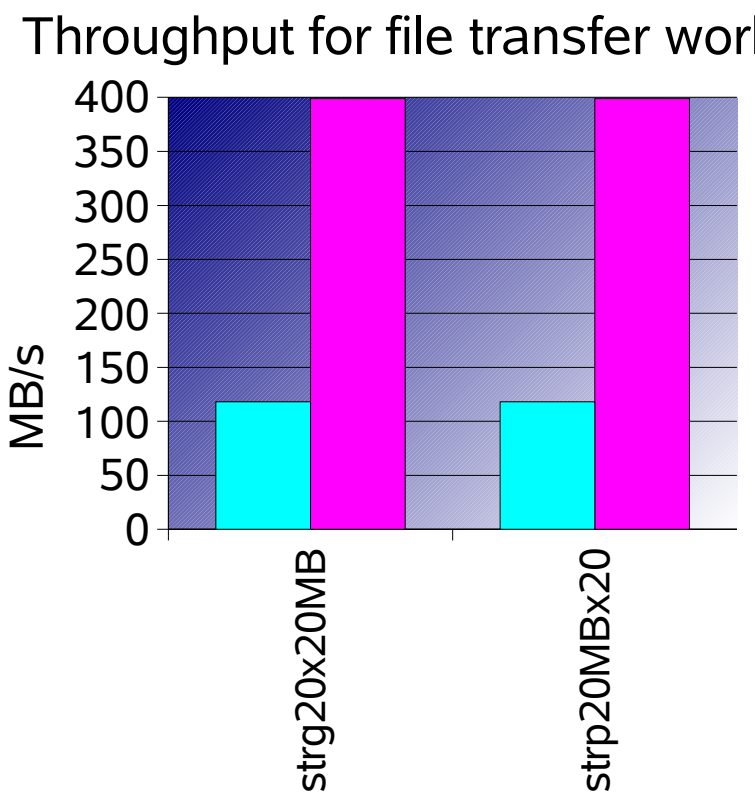
crr64x8k – website request

rr200x1000 – online transaction

rr200x32k – database query

Networking 10 Gigabit Ethernet, MTU 8992



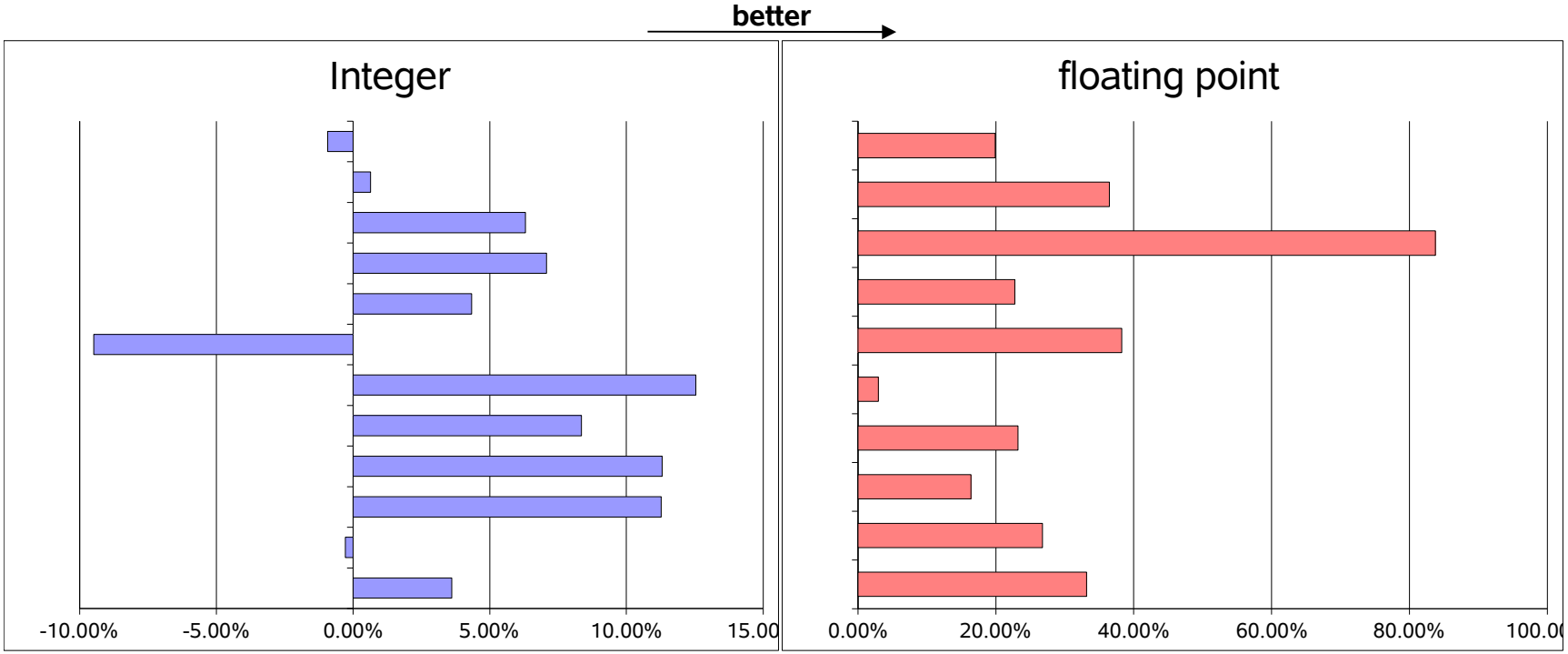


- **rr200x32k improved by 1.9x, str improved by 3.4x**
- **CPU costs equal or less**

