The Benefit of SMT in the Multi-Core Era

Flexibility towards Degrees of Thread-Level Parallelism

Stijn Eyerman, Lieven Eeckhout
Ghent University, Belgium

Motivation

Environments with varying thread-level parallelism

- Multi-program workloads
- Desktop applications (2–3 active threads)
- Server workloads (10–50 % utilization)
- Multi-threaded applications

"How to best design a single-ISA multi-core processor in light of varying degrees of thread-level parallelism in contemporary workloads?"

Number of active threads on 24-core CPU (PARSEC, ROI-only)
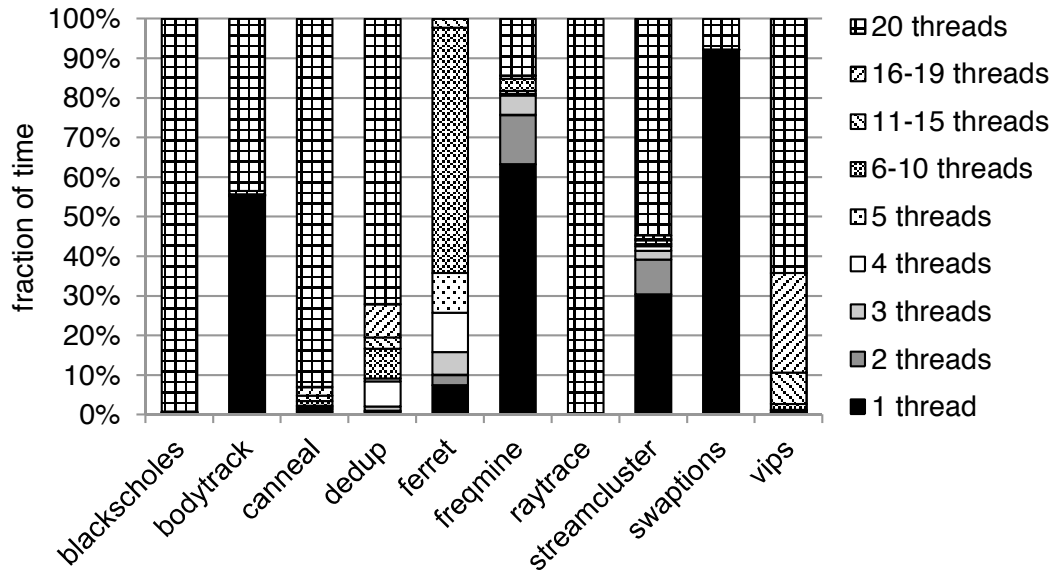
Core configurations

|	Big core	Medium core	Small core
Frequency	2.66GHz	2.66GHz	2.66GHz
Type	Out-of-Order	Out-of-Order	In-Order
Width	4	2	2
ROB size	128	32	N/A
Func. units	3 int, 2 ld/st	2 int, 1 ld/st	2 int, 1 ld/st
	1 mul/div	1 mul/div	1 mul/div
	1 FP	1 FP	1 FP
SMT contexts	up to 6	up to 3	up to 2
L1 I-cache	32KB	16KB	6KB
	4-way assoc	2-way assoc	2-way assoc
L1 D-cache	32KB	16KB	6KB
	4-way assoc	2-way assoc	2-way assoc
L2 cache	256KB	128KB	48KB
	8-way assoc	4-way assoc	4-way assoc
Last-level cache	8MB, 16-way assoc		
On-chip interconn.	2.66GHz, full cross-bar		
DRAM	8 banks, 45ns access time		
Off-chip bus	8GB/s		

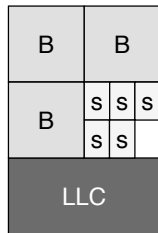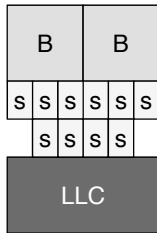# Power-equivalent multi-core designs (46–50 W)
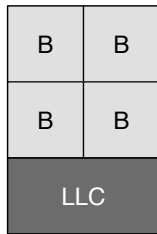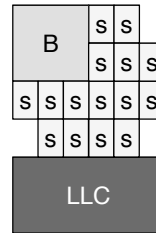

