Power and Thermal Characterization of POWER6 System

Víctor Jiménez[†], Carlos Boneti[*], Francisco J. Cazorla[†], Roberto Gioiosa[†], Eren Kursun[‡], Chen-Yong Cher[‡], Canturk Isci[‡], Alper Buyuktosunoglu[‡], Pradip Bose[‡], Mateo Valero[†]

[†] Barcelona Supercomputing Center, Barcelona (Spain)

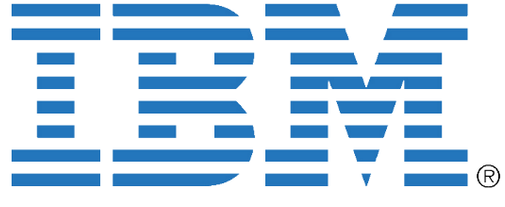
[*] Schlumberger BRGC, Rio de Janeiro (Brazil)

[‡] IBM T.J. Watson Research Center, Yorktown Heights (USA)

September 13th, 2010

Vienna (Austria)

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

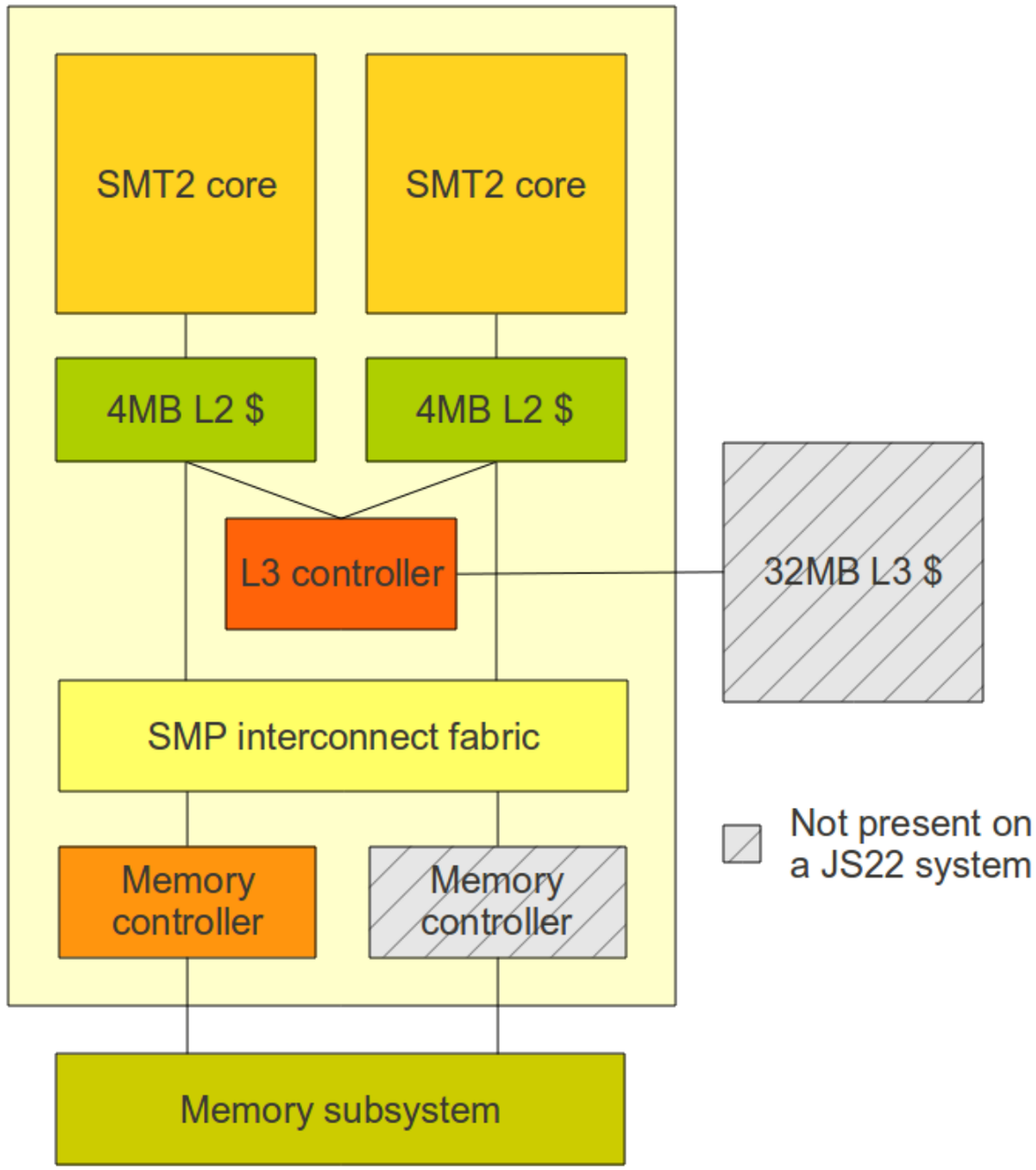


Outline

- Background
- Methodology
- Characterization
  - Idle system
  - Active system
- Power model
- Workload-aware thread placement

Background – POWER6 (JS22)

- JS22 has two POWER6 chips
- Dual-core SMT2
- High-frequency (4GHz) in-order
  - OoO for some FP operations
- 64KB L1 I-cache and D-cache
- Per-core 4MB L2 cache
- Optional off-die 32MB L3 cache
- 1 or 2 memory controllers
  - Depending on configuration



Background – POWER6 (JS22)

- Nap mode
  - Per-core low-power mode
  - Turns off the internal clocks
  - Reduces power consumption and temperature
- Hardware thread priorities
  - Control instruction decode rate for each thread in a core
  - Eight priority levels
    - Special case (1,1) : power saving operation
  - Throughput and execution time can be improved
    - Boneti et al. *Software-Controlled Priority Characterization of POWER5 Processor*. ISCA 2008
  - Address biased thread performance
    - Boneti et al. *A Dynamic Scheduler for Balancing HPC Applications.* SC 2008

Background – Linux Kernel

- CPU Idle Power Manager
  - No process is available to run (other than idle process)
  - Takes advantage of underlying HW low-power mechanisms
- Tickless kernel
  - Frequent timer interrupts (hundreds per second)
  - Interrupts force the system to exit low-power mode
  - Tickless kernel removes periodic timer interrupts
    - Timer set to expire to the next, non-periodic timer event

```
idle_loop:
  while (get_tb() < start_snooze) {
    If (…) goto out;
    …
    HMT_very_low(); /* priority 1 */
  }

  HMT_medium(); /* priority 4 */
  …
  cede_processor(); /* nap mode */
out:
  HMT_medium(); /* priority 4 */
```

Linux idle loop snippet for POWER6

Methodology

- IBM JS22 BladeCenter
  - Two dual-core, 2-way SMT POWER6 chips @ 4.0Ghz
- Power/temperature measurements
  - IBM EnergyScale architecture
  - Accurate measurements via Thermal and Power Management Device (TPMD)
- Benchmarks
  - METbench microbenchmarks
    - Stress different subcomponents (integer unit, FP unit, L1/L2 cache, memory)
  - SPEC CPU2006
- Metrics
  - Energy-delay product : EDP = power / IPC^2 (lower is better)

Results

- Idle system
  - Low-power modes
  - Tickless kernel
- Active system
  - Workload characteristics
  - Core usage effect
  - Hardware thread priorities

