Performance Characterization of SPEC CPU2006 Integer Benchmarks on x86-64 Architecture

Dong Ye
David Kaeli
Northeastern University

Joydeep Ray
Christophe Harle
AMD Inc.

Outline

- Motivation and background
- Performance characteristics of CPU2006 integer benchmarks on x86-64 (64-bit mode vs. 32-bit mode)
- Program characteristics of selected benchmarks
- CPU2006 vs. CPU2000

Motivation and background

- x86-64 architecture brings 64-bit computing to the PC market
  - Need to evaluate whether PC desktop applications can benefit from 64-bit ISA
- SPEC released its latest CPU suite (CPU2006) last month
  - Want to evaluate how applications have changed when moving from CPU2000 to CPU2006

x86-64: extends x86 to 64 bits

- x86-64==x64==(AMD64 + EM64T)
- Fully compatible with existing x86 modes
- Architectural support for 64 bit virtual address space and 52 bit physical address space
- 64-bit mode supports flat addressing
- 64-bit integer operations
- 16 64-bit GPRs, 16 SSE registers

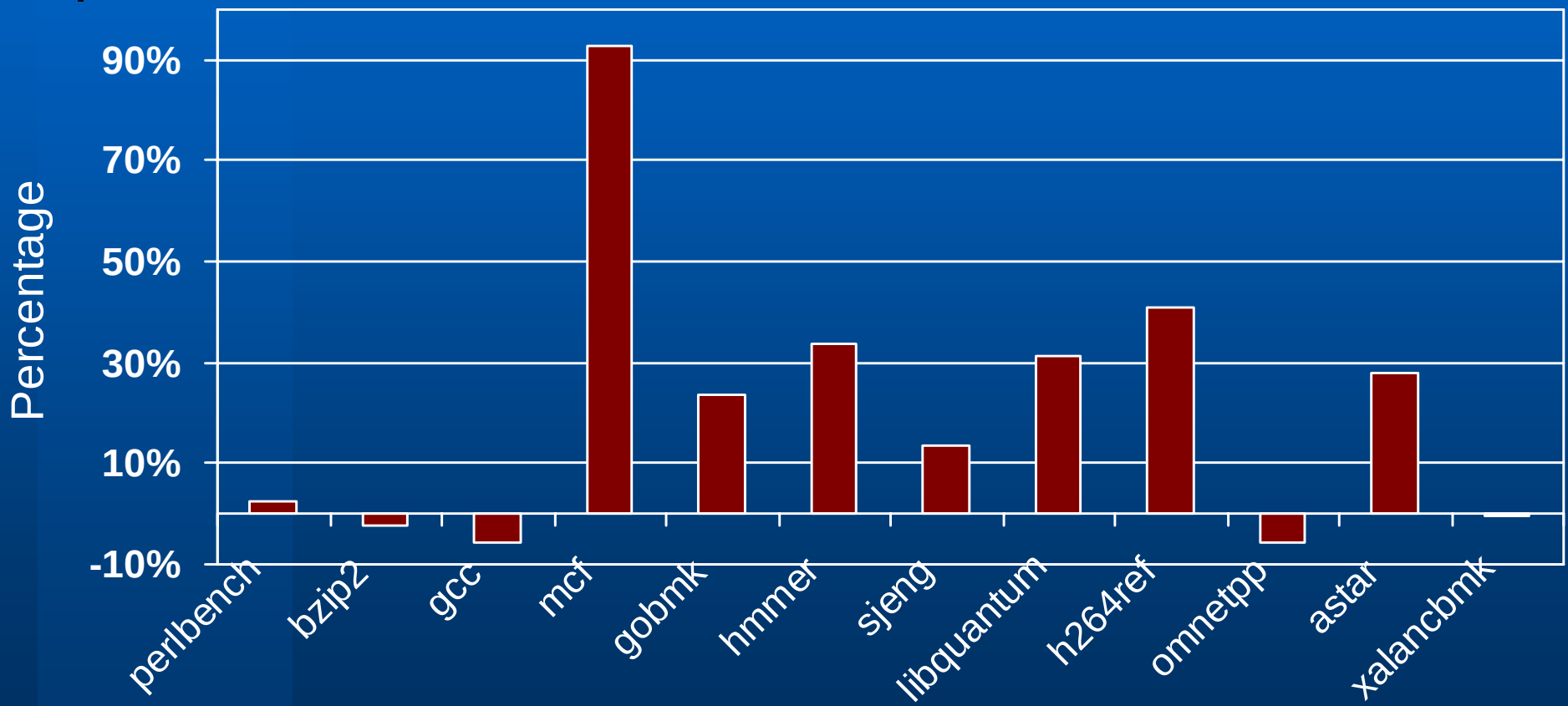
Evaluation environment

- Athlon 64 X2 4400+ Rev E (dual-core, 2.2GHz)
- 2 DIMMs 1GB DDR400 DRAM
- SUSE Linux 9.3 Pro x86-64 edition, run level 3, selected daemons are disabled (kernel 2.6.11.4)
- Benchmark bound to run on a single core
- 64-bit binary run in the 64-bit mode, 32-bit binary run in compatibility mode (referred to as 32-bit mode), both on the same 64-bit OS
- GCC 4.1.1 (-O2 for perlbench, -O3 for all others)
- H/W counters used to collect performance data