The GNU gcc Compiler

- **Compiler supports various architectures**
  - s390 (31-bit) and s390x (64-bit) are integrated in GNU development cycles
- **Recommended compile options**
  - '-O3' to enable many performance optimization options
  - SLES8 and RHEL3 based on gcc-3.2.2
  - Parameter 'march=' and 'mtune=' values <G5,z900,z990>
    - with SLES8 SP3 comes optional experimental gcc-3.3
    - SLES9 includes gcc-3.3
    - RHEL4 AS includes gcc-3.4.3 as default

gcc 64bit compiler

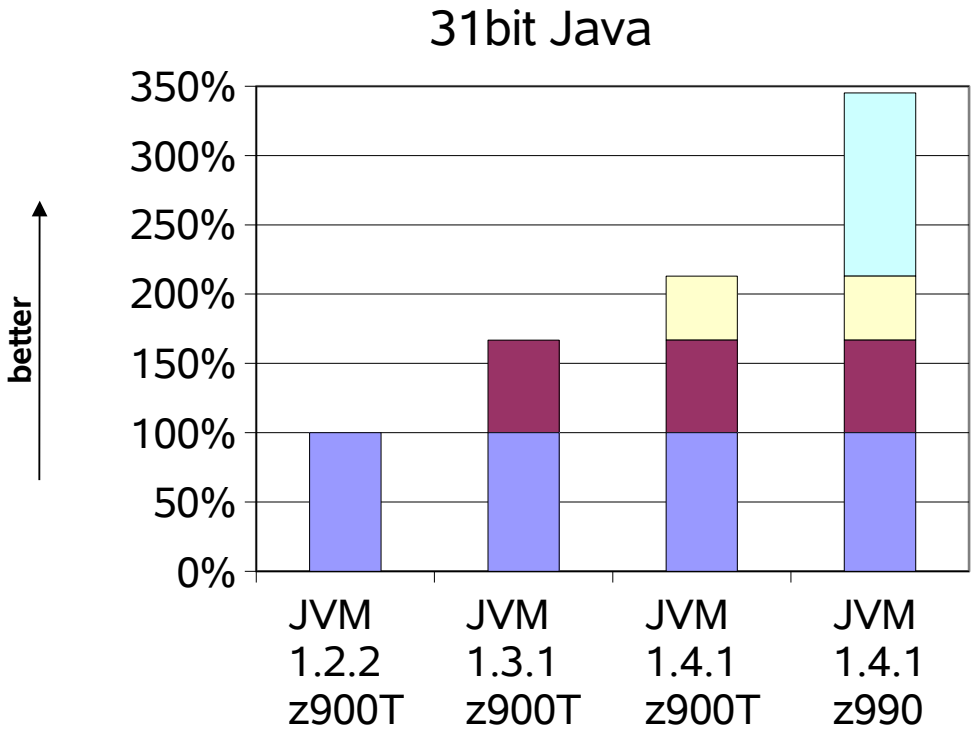


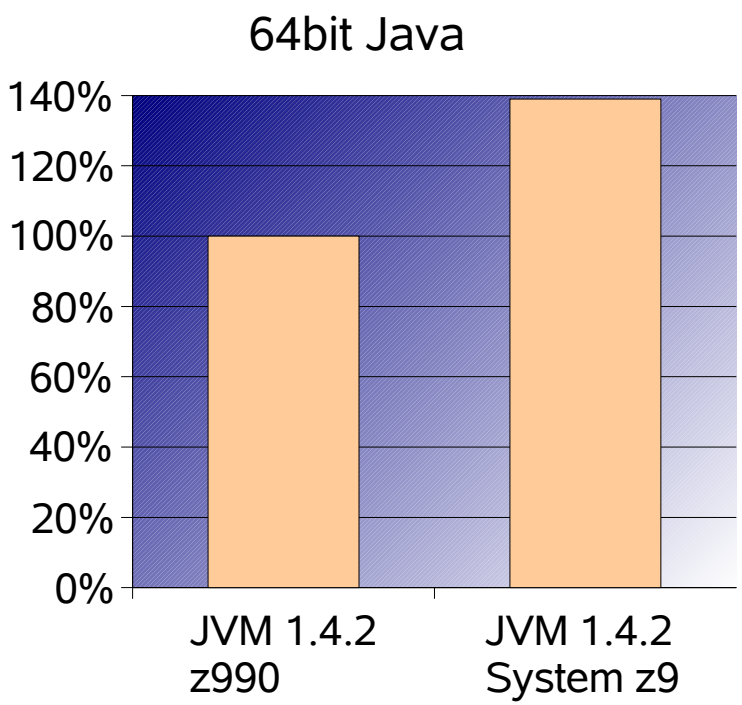
- new compiler SLES9 / RHEL4 is worth a try
- optimize for your architecture e.g. -march=z990

Java

- Java Virtual Machine improved
- zSeries Just in Time Compiler improved
- 2001: JVM 1.2.2, Websphere 3.x
- 2002: JVM 1.3.1, Websphere 4.x, 5.0
- 2003: JVM 1.4.1, Websphere 5.0.x
  - JVM 1.4.1 available in 31-bit | 64-bit
- 2004: JVM 1.4.2, Websphere 5.1, 6.0