Results – Low-Power Modes

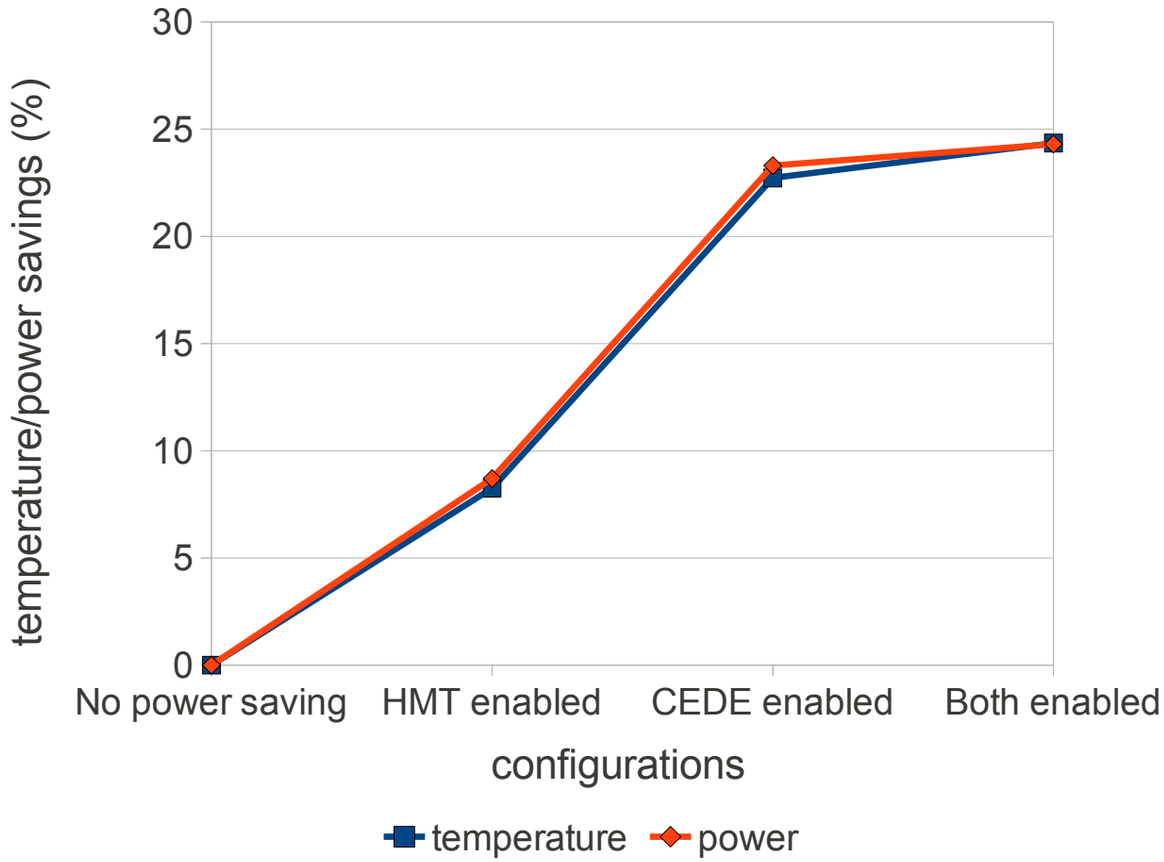
- Four configurations (system is idle)

1) *No power saving* (both nap mode and HW thread priorities are disabled)

2) *HMT enabled* (only priorities are enabled)
   - Low-power priority set (1,1) is used
   - Very low latency

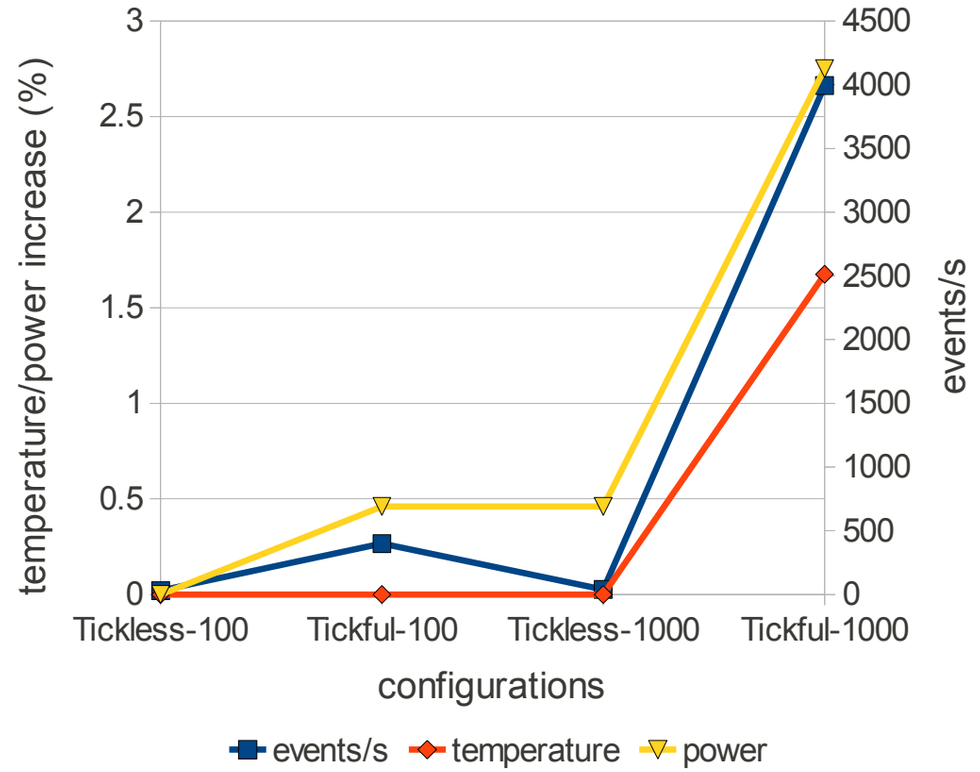
3) *CEDE enabled* (only nap mode is enabled)
   - Higher latency

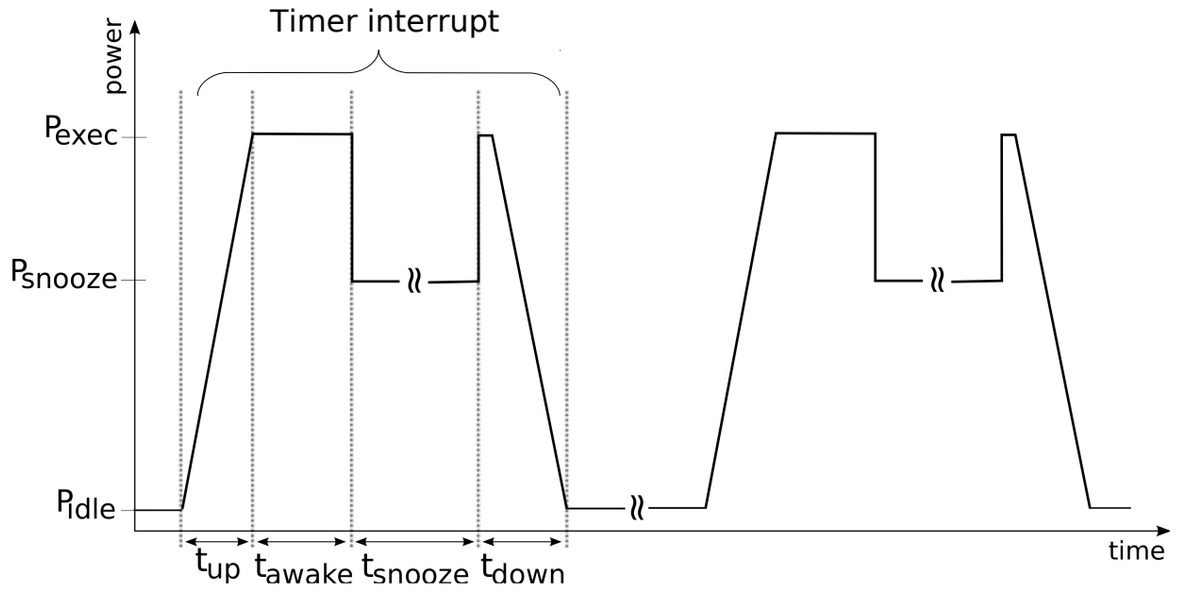
4) *Both enabled* (both nap mode and HW priorities are enabled)



Results – Tickless Kernel

- Four configurations (system is idle)
  - Tickless/tickful
  - 100/1000 timer interrupts per second
- We collect
  - OS events
  - Power/temperature measurements
- Non-significant effect on power
  - On a POWER6 system (for HZ=100)
    - HZ=100 is typical for a server
- Analytical model of tickless effect on power
  - Accurate estimation

