Dataflow & Tiled Architectures

WaveScalar and TRIPS

- *Irene Lin & Kevin Rohan*

WaveScalar

- Motivation
- Solution & Implementation
- Results
- Conclusion & Discussion

Motivation

- Scaling up superscalar is hard
  - Circuit complexity, fast transistors, slow wires, communication infrastructure
- Von Neumann means sequential
  - Sequential fetch (PC) and memory
- Untapped dataflow locality
  - predictability in the dynamic data dependencies

Solution & Implementation

- Chunk CFG into waves
- Every data value carries a tag, every wave has a wave number.
- Total ordering of memory operations
  - From wave number and memory instruction sequence number
- Increment wave number with WAVE-ADVANCE instruction
- Conditional split to steer data value to destination
  - Converting control dependencies into data dependencies

WaveCache

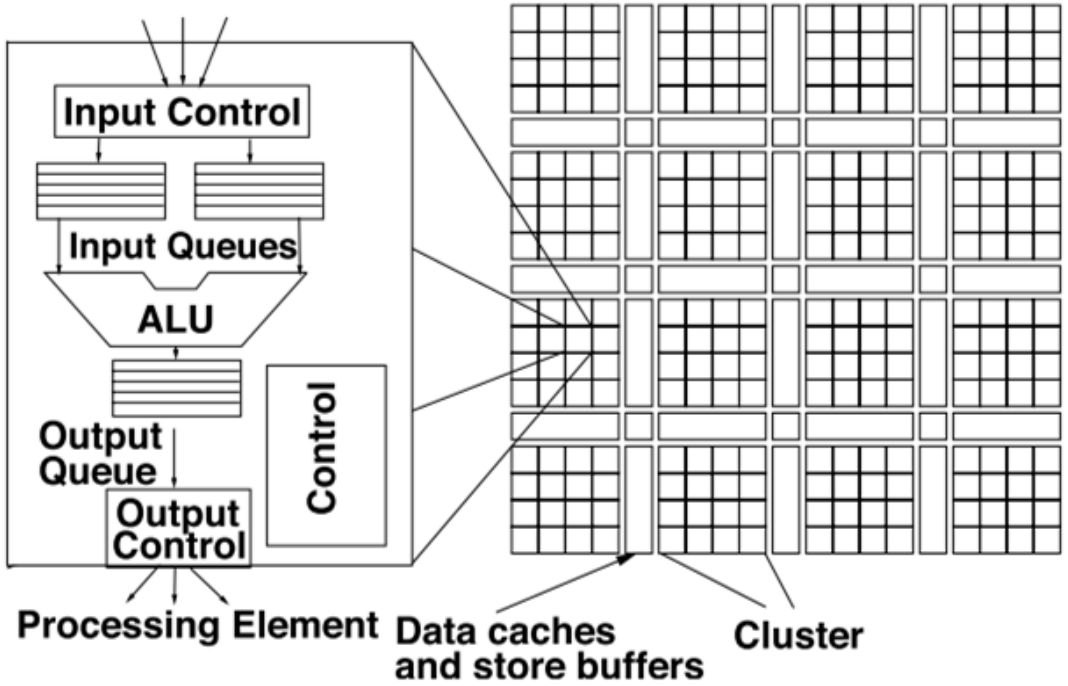
Grid of processing elements for:

- control instruction placement
- input output queues
- communication logic
- functional unit