Java





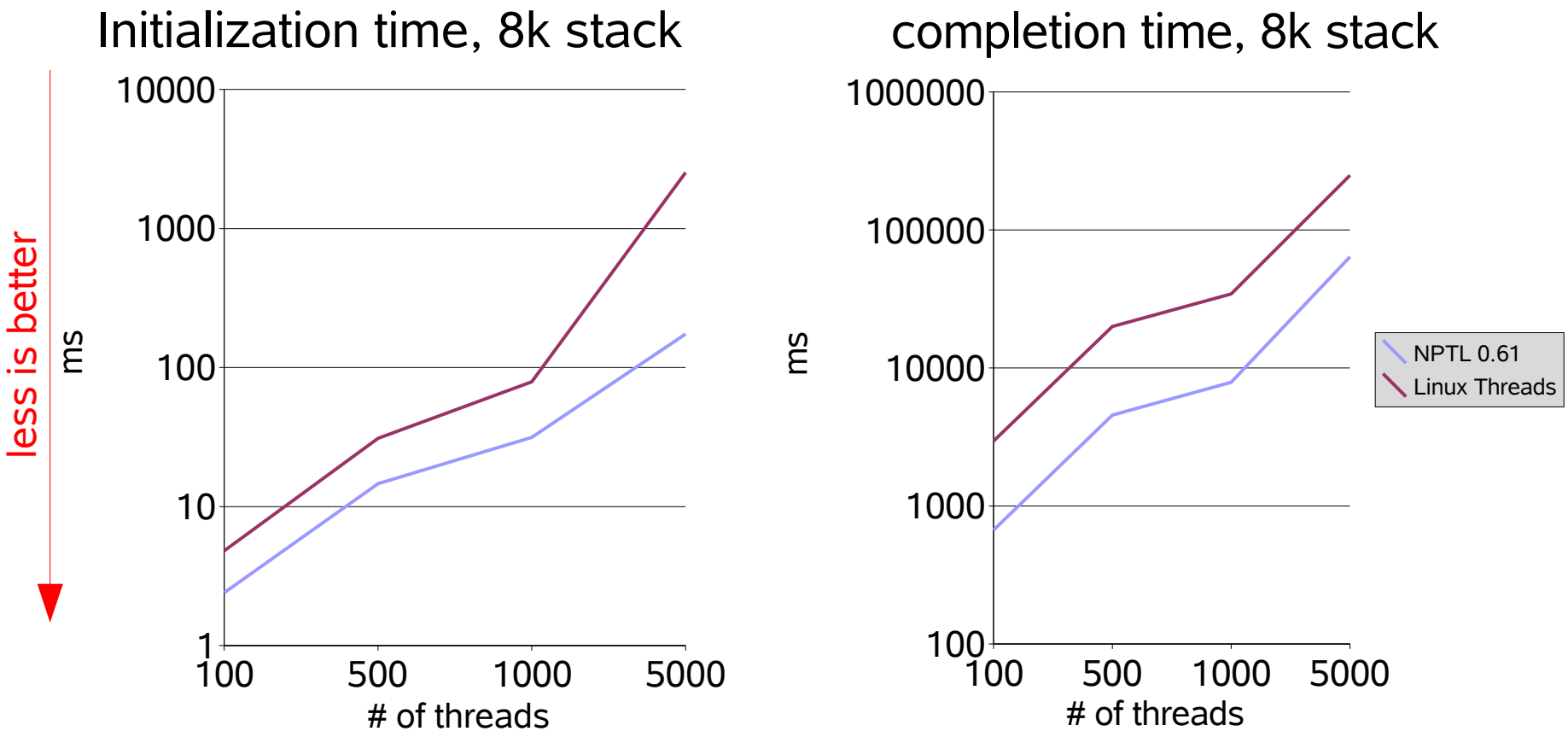
- improvements in HW, Linux, JVM and JIT
- 64 bit Java is now production ready
- System z9: **IBM internal driver, pre-GA hardware → preliminary results**

Linux threading models

- **Linux threads**
  - not fully POSIX compliant
  - per process manager thread to create and coordinate between the threads
  - lack per thread synchronization for inter – thread communication and resource sharing
  - scalability problems
- **New Posix Thread Library**
  - fully POSIX compliant
  - no per process manager but new system calls, ..., TLS
  - high performance threading support
  - exploitation requires minor modifications in most threaded applications
  - NPTL is the strategic direction for Linux threading

- 2.6 based distributions have both
- switch with `export LD_ASSUME_KERNEL=2.4.21`

NPTL results, 8 CPUs



Linux 2.6 I/O Schedulers

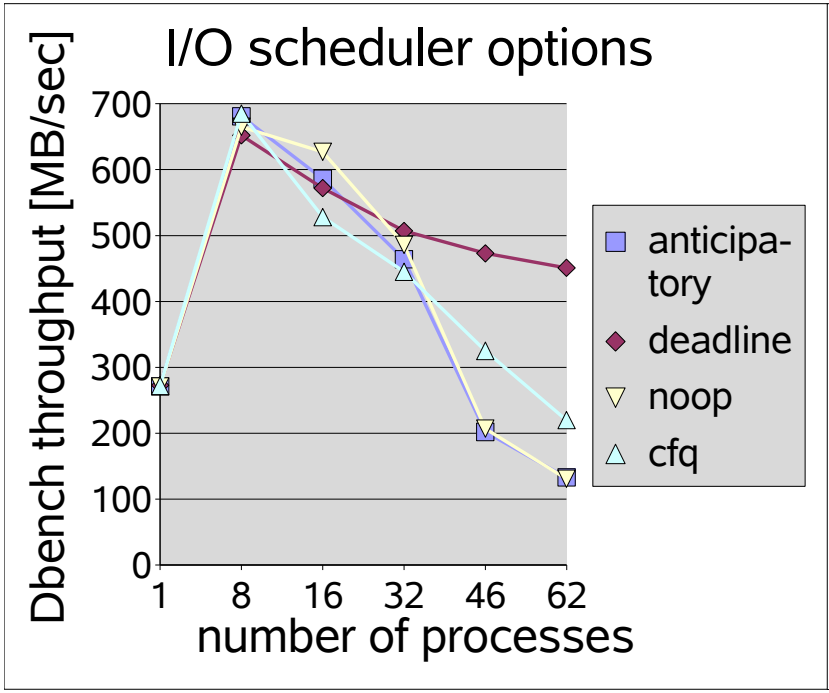
- Four different I/O scheduler are now available
  - noop scheduler
    - only request merging
  - deadline scheduler
    - avoids request starvation
  - anticipatory scheduler (as scheduler)
    - designed for the usage with physical disks, not intended for storage subsystems
  - complete fair queuing scheduler (cfq scheduler)
    - all users of a particular drive would be able to execute about the same number of I/O requests over a given time.