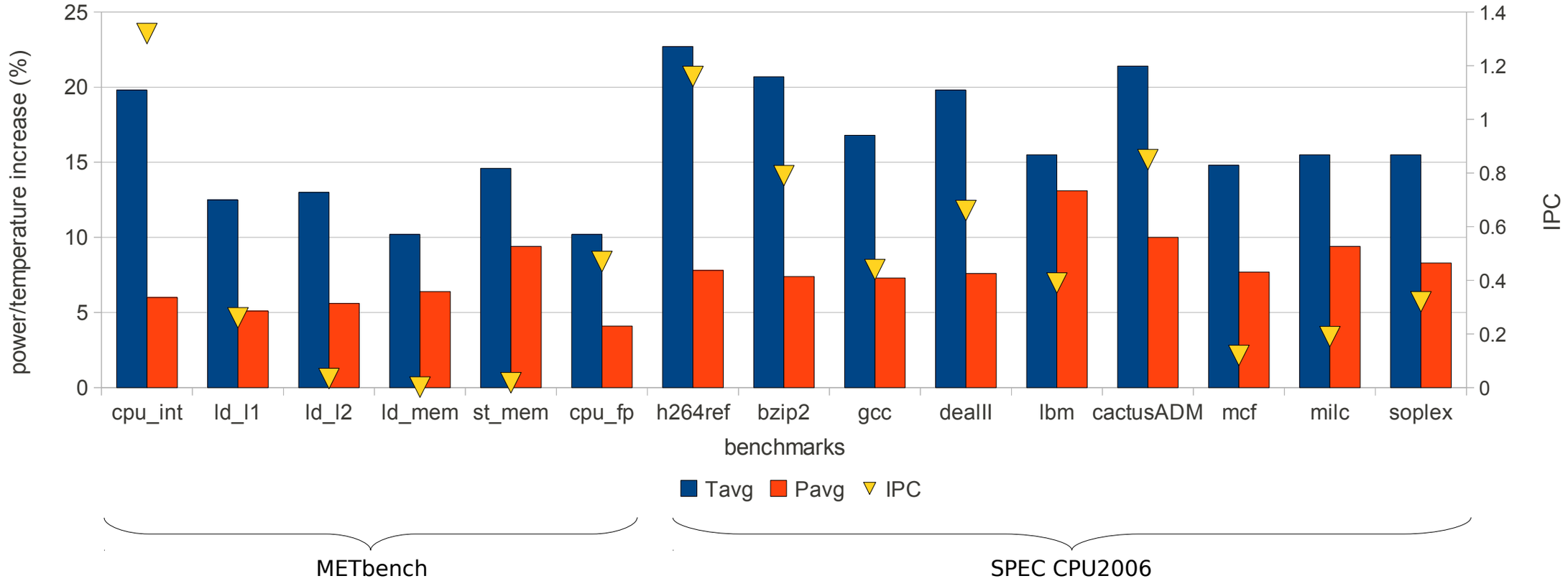




Results

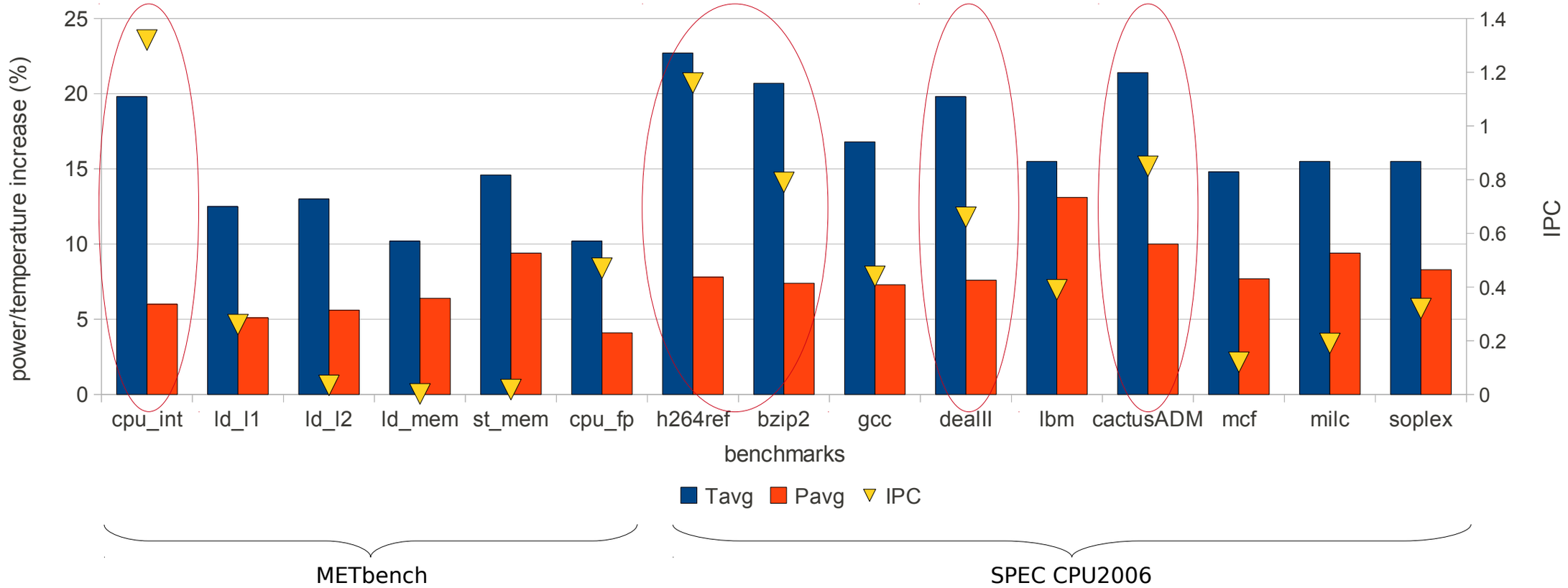
- Idle system
  - Low-power modes
  - Tickless kernel
- Active system
  - Workload characteristics
  - Core usage effect
  - Hardware thread priorities

Results – Workload Characteristics (single thread)



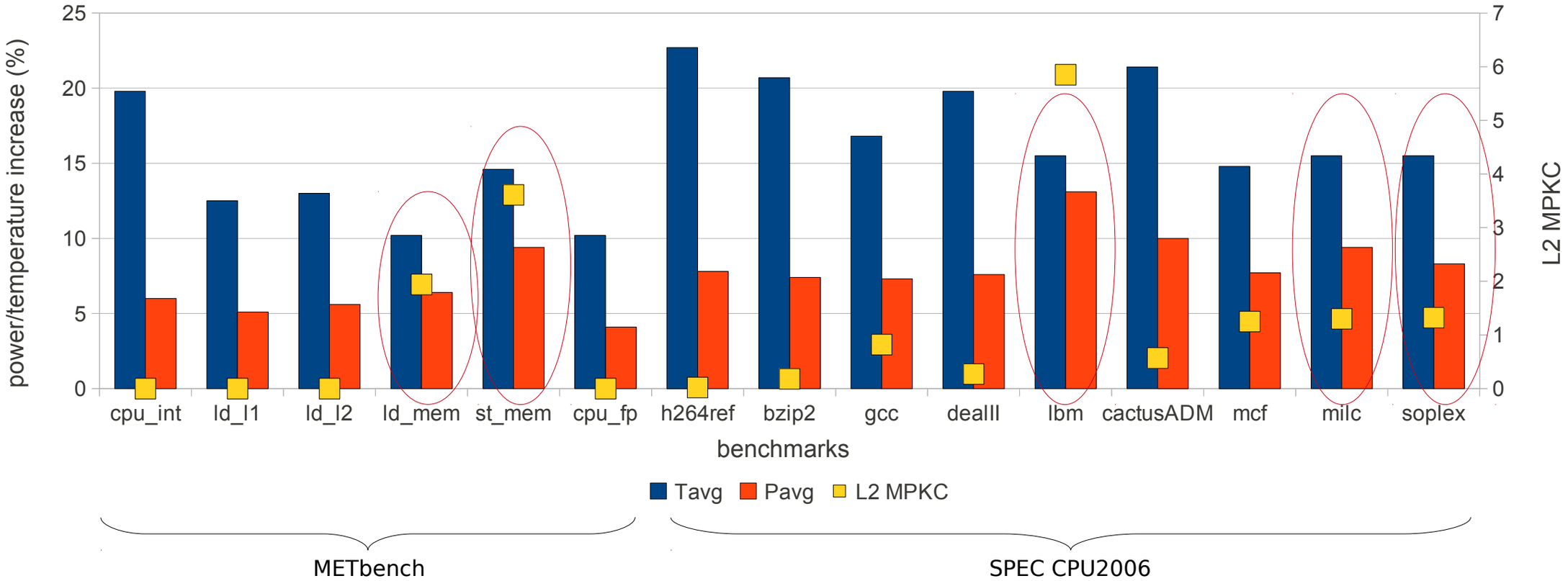
- Effect of CPU and memory intensity on system power and core temperature
  - Power and temperature values are relative to their values when the system is idle

Results – Workload Characteristics (single thread)