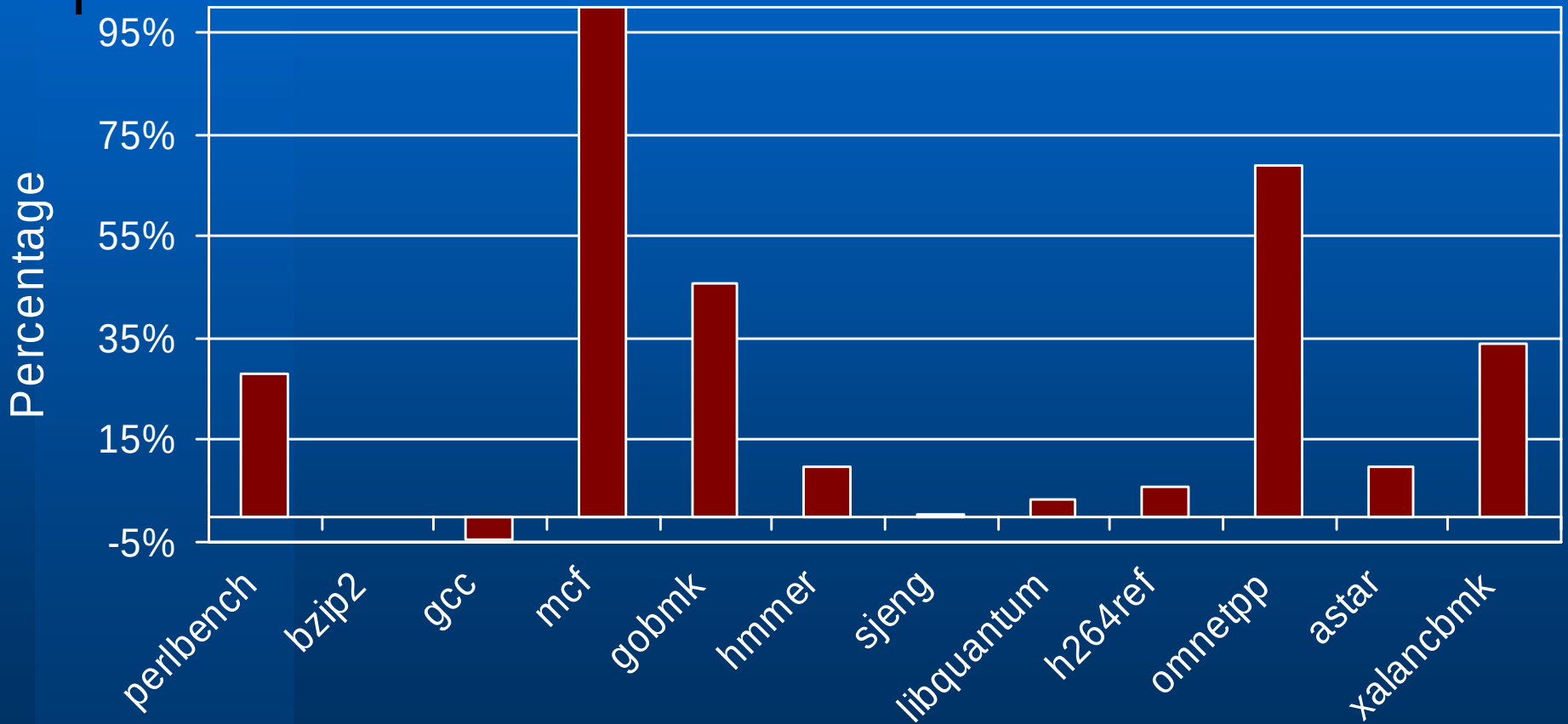
How much faster is 64-bit mode?

Benchmark	Language	64-bit vs. 32-bit speedup
perlbench	C	3.42%
bzip2	C	15.77%
gcc	C	-18.09%
mcf	C	-26.35%
gobmk	C	4.97%
hmmer	C	34.34%
sjeng	C	14.21%
libquantum	C	35.38%
h264ref	C	35.35%
omnetpp	C++	-7.83%
astar	C++	8.46%
xalancbmk	C++	-13.65%
Average		**7.16%**