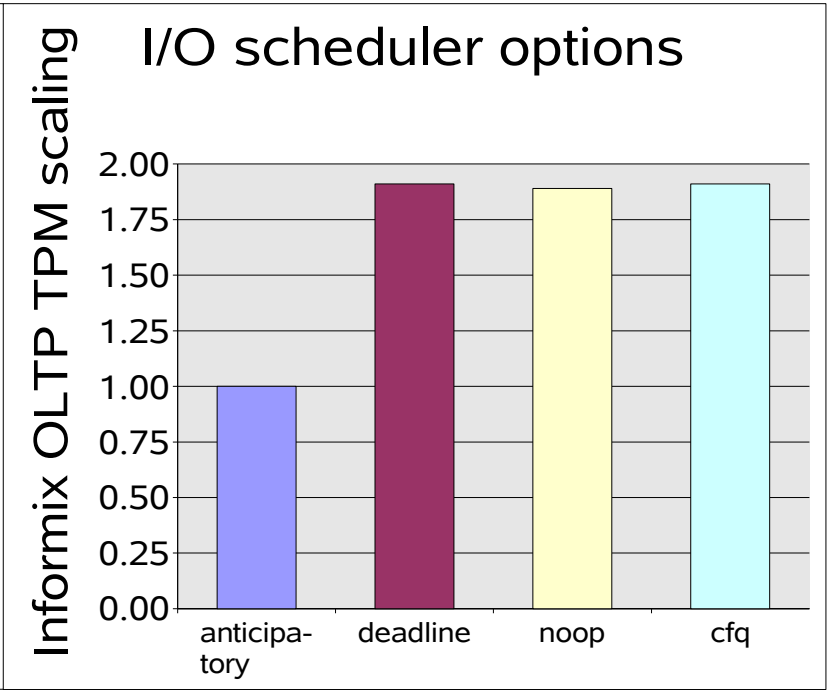
Linux 2.6 I/O Scheduler

- **Defaults**
  - Kernel 2.6 anticipatory scheduler
  - SUSE SLES 9 (s390, s390x), RHEL4 (s390, s390x): cfq scheduler
- **How to identify which I/O scheduler is used**
  - **Red Hat RHEL4: cat /var/log/dmesg | grep scheduler**
  - **SuSE SLES9: cat /var/log/boot.msg | grep scheduler**

    -> Using cfq io scheduler
- **How to select the scheduler**
  Set boot parameter elevator in zipl.conf, e.g.
  - **[ipl2GB8CPUdeadl]**
    target = /boot/zipl
    image = /boot/image
    ramdisk = /boot/initrd
    parameters = "maxcpus=8 dasd=5849 root=/dev/dasda1 elevator=deadline"
    possible values: as | deadline | cfq | noop

I/O scheduler





- Test characteristics: random disk I/O, many processes
- Significant difference between best and worst case