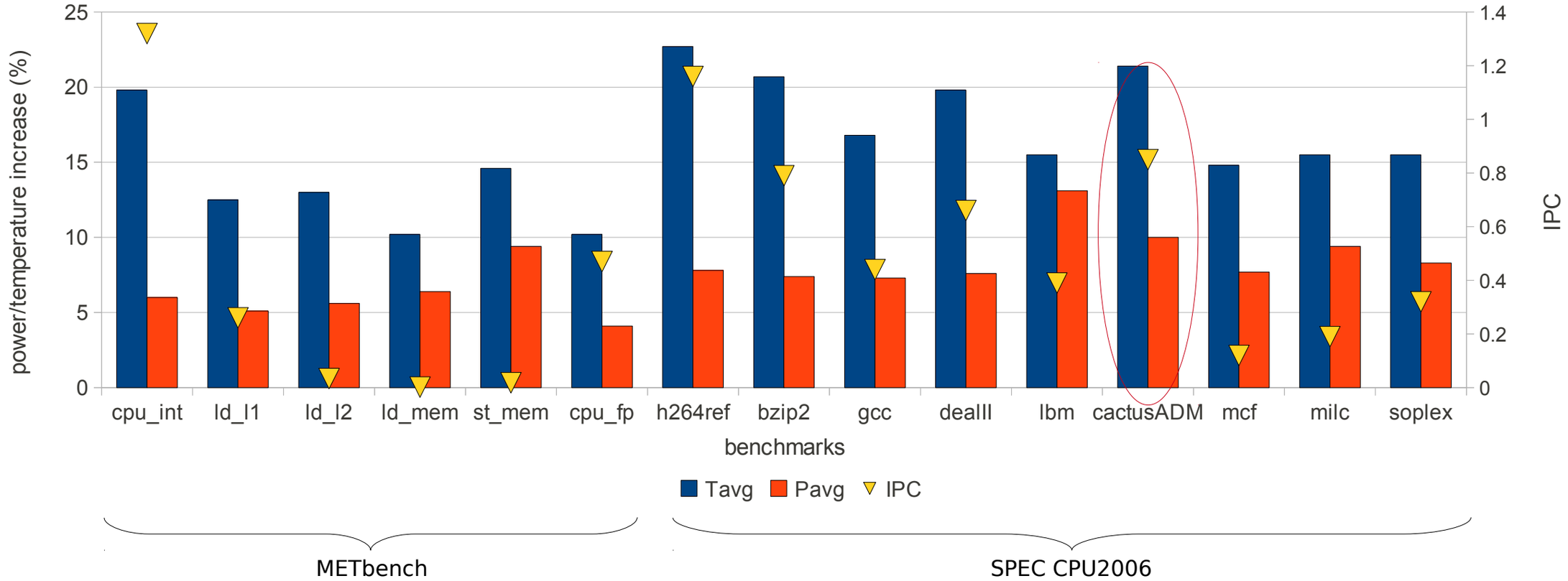
- Effect of CPU and memory intensity on system power and core temperature
  - Temperature is correlated with CPU intensity (high-IPC benchmarks)
    - Up to 9.6% variation

Results – Workload Characteristics (single thread)



- Effect of CPU and memory intensity on system power and core temperature
  - Temperature is correlated with CPU intensity (high-IPC benchmarks)
  - Power consumption is correlated with memory intensity...
    - Up to 5.8% variation

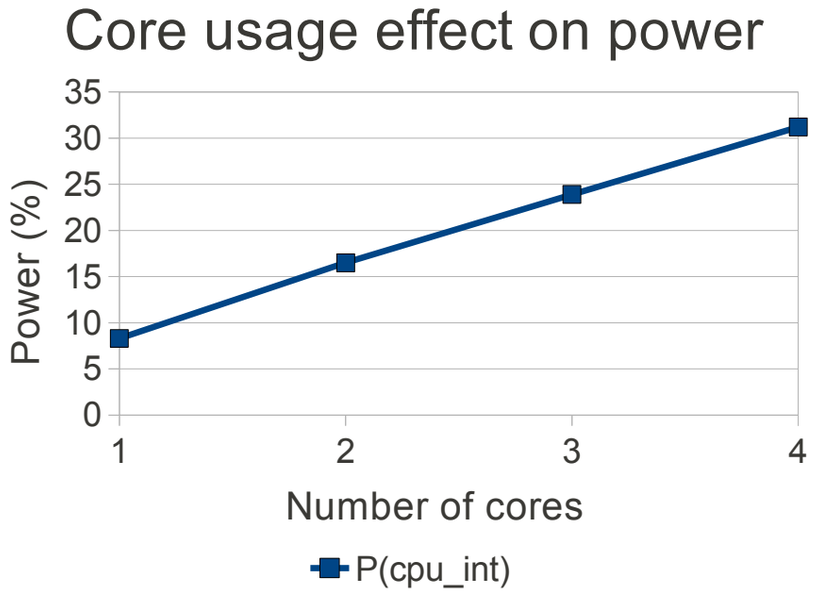
Results – Workload Characteristics (single thread)



- Effect of CPU and memory intensity on system power and core temperature
  - Temperature is correlated with CPU intensity (high-IPC benchmarks)
  - Power consumption is correlated with memory intensity...
  - … and CPU intensity as well

Results – Core Usage Effect

- Incremental execution of multiple microbenchmark copies
  - CPU-bound (cpu_int), MEM-bound (ld_mem)
- Power consumption increases linearly wrt. to the number of copies
  - No significant difference between using one or two chips
- Performance scales linearly for CPU-bound workloads
- For MEM-bound workloads there is intra-chip saturation
  - Most probably as there is only on memory controller per chip

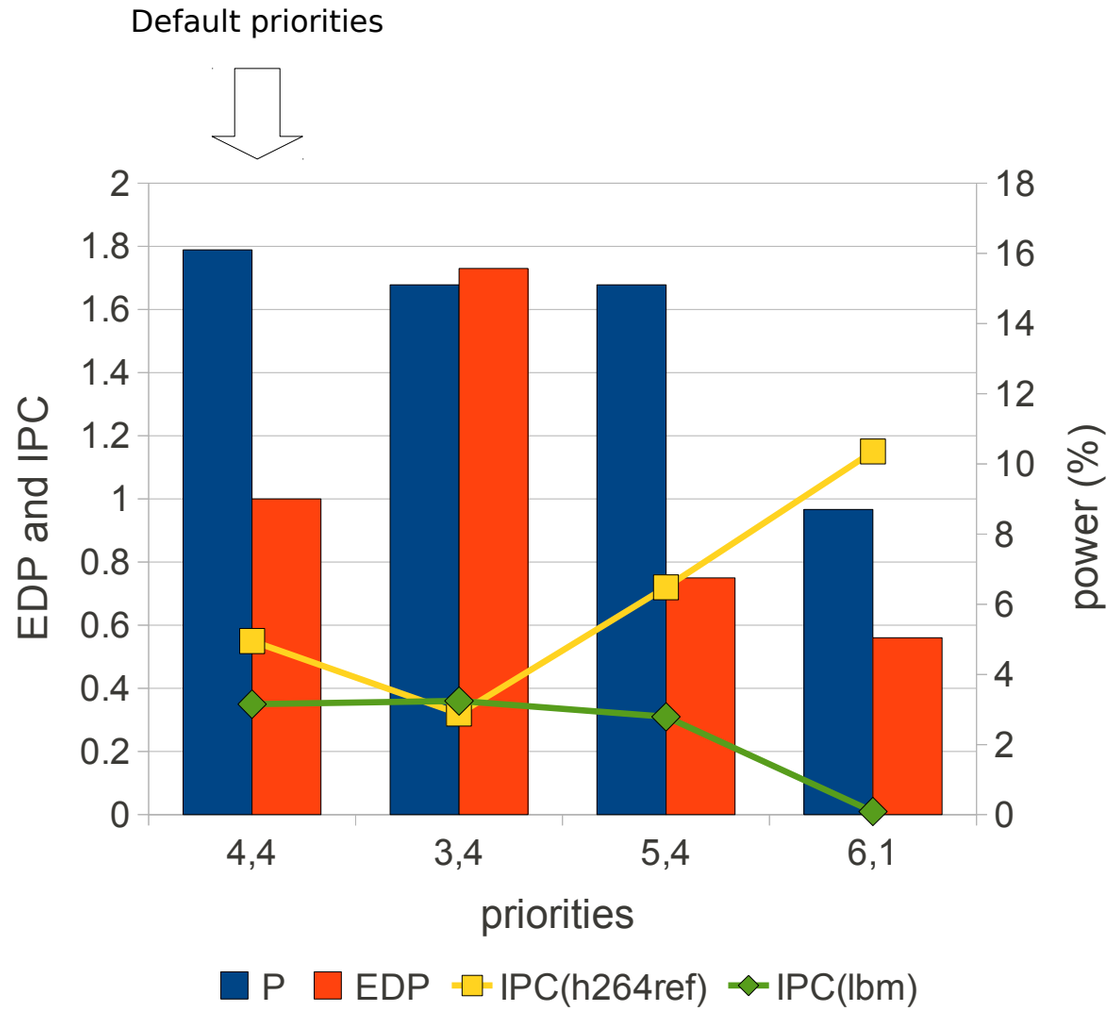


IPC (cpu_int)				
Core 1	Core 2	Core 3	Core 4	Total
1.7				1.7
1.7	1.7			3.4
1.7	1.7	1.7		5.1
1.7	1.7	1.7	1.7	6.8

IPC (ld_mem)				
Core 1	Core 2	Core 3	Core 4	Total
0.0034				0.0034
0.0022	0.0022			0.0044
0.0020	0.0020	0.0032		0.0072
0.0022	0.0022	0.0022	0.0022	0.0088