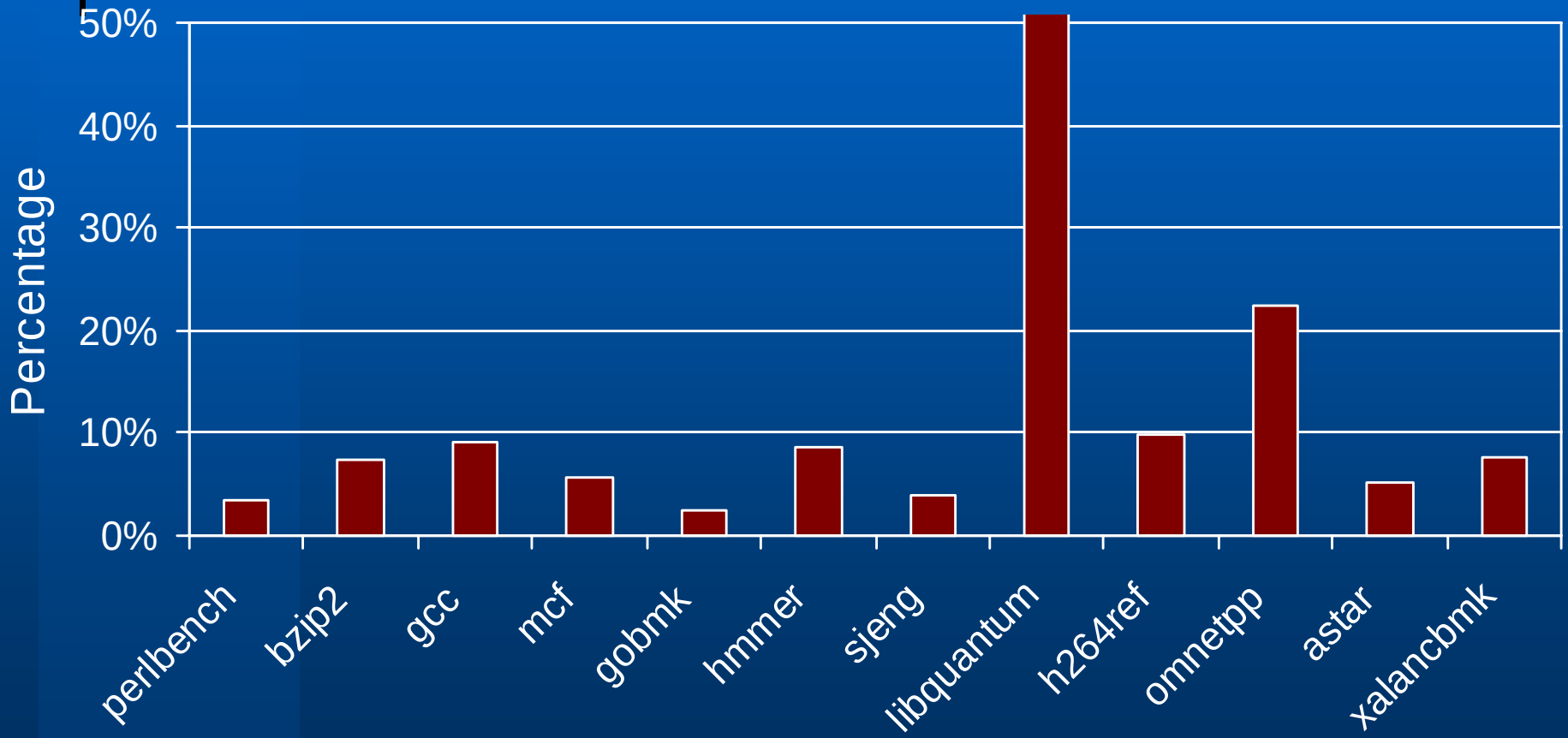
Code size ***increases*** in 64-bit mode