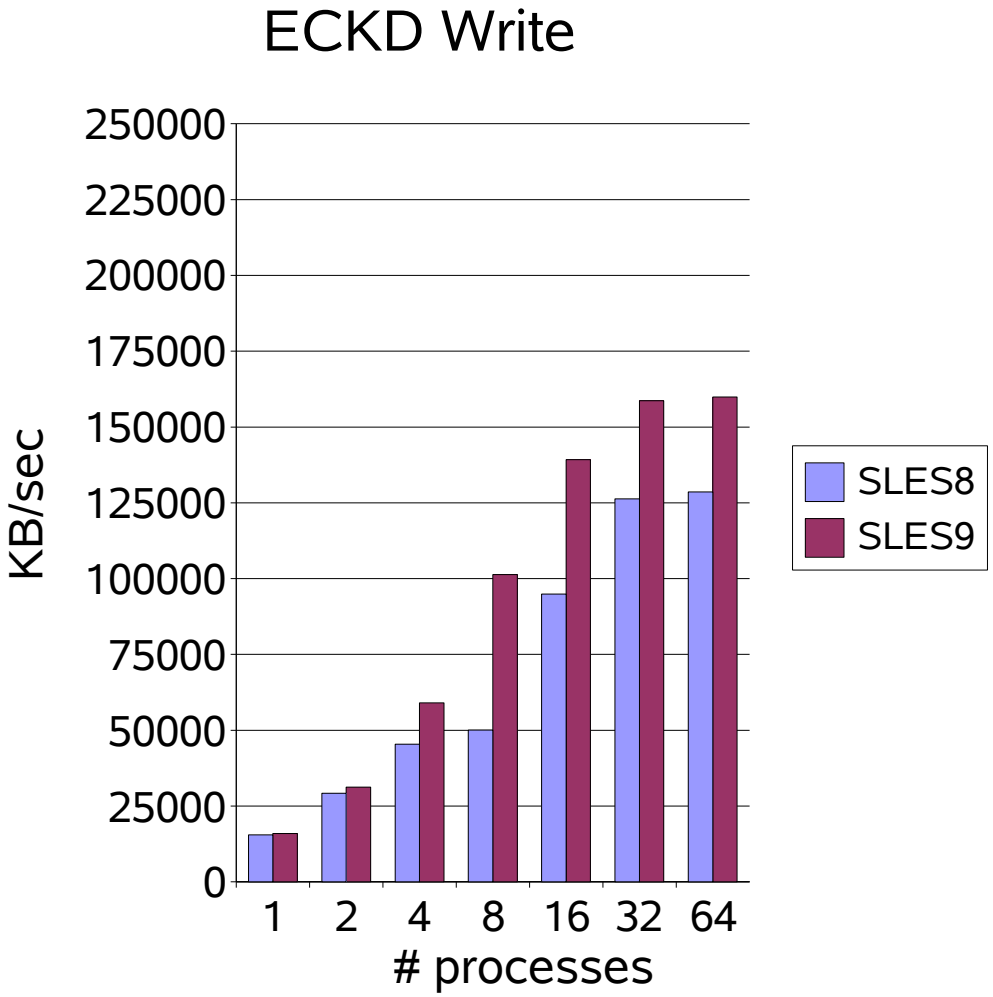
Random I/O - Summary

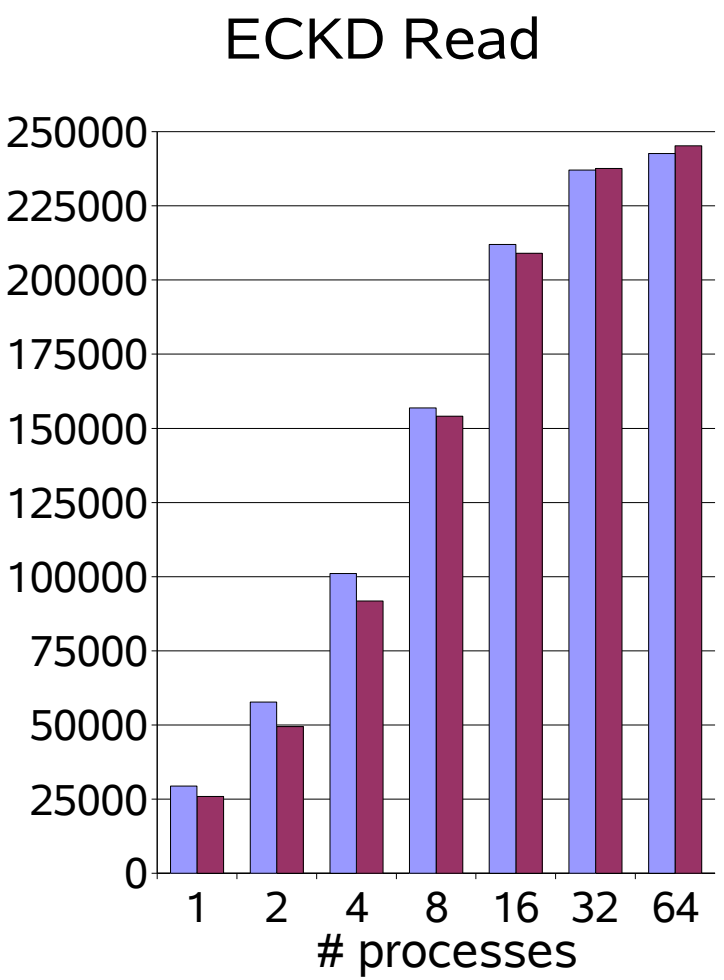
- Choice of the I/O scheduler is workload dependent
  - Deadline option performs best in our experiments with Dbench and Informix OLTP
  - Anticipatory I/O scheduler is not recommended for zSeries
- Sorting of requests (elevator) is not be an advantage on storage subsystems
- I/O scheduler influence not seen for sequential I/O, but experiments are ongoing