Results – Thread Priorities

- Heterogeneous mix: CPU-bound and MEM-bound workloads
- Increasing priority for MEM-bound thread
  - No significant performance benefit for lbm
    - Performance for h264ref decreases
  - EDP worsens by 73% (3,4)
- Increasing priority for CPU-bound thread
  - Performance benefit for h264ref
  - 25% improvement in EDP (5,4)
    - Without significantly hurting lbm
  - 44% improvement in EDP (6,1)
    - At the expense of hurting IPC for lbm → still, it can be useful under some circumstances
    - Moreover, power consumption is actually reduced



Results - Applications

- Power model
- Thread placement effect

Results – Power Model

- Access to power sensors is not easy for the end-user
- We provide a power model to overcome this difficulty
  - Based on performance counters (PMCs)
  - End-user can understand/predict power consumption for his/her applications
- Cores and memory are the biggest contributors to dynamic power consumption
  - Modeled by using core activation cost, IPC, and memory accesses
- Model obtained via linear regression

$$P = N_{AC} \times P_{AC} + C_1 \times IPC + C_2 \times L1LDMPC + C_3 \times L2LDMPC + C_4 \times L2STMPC$$

N_{AC} : number of active cores

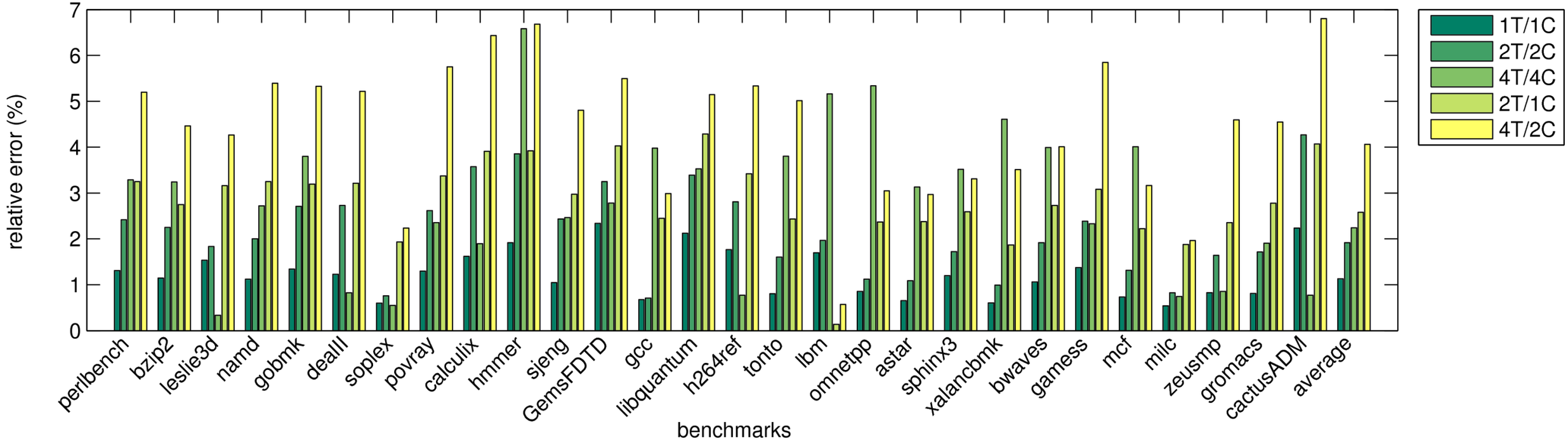
P_{AC} : power consumption due to a core activation

Results – Power Model

- Two approaches for training the model

1) **METbench training:** Training → METbench / Testing → SPEC CPU2006
   - Time for collecting training data is significantly reduced
   - However, less accuracy is expected
2) **Shared training:** Training & testing → METbench + SPEC CPU2006
   - By using cross-validation
   - Higher accuracy expected

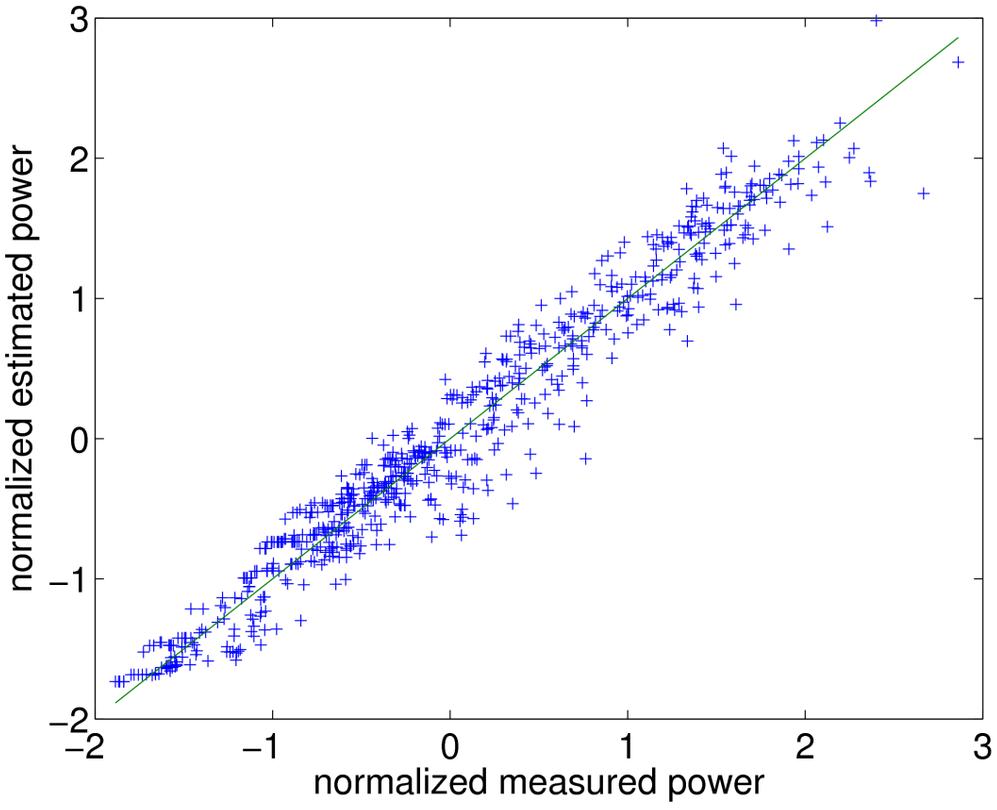
Results – Power Model (METbench training)



- METbench is used for the model training
  - The model is then tested on all the SPEC CPU2006
  - With several thread/core configurations
- Average error is below 4% for all cases
  - The maximum error is observed when the number of cores and hardware threads is highest
  - Similar error to other published works

Results – Power Model (Shared training)