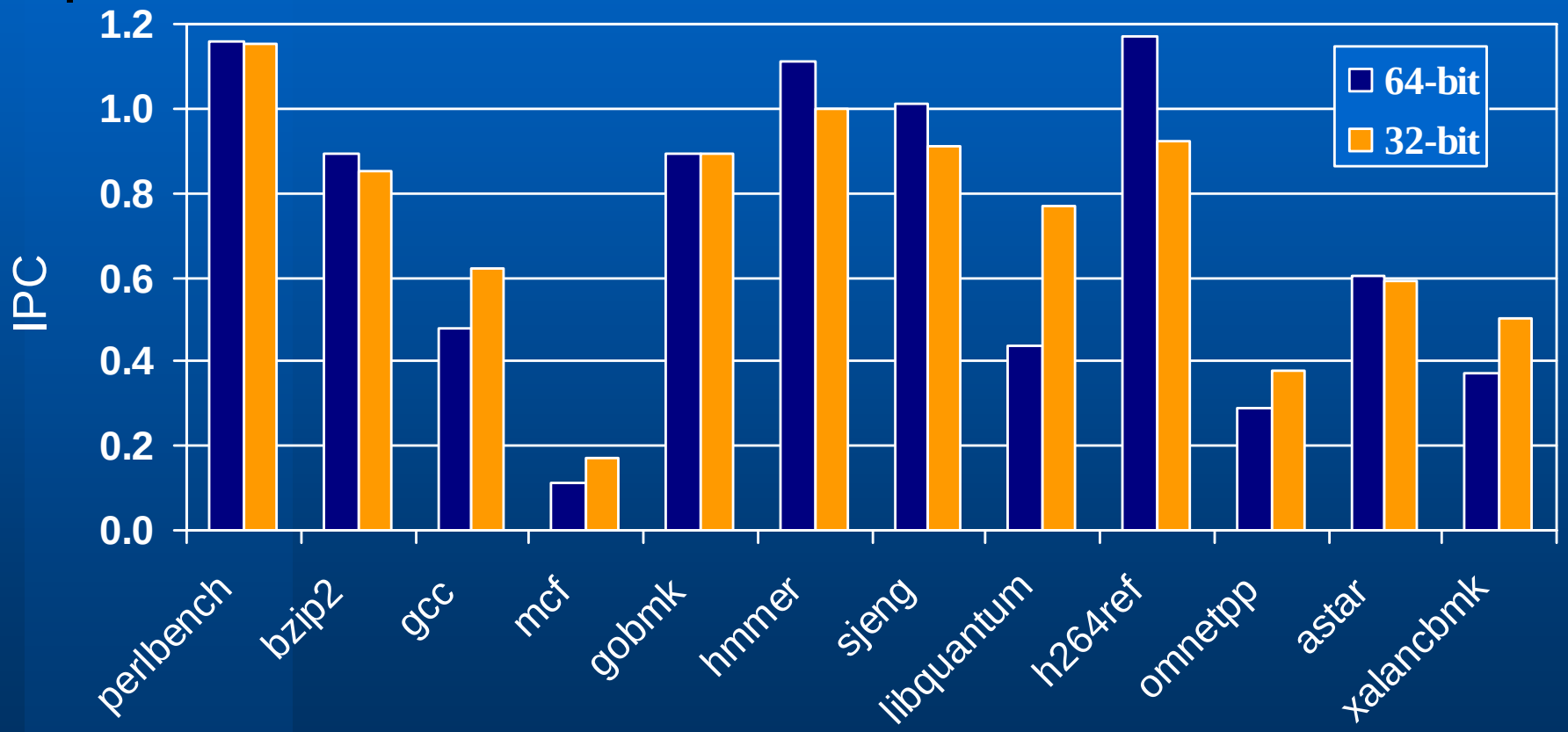
Runtime memory footprint ***increases*** in 64-bit mode