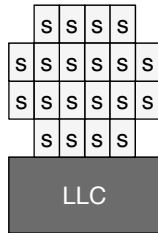4B
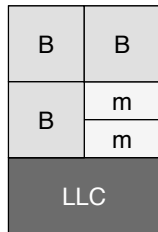
3B2m
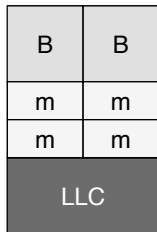
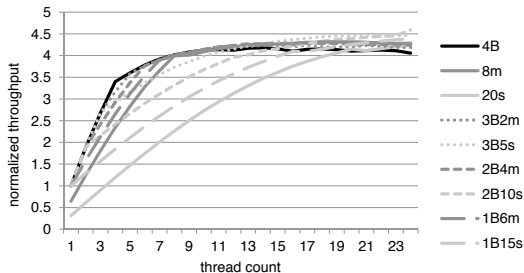2B4m
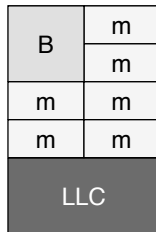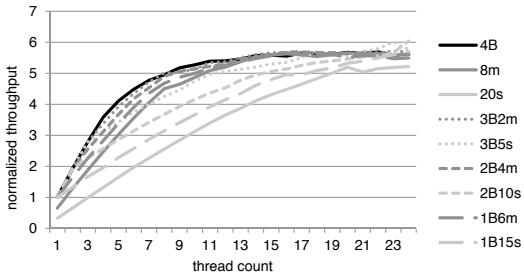
1B6m
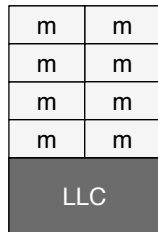
8m

3B5s

2B10s

1B15s

20s

Multi-program workloads (SPEC CPU 2006)

Homogenous workloads

Heterogenous workloads

Findings

1. A homogeneous multi-core consisting of all big SMT cores yields better performance than a heterogeneous multi-core for a small number of threads and only slightly worse for a large number of threads.
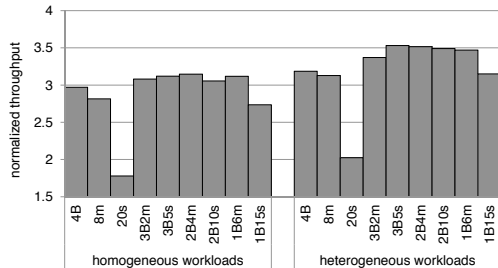
Uniform thread count distribution
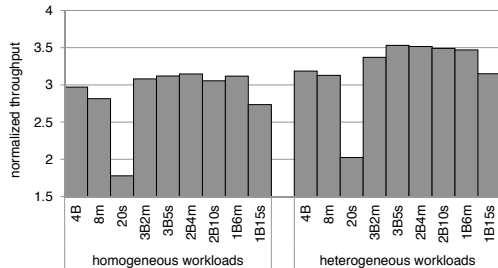
Uniform thread count distribution

Uniform thread count distribution

Findings (continued)
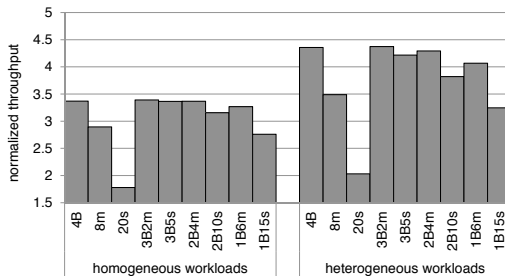
2. In the absence of SMT, heterogeneous multi-cores outperform homogeneous multi-cores across varying thread counts.