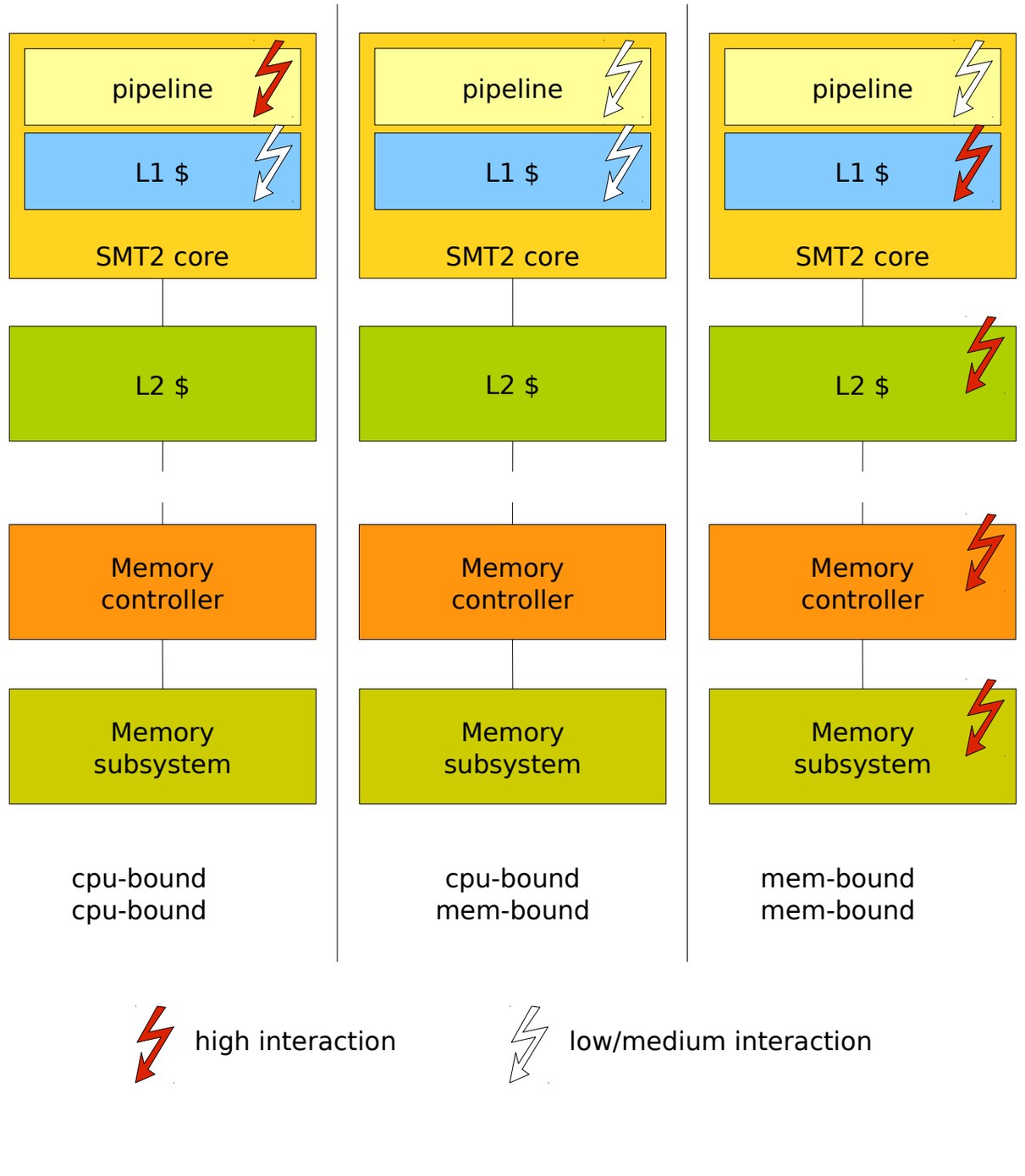
- Shared training
  - Both data from METbench and SPEC CPU2006 is used
  - Capture wider resource usage patterns
- Cross-validation is used test the model
- Accuracy is improved
  - Average error is less than 1.2%
  - Increased time for collecting the training data



Measured vs. estimated power consumption

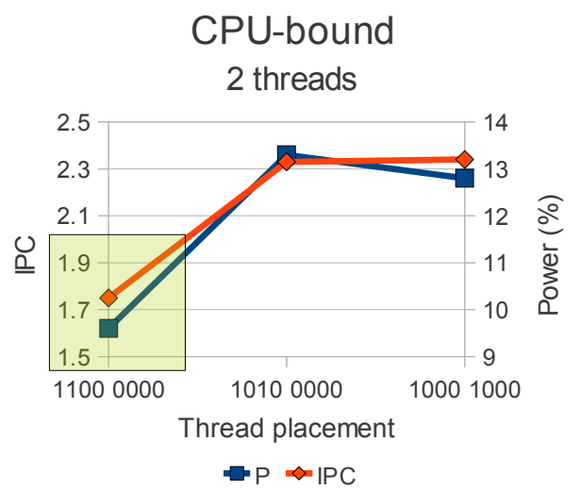
Results – Thread Placement

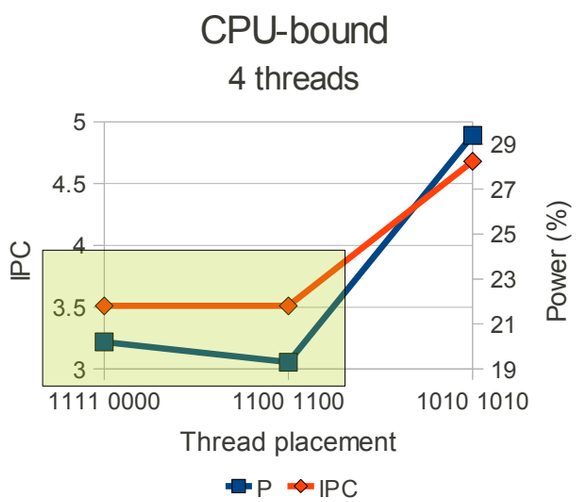
- Performance/power effect
  - Due to resource sharing
- Already considered in Linux
  - Spread tasks across domains
    - Increase performance
  - Reduce the number of domains
    - Power reduction
  - Not workload-aware
- Depending on workload characteristics
  - CPU-bound
  - Memory-bound

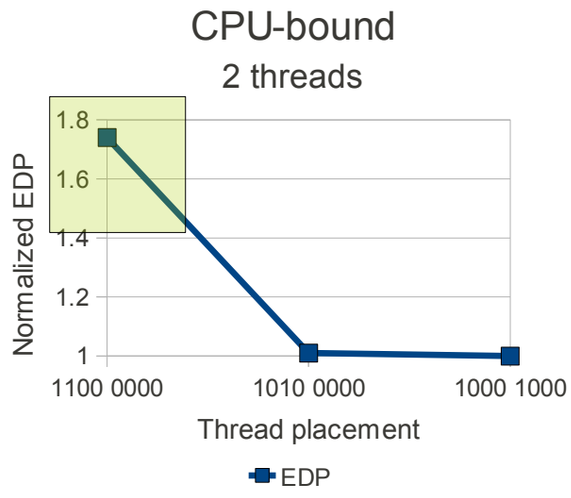


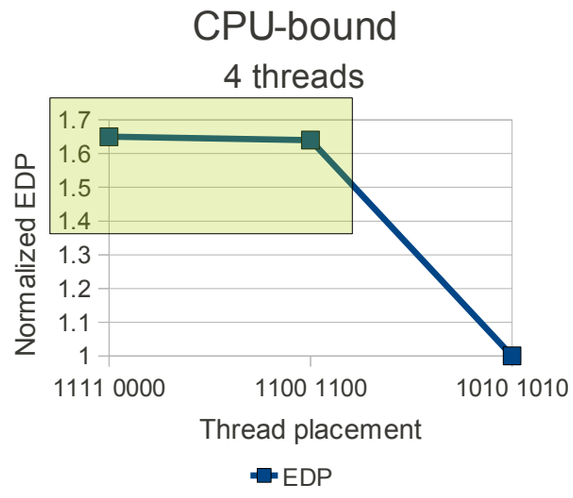
Results – Thread Placement (CPU-bound)

- CPU-bound workloads
  - Highly sensitive to intra-core resource sharing
    - IPC decreases 25%
    - EDP worsens up to 74% (2 threads case)
    - However, power consumption is lower
  - No difference at the inter-core level
    - Consolidating processes into a single chip does not offer any significant advantage
    - POWER6 saves power at the core level







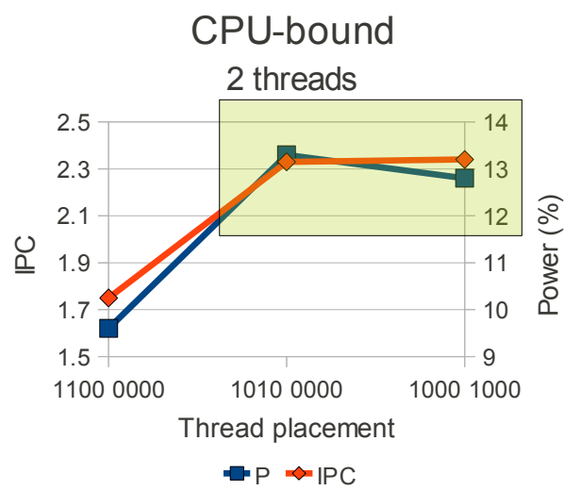


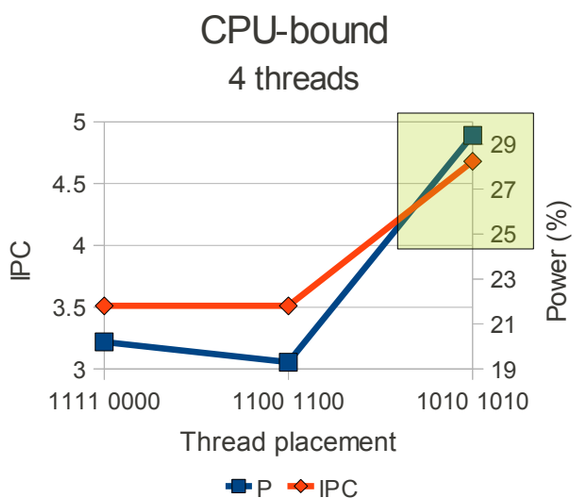
Placement reference:

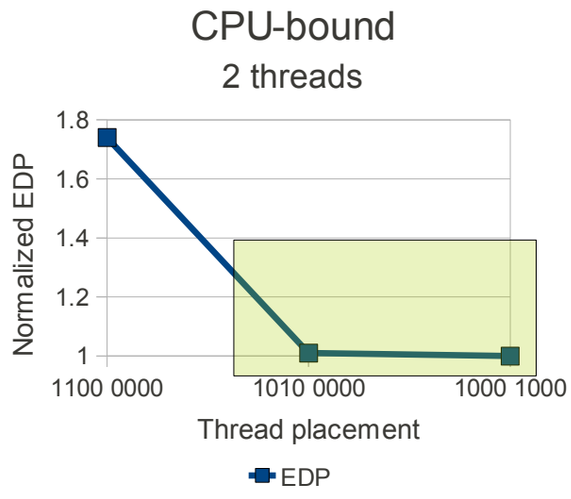
XX	XX	XX	XX
Core 1	Core 2	Core 3	Core 4
Chip 1		Chip 2	

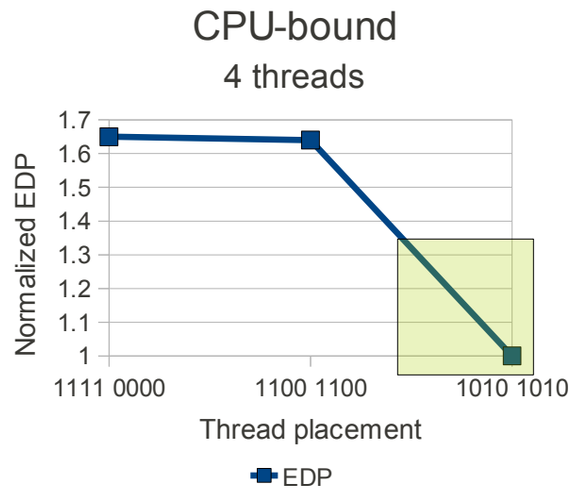
Results – Thread Placement (CPU-bound)

- CPU-bound workloads
  - Highly sensitive to intra-core resource sharing
    - IPC decreases 25%
    - EDP worsens up to 74% (2 threads case)
    - However, power consumption is lower
  - No difference at the inter-core level
    - Consolidating processes into a single chip does not offer any significant advantage
    - POWER6 saves power at the core level







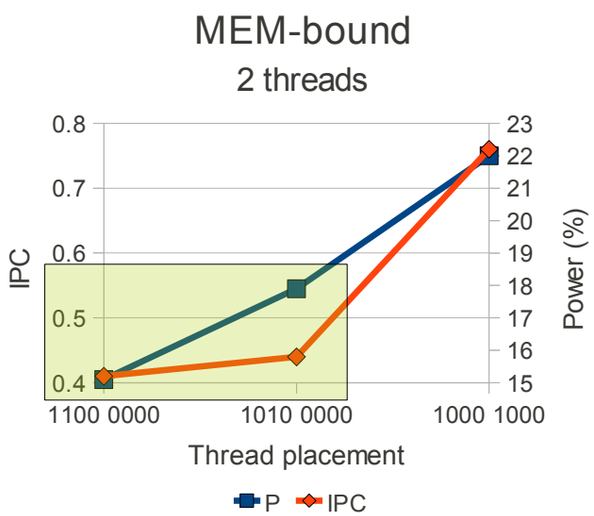


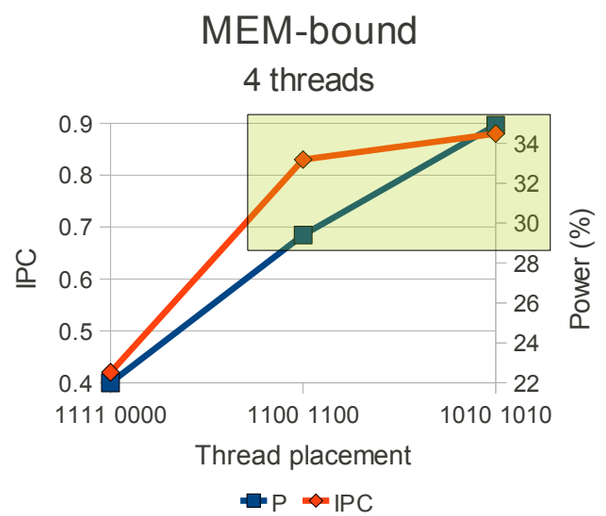
Placement reference:

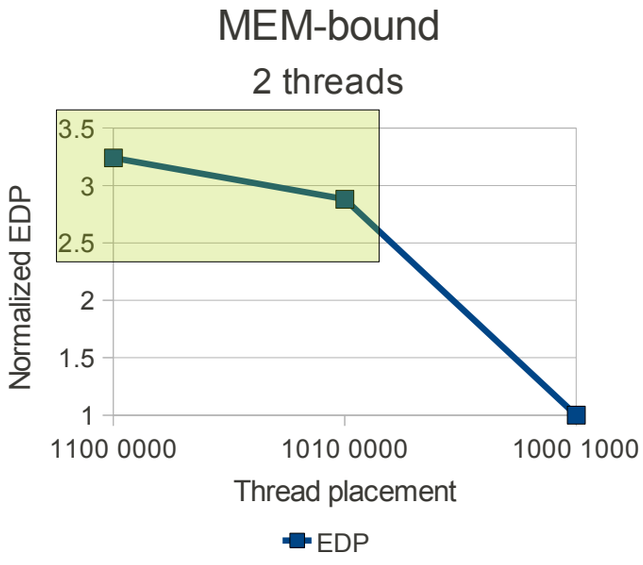
```
  XX      XX      XX      XX
Core 1  Core 2 | Core 3  Core 4
   Chip 1           Chip 2
```

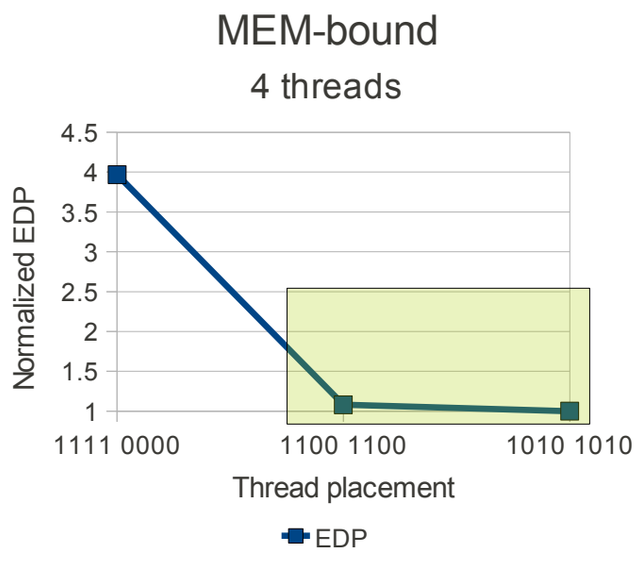
Results – Thread Placement (MEM-bound)

- MEM-bound workloads
  - Slightly sensitive to intra-core resource sharing
    - IPC only decreases 6-7%
    - EDP only worsens 13%
    - Lower power consumption
  - Significant difference at the inter-chip level
    - Both performance and EDP improves
    - Allows to better use the per-chip single memory controller
    - Up to a 2X IPC improvement
    - Up to a 4X EDP improvement