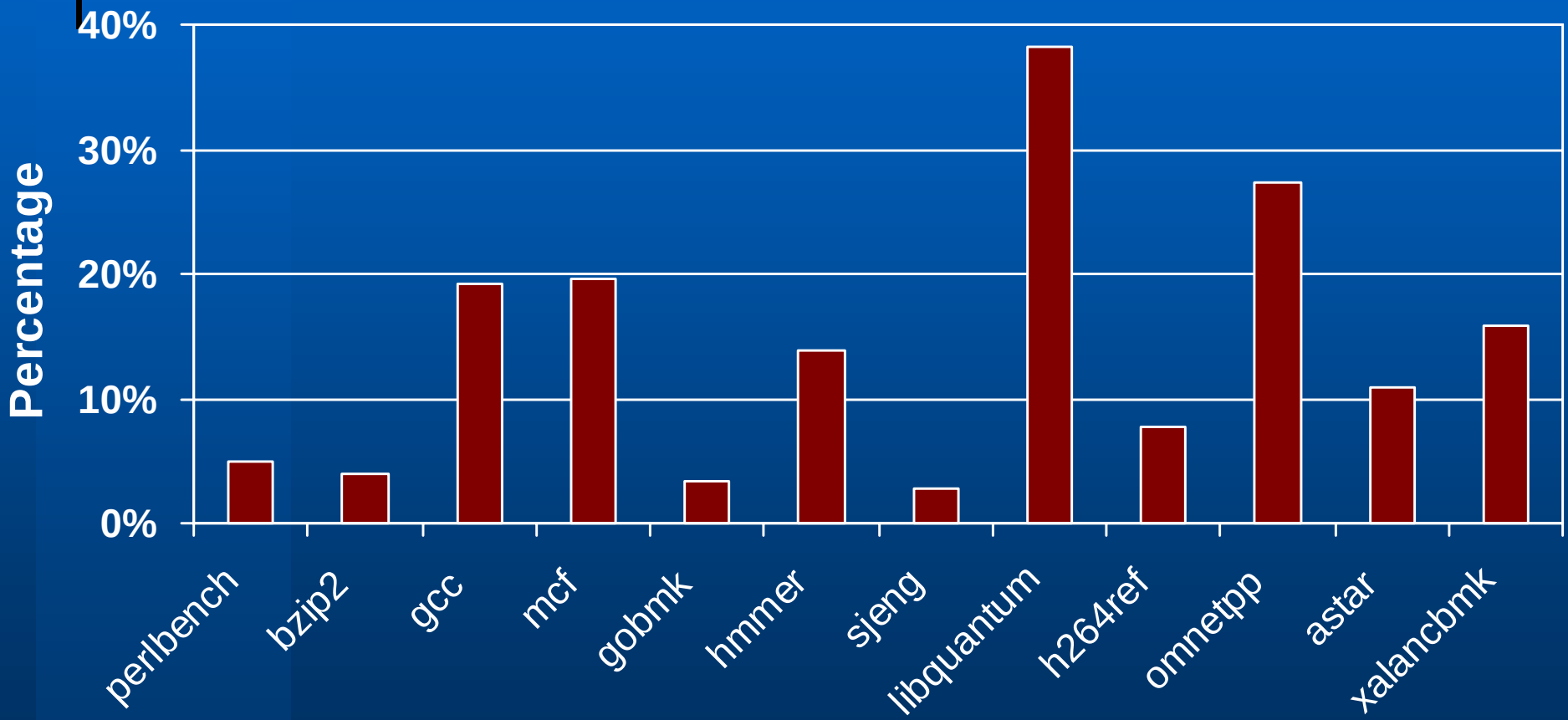
Dynamic instruction count ***decreases*** in 64-bit mode