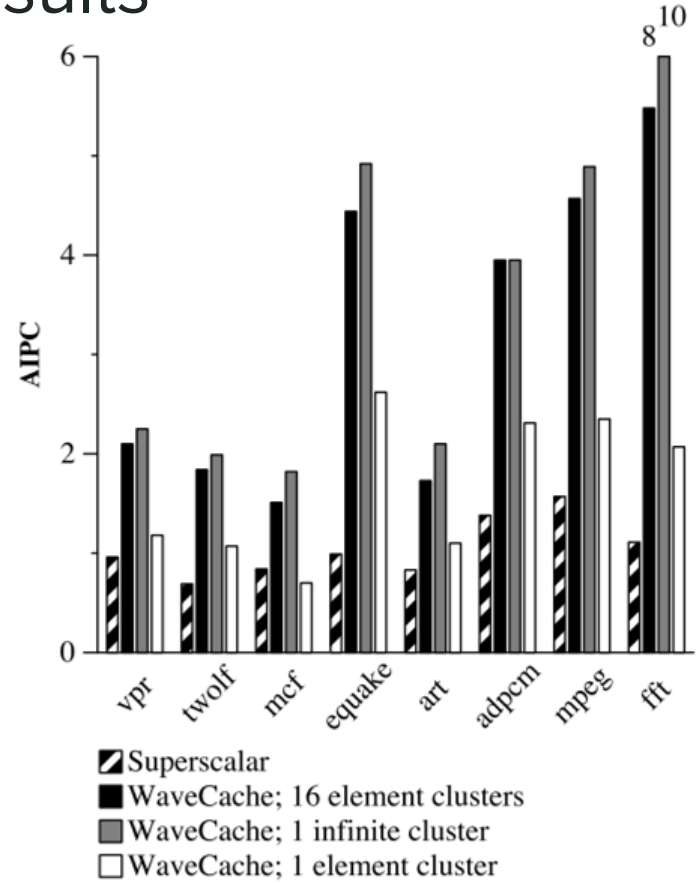
Cluster of 4 PEs:

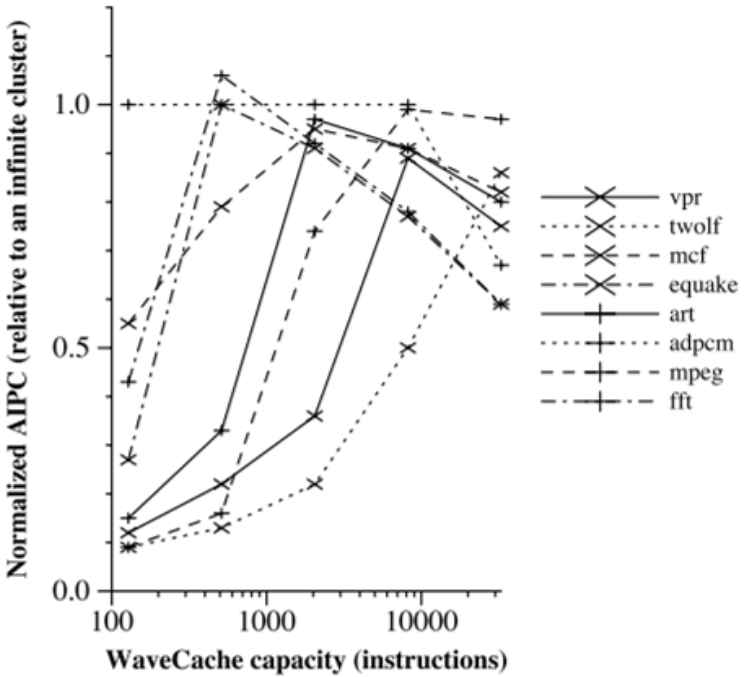
- L1 cache
- Store buffer

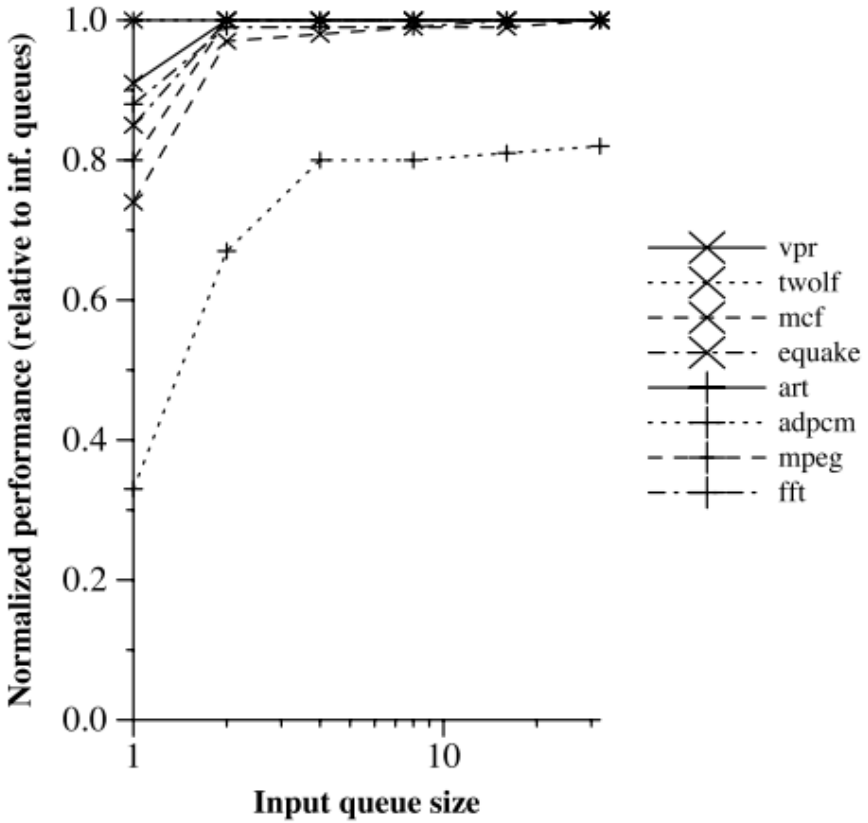
processor fetch is data-driven

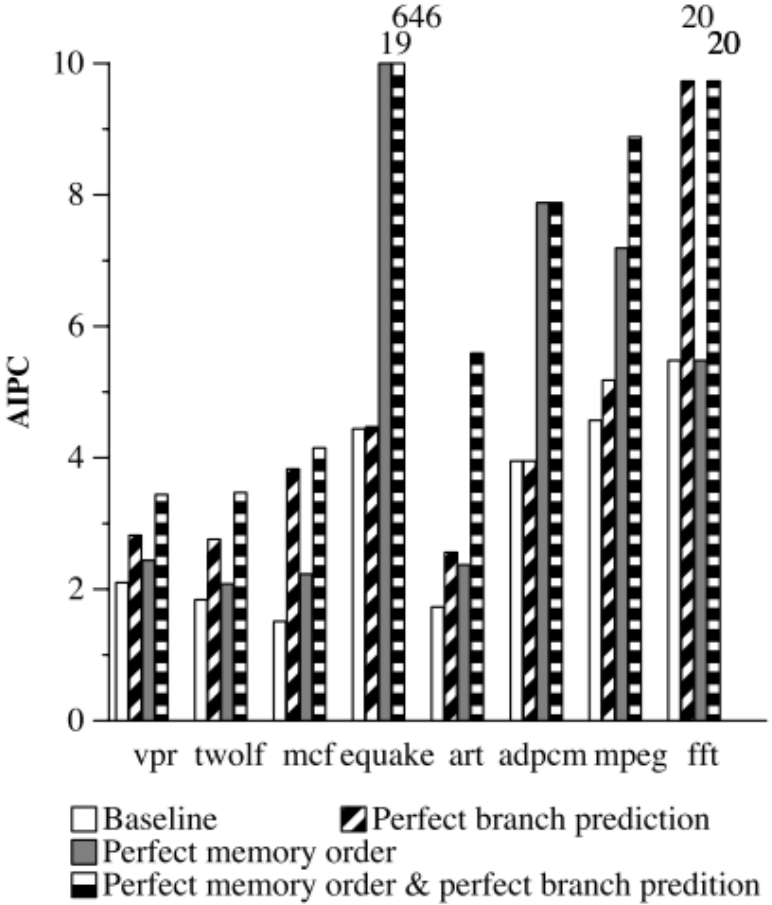


Results


AIPC
0
2
4
6
8
10
vpr
twolf
mcf
equake
art
adpcm
mpeg
fft
Superscalar
WaveCache; 16 element clusters
WaveCache; 1 infinite cluster
WaveCache; 1 element cluster


Normalized AIPC (relative to an infinite cluster)
0.0
0.5
1.0
100
1000
10000
WaveCache capacity (instructions)
vpr
twolf
mcf
equake
art
adpcm
mpeg
fft