3. A homogeneous multi-core with big SMT cores outperforms a heterogeneous multi-core (without SMT) under the same power budget. Put differently, SMT outperforms heterogeneity as a means to cope with varying thread counts.

4. The added benefit of combining heterogeneity and SMT is limited.

5. Adding SMT to the heterogeneous designs makes the optimum shift towards fewer and larger cores.

# Datacenter distribution distribution, heterogeneous workloads
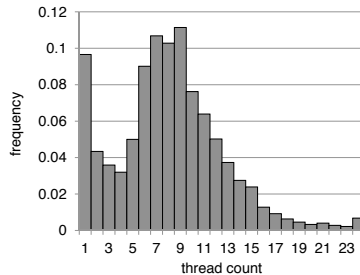
# Datacenter distribution distribution, heterogeneous workloads
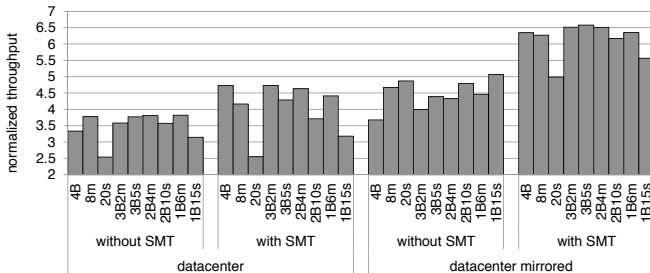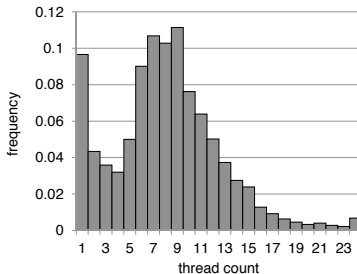
6. For distributions that are skewed to fewer threads, the 4B configuration with SMT is optimal. For distributions that are skewed towards more active threads, 4B with SMT becomes less optimal, but its performance is very close to the optimum.

Normalized speedup for PARSEC benchmarks
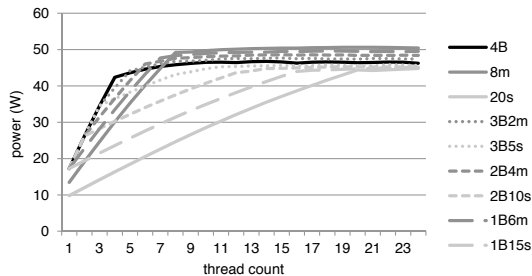

ROI-only
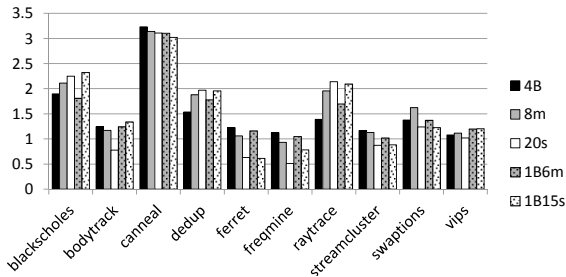
Whole program
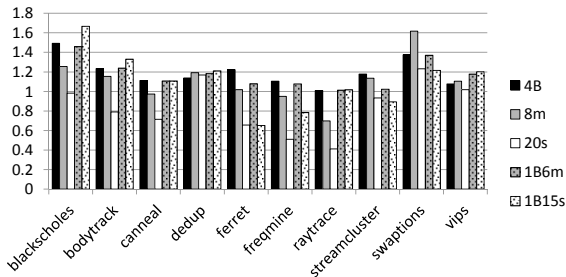
7. SMT is also beneficial for multi-threaded workloads. As for the multi-program workloads, adding SMT lets the optimal design shift to fewer but larger cores. A homogeneous design with big SMT cores outperforms the best heterogeneous design without SMT, and performs close to, and sometimes even slightly better than, the best heterogeneous design with SMT.
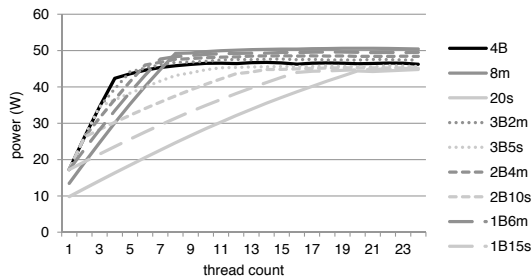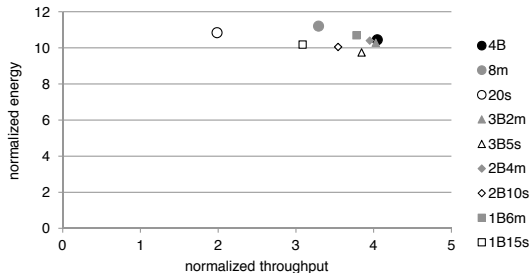
Energy Efficiency