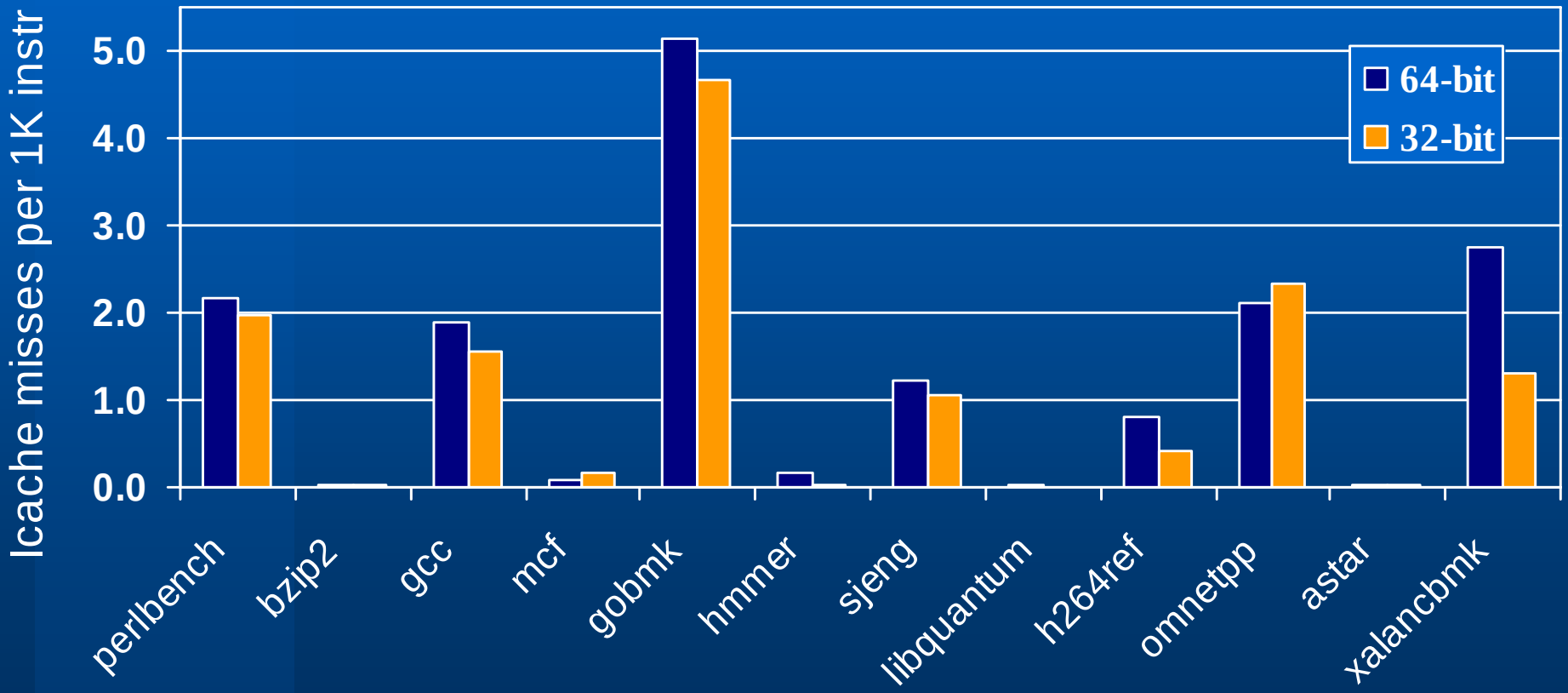
IPC comparison



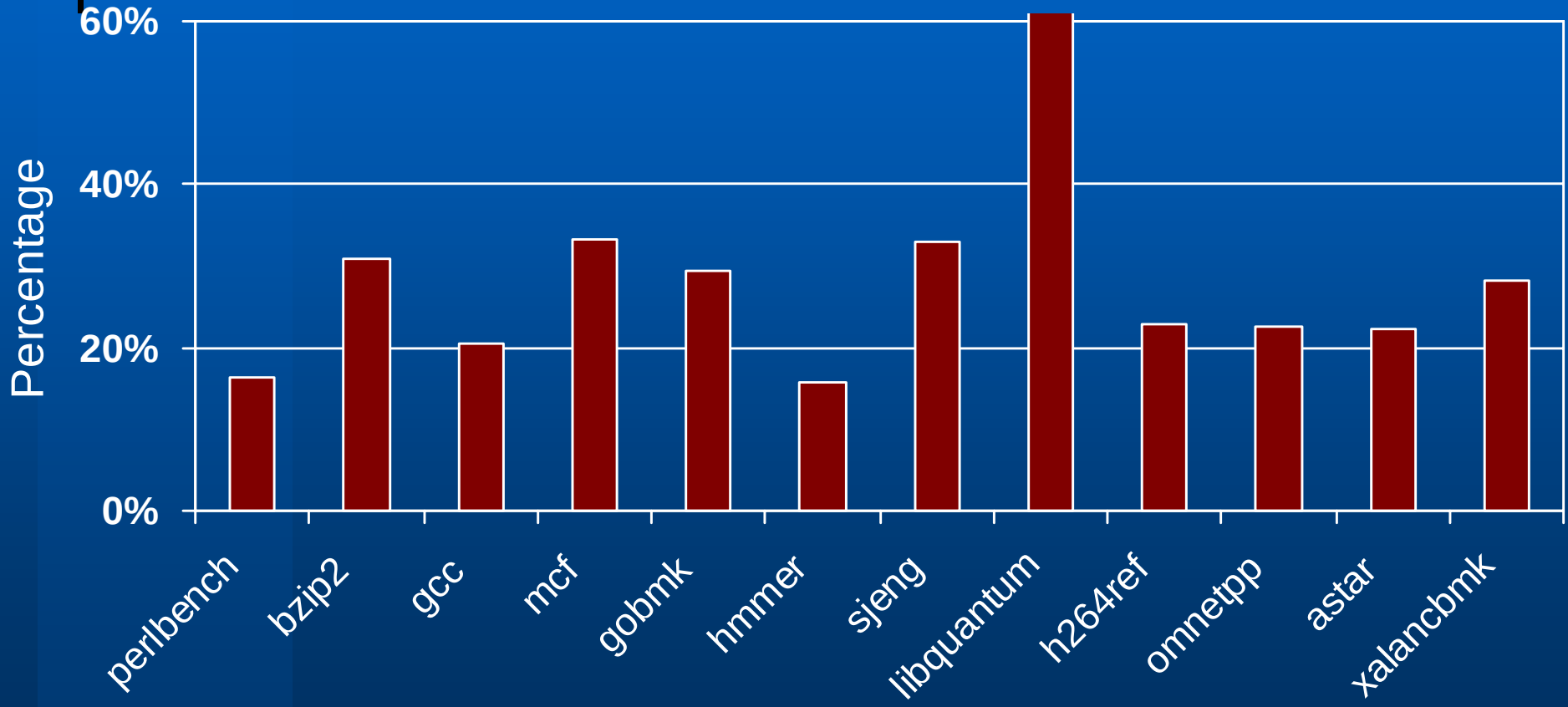
Instruction cache (L1) request rate ***increases*** in 64-bit mode