Normalized performance (relative to inf. queues)
1.0
0.8
0.6
0.4
0.2
0.0
1
10
Input queue size
vpr
twolf
mcf
equake
art
adpcm
mpeg
fft


AIPC
646
19
20
20
10
8
6
4
2
0
vpr
twolf
mcf
equake
art
adpcm
mpeg
fft
Baseline
Perfect branch prediction
Perfect memory order
Perfect memory order & perfect branch predition

Conclusion & Discussion

- “Scalable, low complexity, high performance”
  - Is it actually scalable?? Compiler Scalability??
- Dataflow driven rather than von neumann style linearity
  - Increase in parallelism
- Binaries are bigger
  - Maybe not relevant in the present scenario
- Miss is a heavyweight event
- Compiler vs programmer responsibility
  - Superscalar and WaveScalar both split up the program into blocks
  - Multi-core systems - programmer's responsibility

TRIPS

- Motivation
- Solution & Implementation
- Results
- Conclusion
- Discussion

Motivation

- Pretty much the same as before
  - Scalability, reduce the circuit complexity etc.
  - Reduce burden on programmer
- Does not replace Von Neumann architecture

Solution & Implementation

EDGE ISA

- Block atomic execution
- Direct instruction communication within a block

Micro Arch

- Operand Network (bypass reg file and memory)
- Tiles (global control, execution, register, data, instruction)

Compiler

- Conventional optimizations
- Translate code to TRIPS intermediate language and make TRIPS blocks
- Translate blocks into assembly