I/O Sequential Benchmark

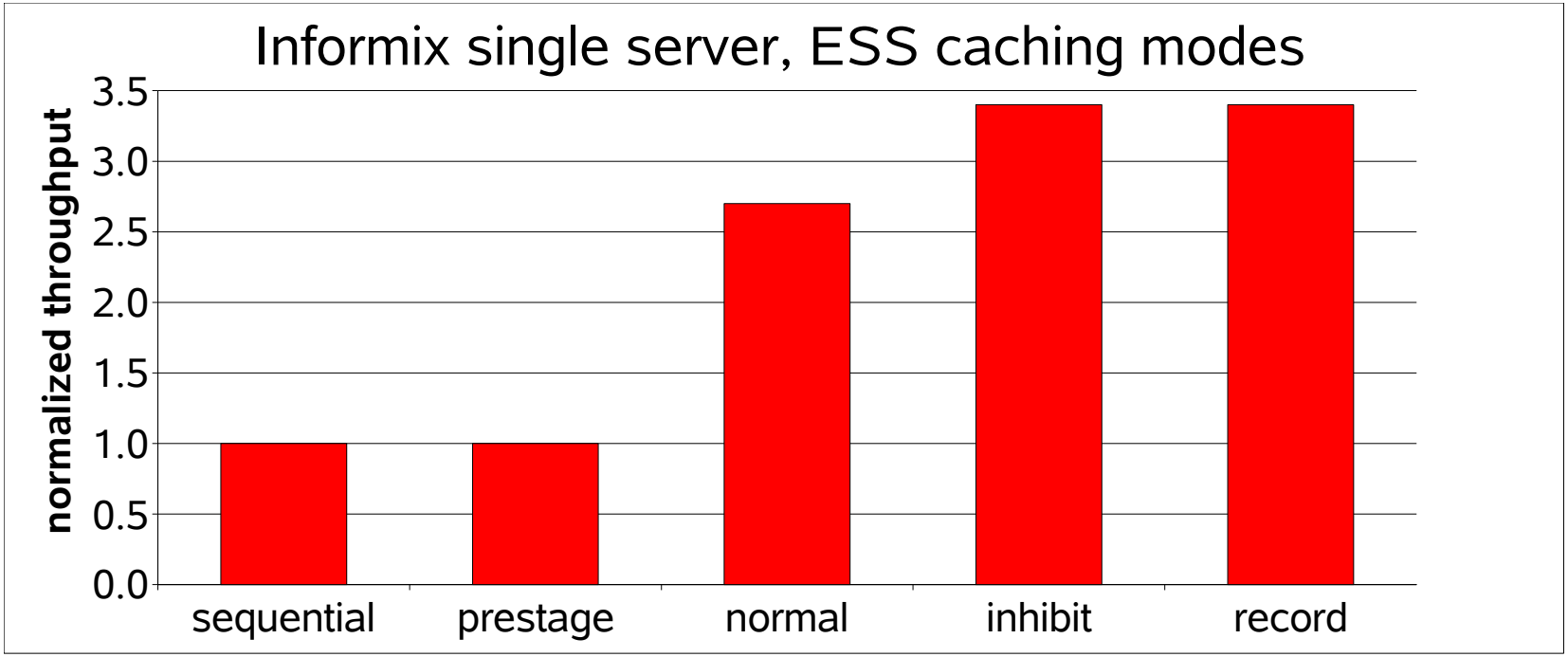
- **Iozone**
  - Threaded file system benchmark used to measure synchronous I/O
  - write, rewrite, read of a 700MB file
  - 1,2,4,8,16,32,64 threads write on the same number of disks
  - Used on FICON and SCSI disks
  - Main memory was restricted to 256MB

Kernel 2.6 Sequential I/O





ESS Caching Modes

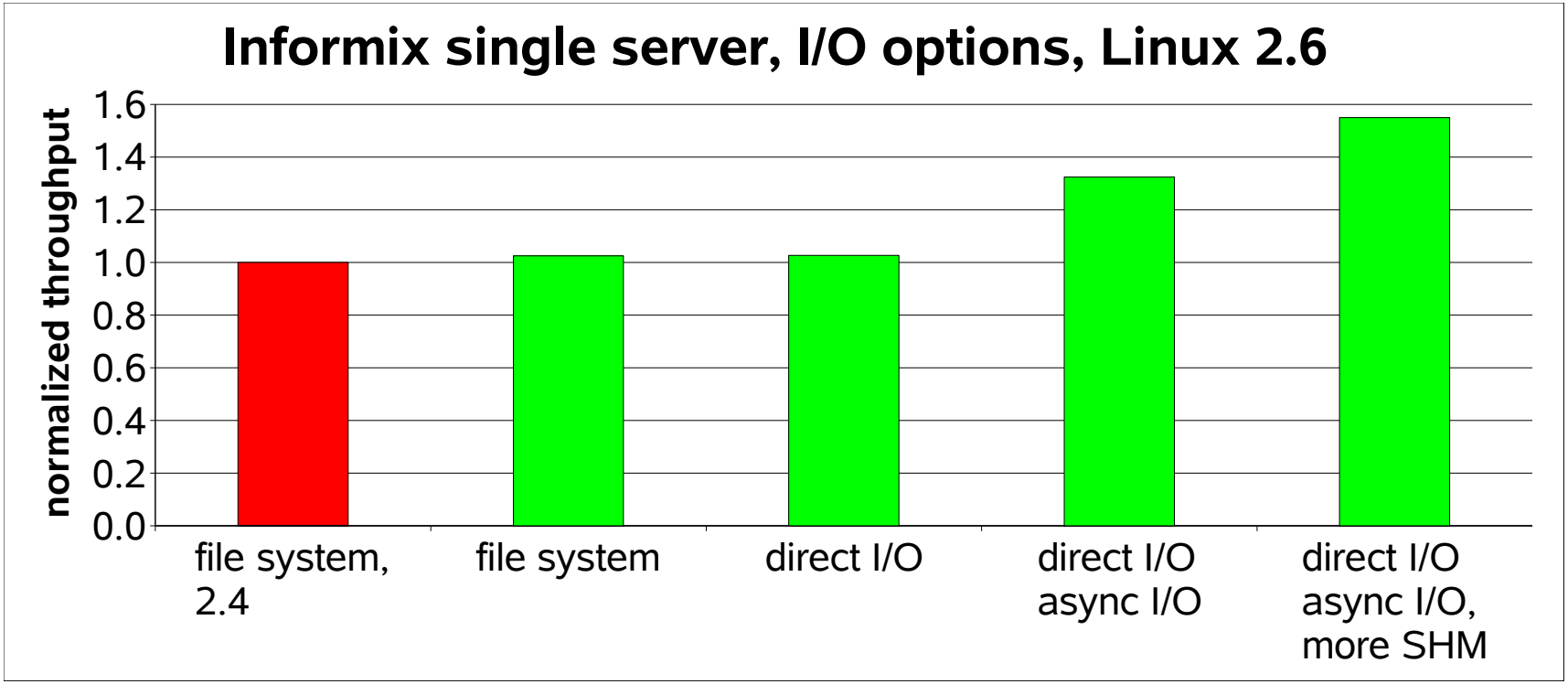


- The caching mode “record” returns the best result for database OLTP
- ESS caching modes are described in
  - Command Reference 2105 Models SC26-7298-xx
- On 2.6 based distros the caching mode can be changed with the tool "tunedasd"