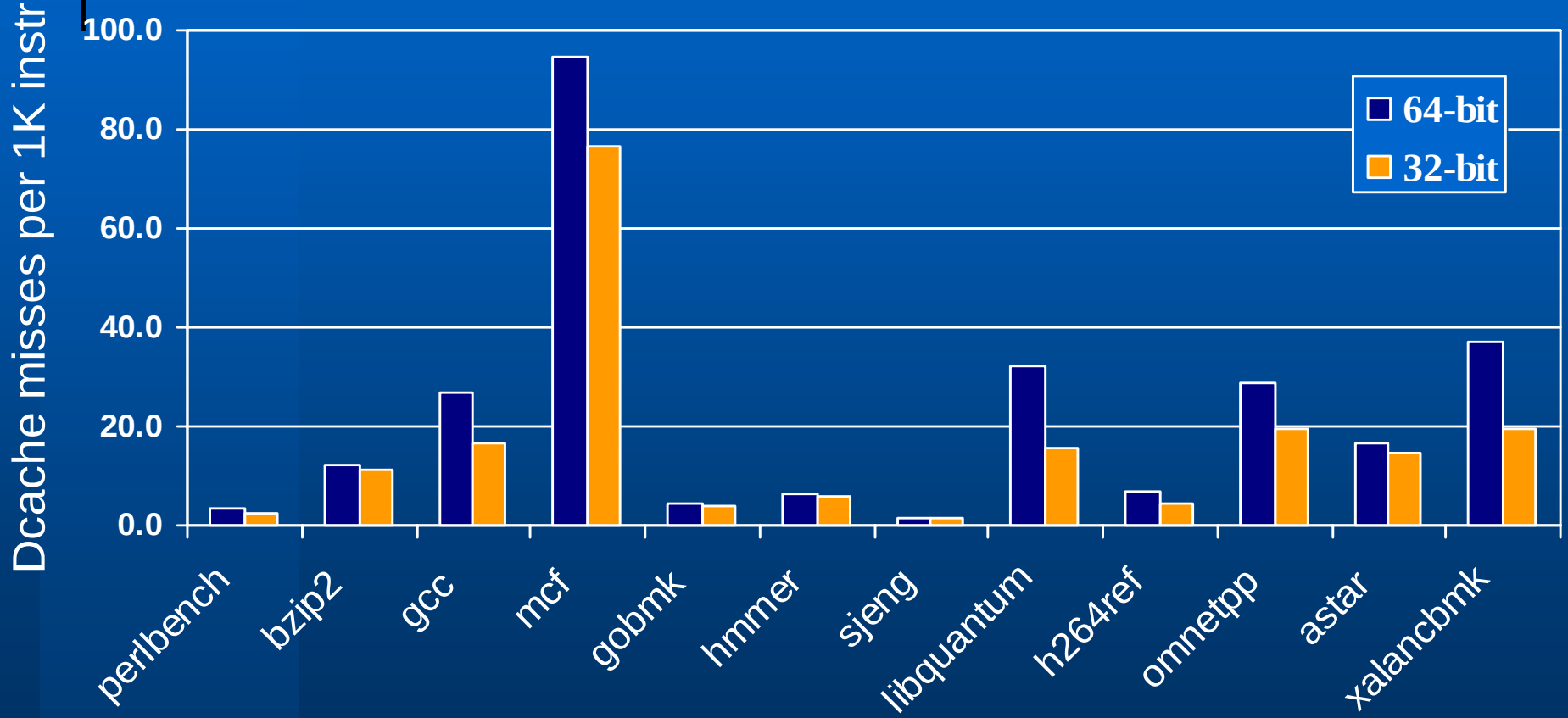
Instruction cache miss rate comparison



Data cache (L1) request rate ***decreases*** in 64-bit mode



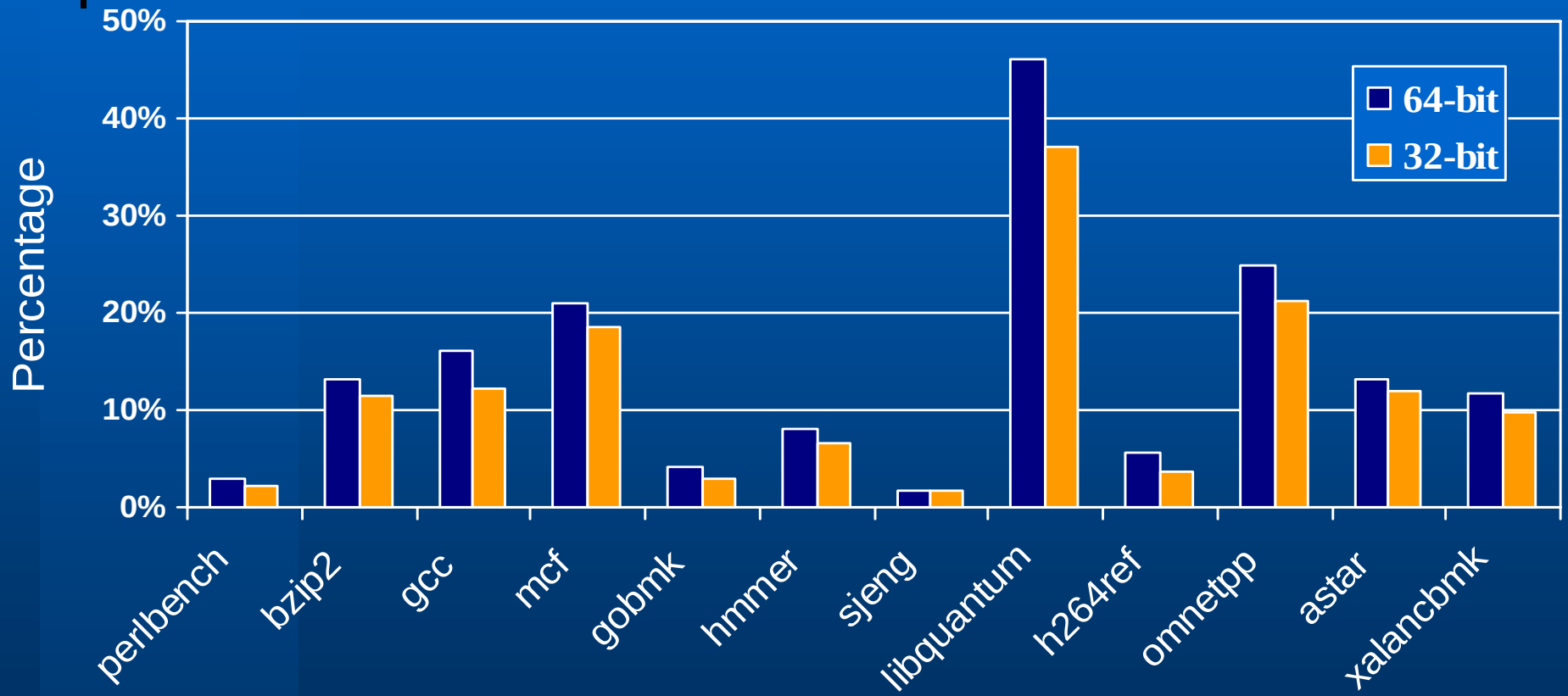
Data cache miss rate comparison



Observations

- Instruction cache miss rate is very low in both 64-bit and 32-bit modes
- Data cache request rate decreases significantly in 64-bit mode
  - Extra registers help
- Data cache miss rate increases in 64-bit mode
  - The increased size of long and pointer data types has an adverse impact on data cache performance
  - A lower instruction count magnifies this