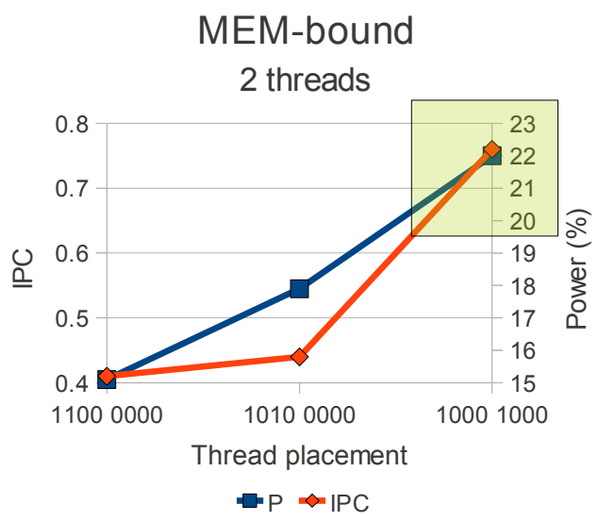


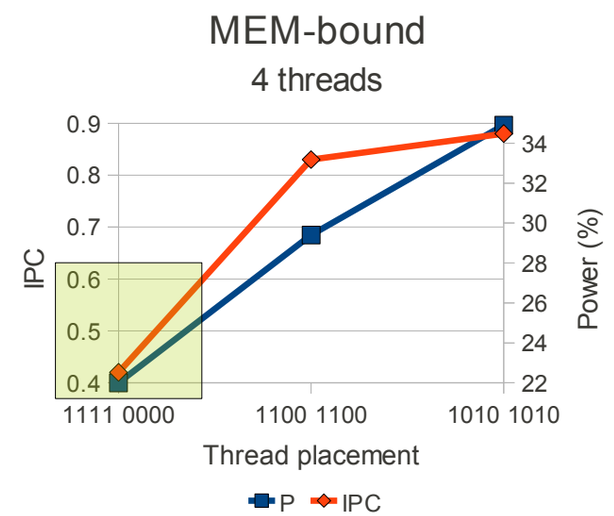


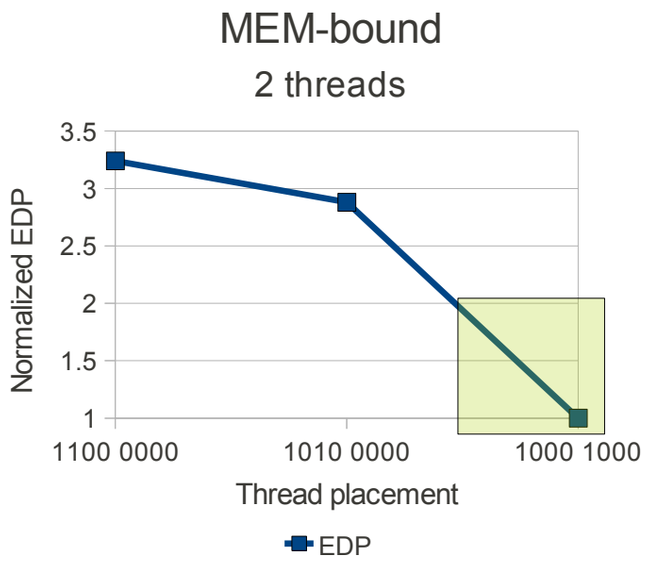


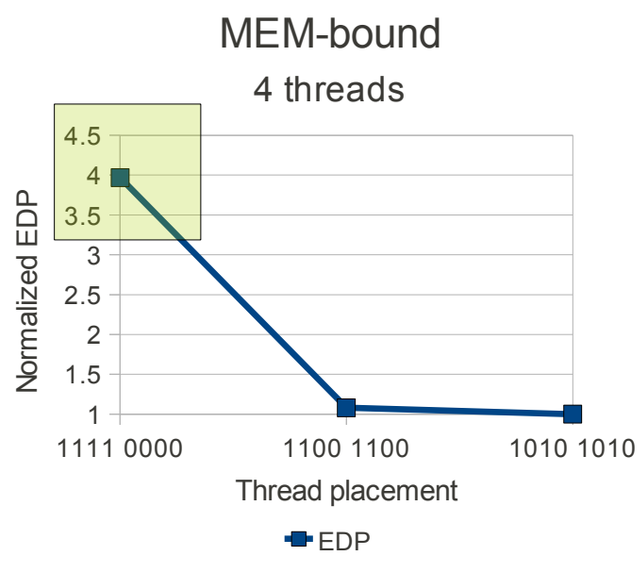
Results – Thread Placement (MEM-bound)

- MEM-bound workloads
  - Slightly sensitive to intra-core resource sharing
    - IPC only decreases 6-7%
    - EDP only worsens 13%
    - Lower power consumption
  - Significant difference at the inter-chip level
    - Both performance and EDP improves
    - Allows to better use the per-chip single memory controller
    - Up to a 2X IPC improvement
    - Up to a 4X EDP improvement









Results – Thread Placement (CPU-MEM-mix)

- CPU-MEM workload mix

  A) No significant effect

  - Flat change in performance and EDP

  B) Memory controller saturation

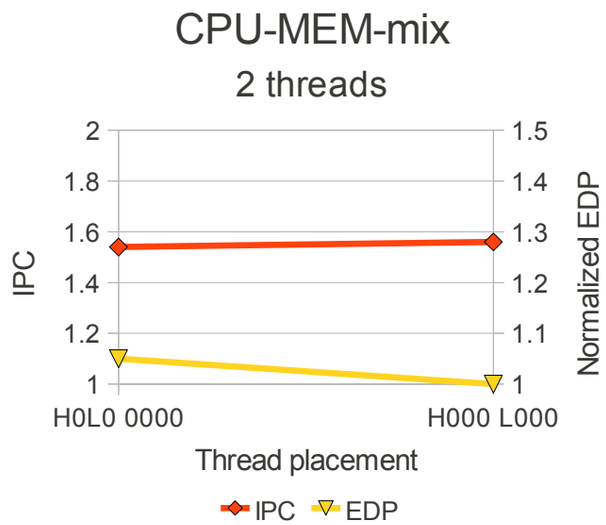
  - 20% improvement in IPC
  - 1.5X improvement in EDP

  C) Pipeline and memory controller saturation

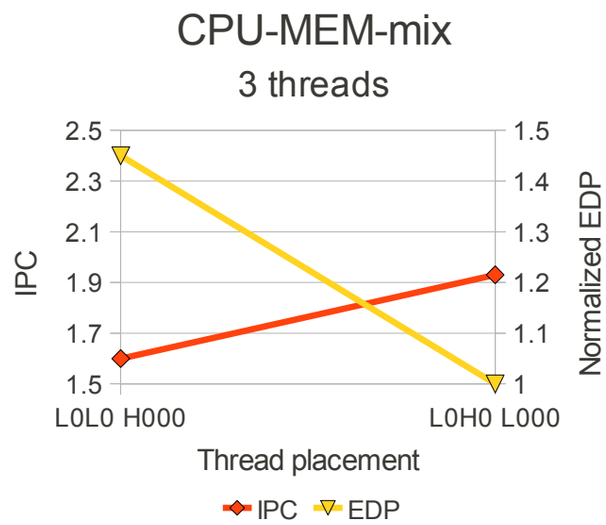
  - 10% improvement in IPC
  - 1.2X improvement in EDP

Placement reference:

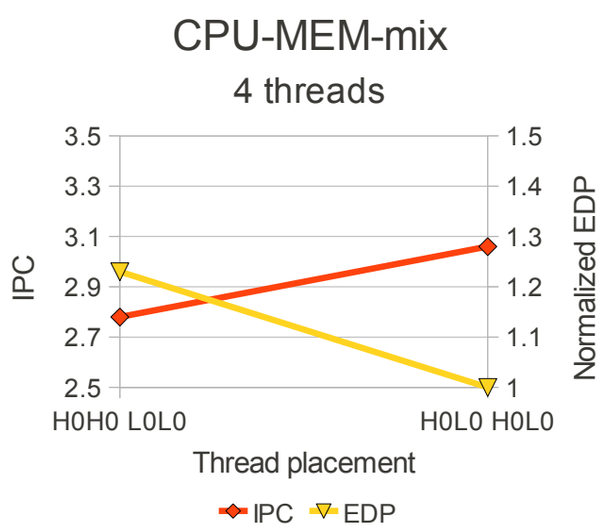
H: CPU-bound workload (h264ref)
L: MEM-bound workload (lbm)



A)



B)



C)

Results – Thread Placement

- Thread placement can significantly affect performance, power and EDP
- A workload-aware task scheduler
  - Increase system performance
  - Reduce power/energy consumption

Conclusions

- We presented a power and thermal characterization for a POWER6-based system
  - Both when the system is idle and active
  - Multiple-level characterization
    - HW, OS and application
- Results when idle
  - Nap mode + hardware thread priorities reduce power and temperature by a 25%
  - Linux tickless kernel does not significantly affect power consumption for POWER6
- Results when active
  - Compute-intensity is the most relevant factor determining core temperature
  - Memory-intensity is the main factor related to system power consumption
    - Power consumption is also affected by high-IPC benchmarks

Conclusions

- We provide a system power consumption model
  - Based on performance counters
  - Its prediction error is between 4% and 1.2%
    - Depending on training data
- We study the effect of thread placement
  - Thread placement affects performance and power/energy consumption
  - Significants benefits are possible with a workload-aware scheduler
    - Up to 2X IPC improvement
    - Up to 4X EDP improvement

Power and Thermal Characterization of POWER6 System

Víctor Jiménez[†], Carlos Boneti[*], Francisco J. Cazorla[†], Roberto Gioiosa[†], Eren Kursun[‡], Chen-Yong Cher[‡], Canturk Isci[‡], Alper Buyuktosunoglu[‡], Pradip Bose[‡], Mateo Valero[†]

[†] Barcelona Supercomputing Center, Barcelona (Spain)

[*] Schlumberger BRGC, Rio de Janeiro (Brazil)

[‡] IBM T.J. Watson Research Center, Yorktown Heights (USA)

September 13th, 2010

Vienna (Austria)

Barcelona Supercomputing Center
Centro Nacional de Supercomputación