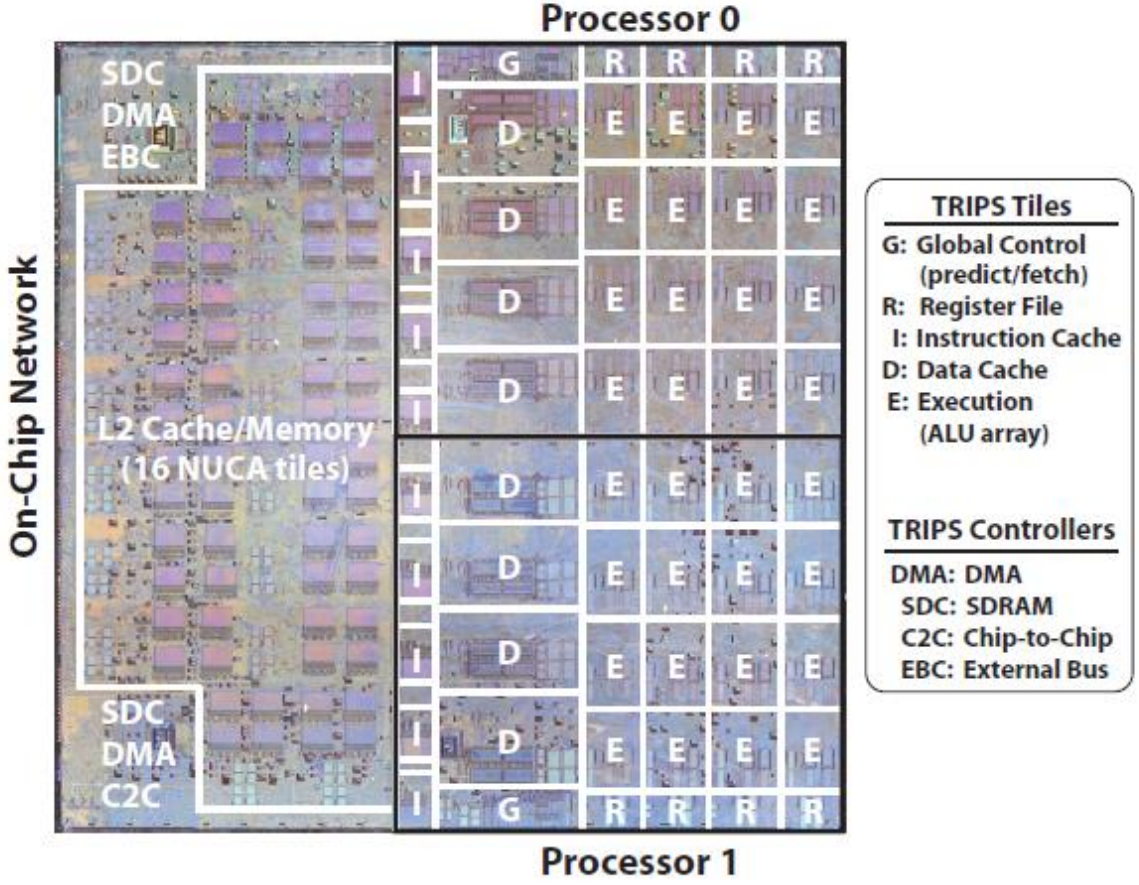


Figure 2. TRIPS die photo with tile overlays.

“Need as many as 2–4 times more instructions than the Alpha, due to aggressive predication.”

Predication works by executing instructions from both paths of the branch and only permitting those instructions from the taken path to modify architectural state.

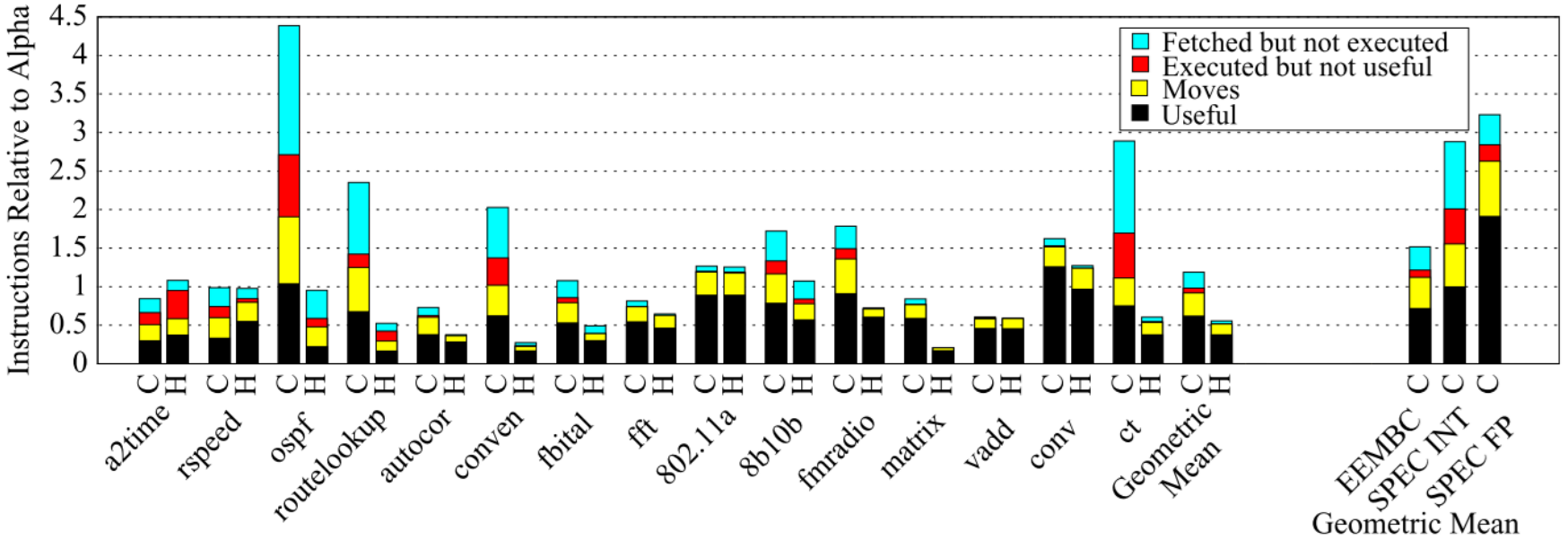


Figure 4. TRIPS instructions normalized to Alpha for compiled (C) and hand-optimized (H) benchmarks.