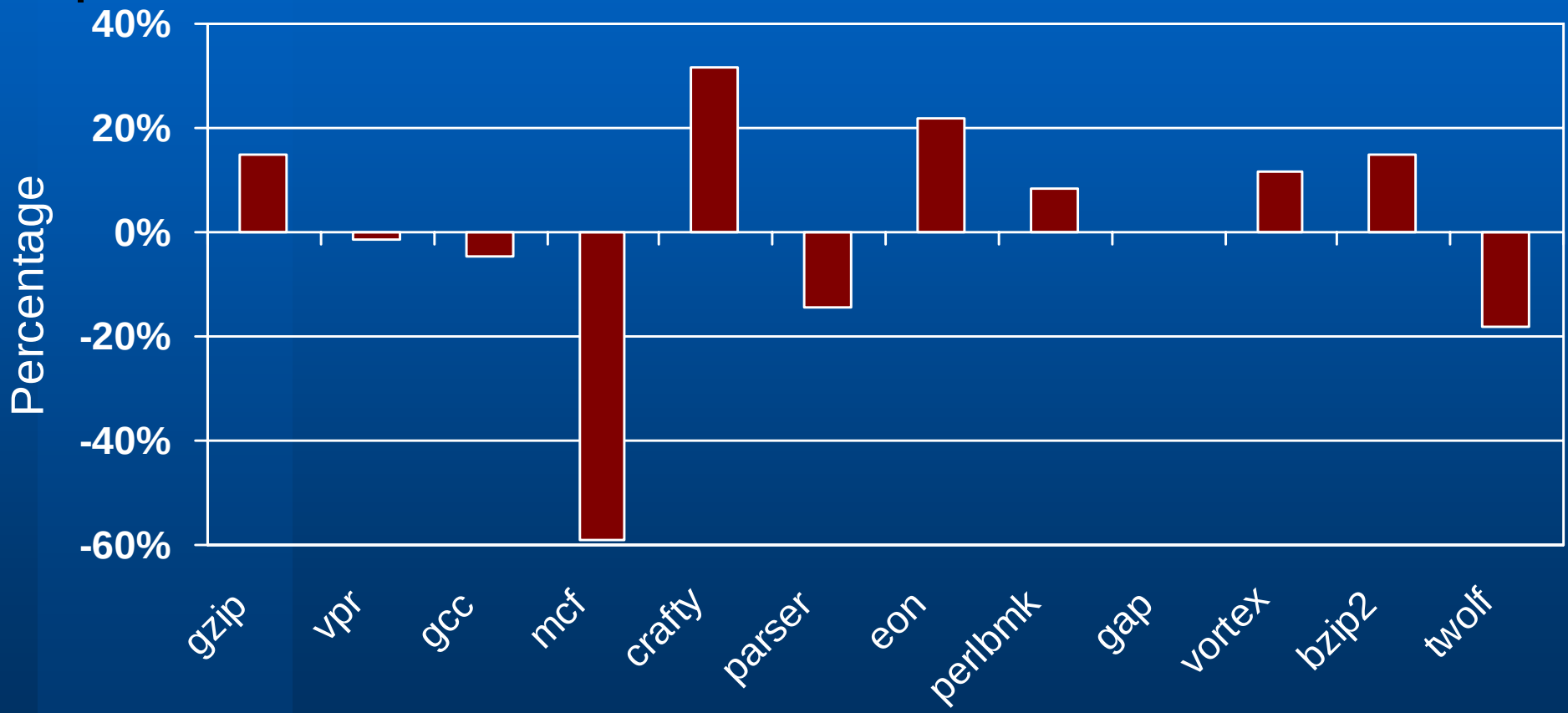
Memory controller utilization ratio comparison



Program characteristics of five selected benchmarks

Benchmark	Performance	Observations	Root cause analysis
mcf	64-bit: 26.4% slower	Memory footprint doubles	Larger memory footprint due to extensive uses of longs and pointers
xalancbmk	64-bit: 13.7% slower	Memory footprint increases by 33.9%	Larger memory footprint due to extensive uses of pointers
hmmer	64-bit: 34.3% faster	Dynamic instruction count decreases by 8.7%	More registers available in 64-bit mode
libquantum	64-bit: 35.4% faster	Dynamic instruction count decreases by 54%	Native 64-bit integer arithmetic in 64-bit mode
h264ref	64-bit: 35.4% faster	Dynamic instruction count decreases by 10%	Faster calling convention (mainly because of more registers) in 64-bit mode

CPU2000int: Speedup in 64-bit mode



Conclusions

- Programs that use features in 64-bit mode (64-bit integer arithmetic and more registers) stand to benefit more from x86-64
- Programs that are memory intensive or make heavy use of long and pointer need to carefully evaluated when porting to x86-64

Disclaimer

- **All performance numbers referred to in this presentation are ‘estimates’ because they are from a ‘peak-only’ SPEC CPU2006 run, and hence is not fully compliant with SPEC run rules. It is expected, though not proven, that results from a fully compliant run would be very close.**

Thank you and questions?