Linux 2.6 Disk I/O Options

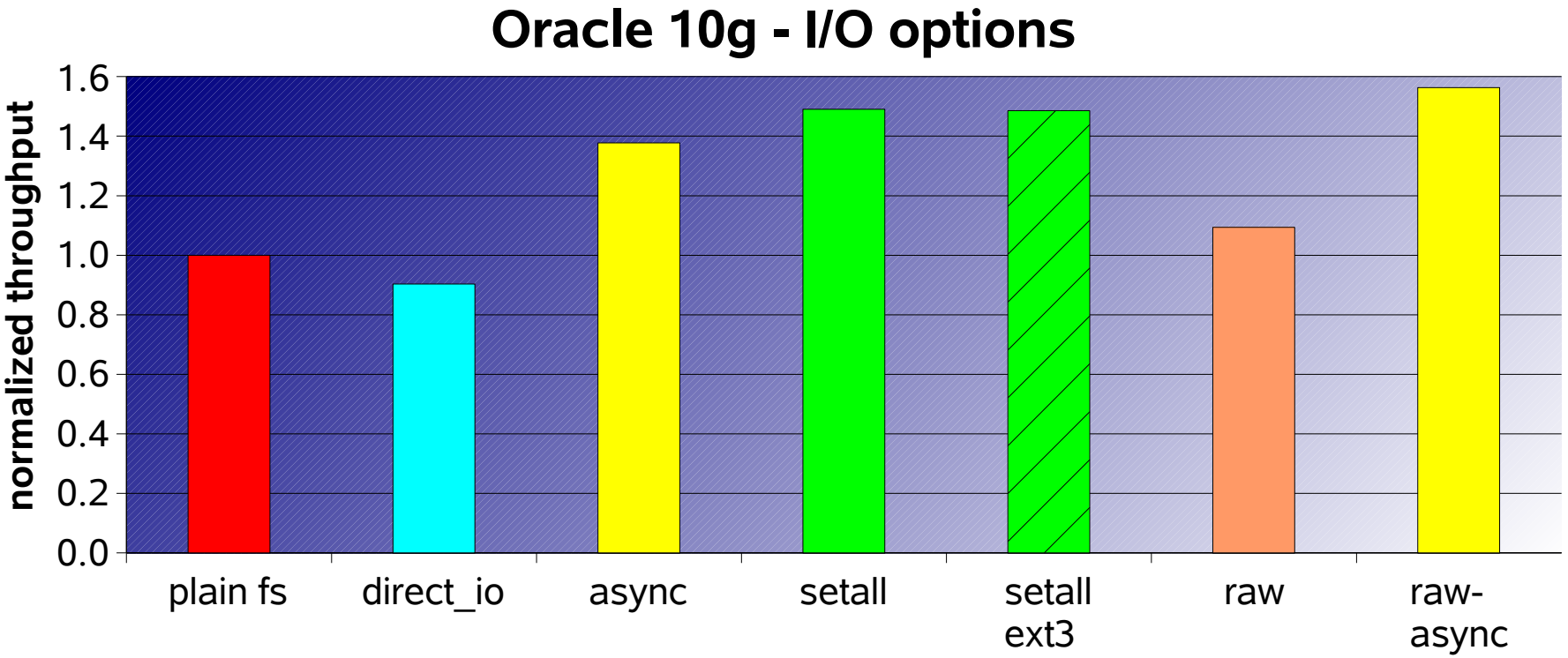
- **new I/O options now available with Informix:**
  - direct I/O on block device
    similar to the raw devices from 2.4,
    now a block device, like /dev/sda1, is used directly
  - async I/O on a block device
    the **issuer of a read/write operation is no longer waiting until the request finishes.**

Linux 2.6 Disk I/O Options - Results



- *the combination of direct I/O and async I/O is a very good improvement*
- *Further enhancements:*
  the dedicated I/O processes of the database are not longer needed, the additional free memory can be used to increase the database buffer in shared memory
- *see: http://www.ibm.com/developerworks/db2/library/techarticle/dm-0503szabo/*

Linux 2.6 Disk I/O Options - Results



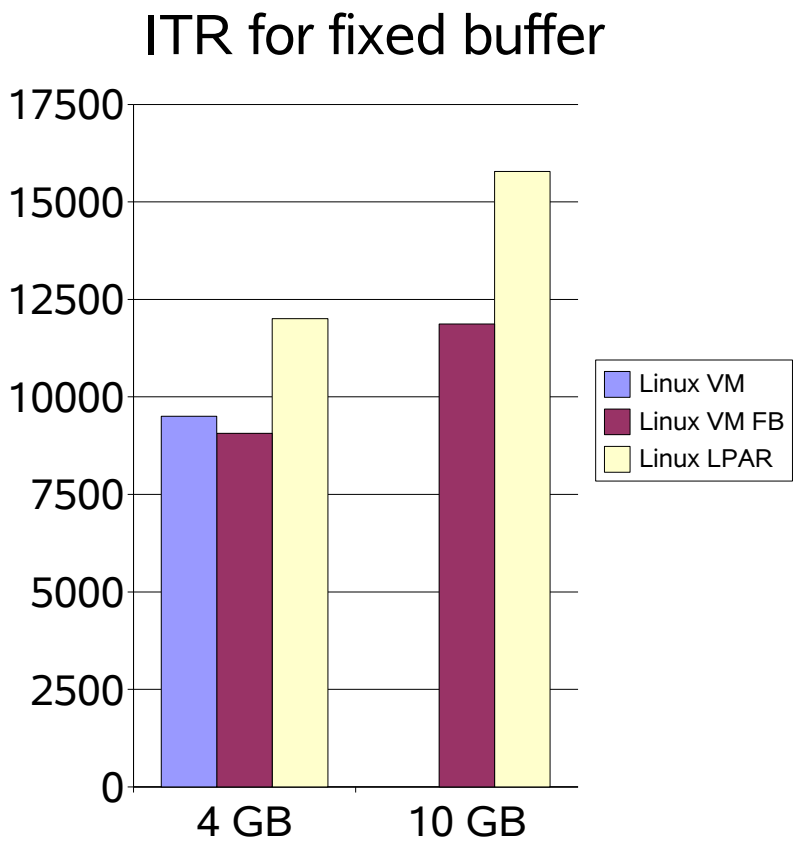
- The combination of direct I/O and async I/O (setall) shows best results when using the Linux file system, raw I/O with asynch I/O was best.
- ext2 and ext3 lead to identical throughput