Power consumption
assuming power gating

Power consumption
assuming power gating

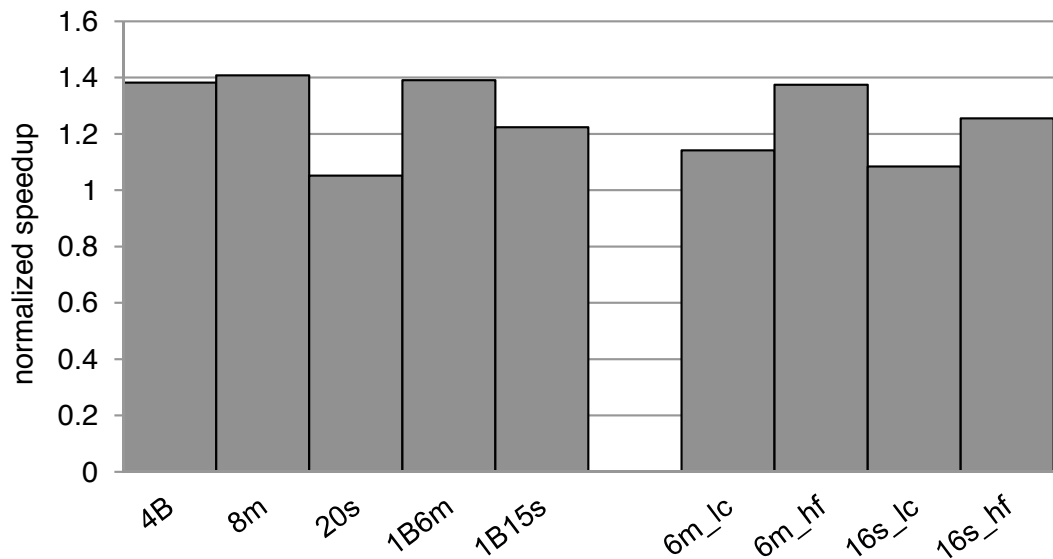
Energy vs. performance

9. Heterogeneous multi-core designs, when power gating idle cores, yield an (only) slightly better energy-efficiency compared to homogeneous multi-cores with big SMT cores under variable active thread count conditions.

Average PARSEC performance with large-cache/higher frequency

Findings (continued)

10. Enlarging the caches or increasing the frequency of the medium and small cores does not affect the general observation that a homogeneous multi-core with big SMT cores is close to optimal.

- Number of active threads usually varies over time
- Homogeneous big SMT cores provide adaptivity
  - high per-thread performce for few threads
  - competitive throughput for higher thread numbers
  - flexible use of private caches
- Heterogeneous multi-core slightly more energy-efficient

  "[W]hile multi-cores with many small cores, be it homogeneous or heterogeneous architectures, outperform homogeneous multi-cores with big SMT cores at full utilization, the inverse is typically true under variable active thread workload conditions[...]"