Fixed IO buffers

- **problem with large z/VM guests doing heavy disk IO**
  - 2 GB for CP can become a bottleneck
  - see http://www.vm.ibm.com/perf/tips/2gstorag.html
- **mitigation for ECKD disks:**
  - fixed io buffers in SLES9 SP1 and RHEL4
    - extra copy for all disk I/O
  - enable using dasd driver kernel parameter “fixedbuffers” e.g.
    - dasd=<dasd device list>,fixedbuffers
    - See http://www.ibm.com/developerworks/linux/linux390/perf/tuning_res_fixed_io_buffers.shtml

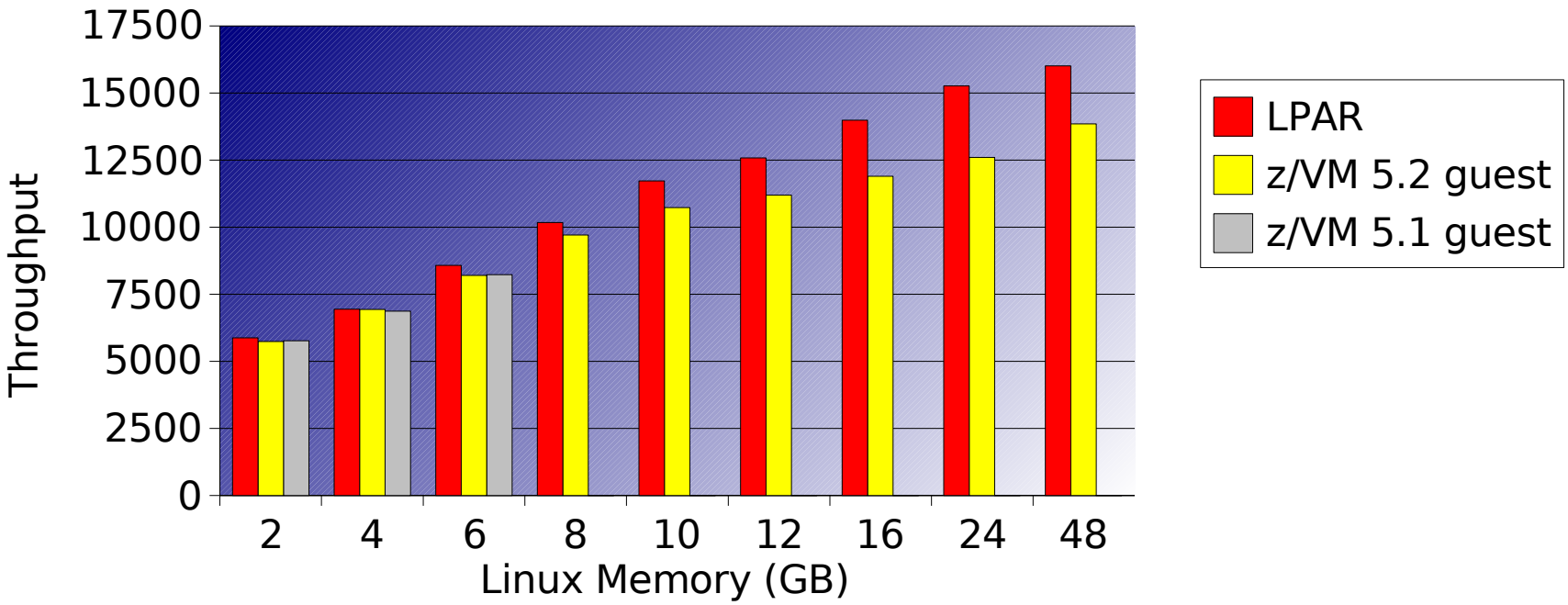
Informix – Fixed IO buffer results - ITR

- **large guest can now be run under z/VM**
- **price to pay:**
  - for smaller guest 4% additional ITR loss
- **LPAR well suited for high utilized Linux**
- **more results:**



http://www.ibm.com/developerworks/linux/linux390/perf/tuning_res_fixed_io_buffers.shtml

Large Linux guests with z/VM 5.2



Large guests can now be run under z/VM without special treatment in the disk device driver

SLES9, pre-GA z/VM 5.2 → preliminary results

Visit us !

- **Linux on zSeries Tuning Hints and Tips**
  http://www.ibm.com/developerworks/linux/linux390/perf/index.html
- **Linux-VM Performance Website**
  http://www.vm.ibm.com/perf/tips/linuxper.html

Questions