Memory Instruction ⇒ Register Instructions

Register Instructions ⇒ Direct Communication

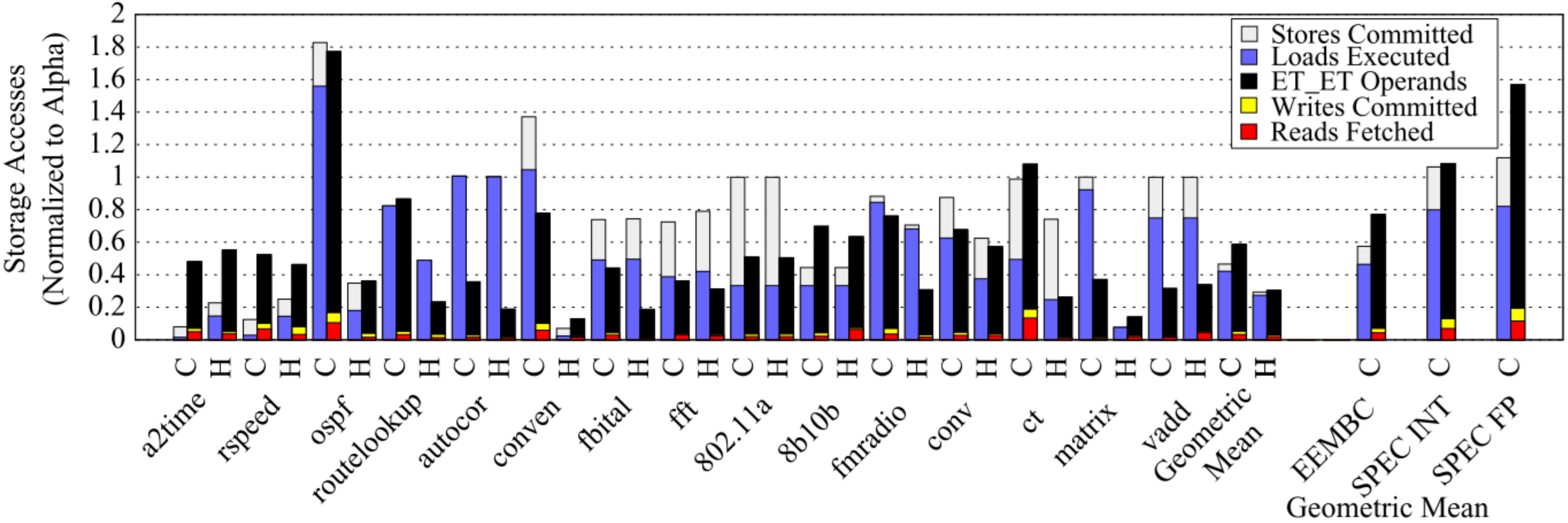


Figure 5. Storage accesses normalized to Alpha for compiled (C) and hand-optimized (H) benchmarks.

Microarchitecture Evaluation : ILP Evaluation

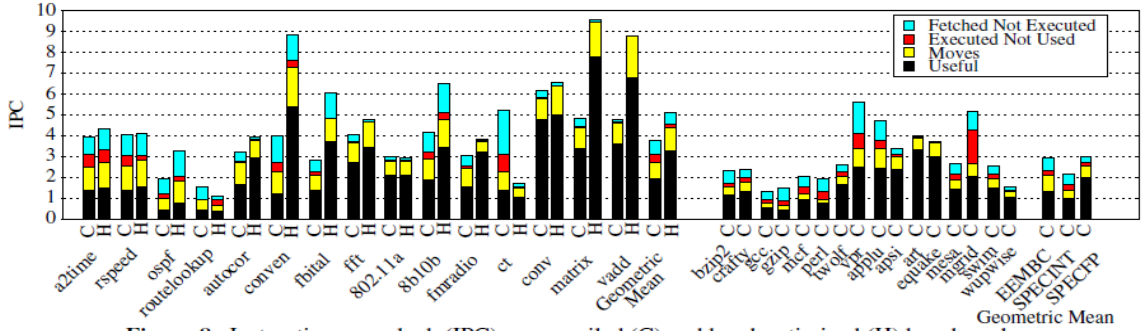


Figure 8. Instructions per clock (IPC) on compiled (C) and hand-optimized (H) benchmarks.

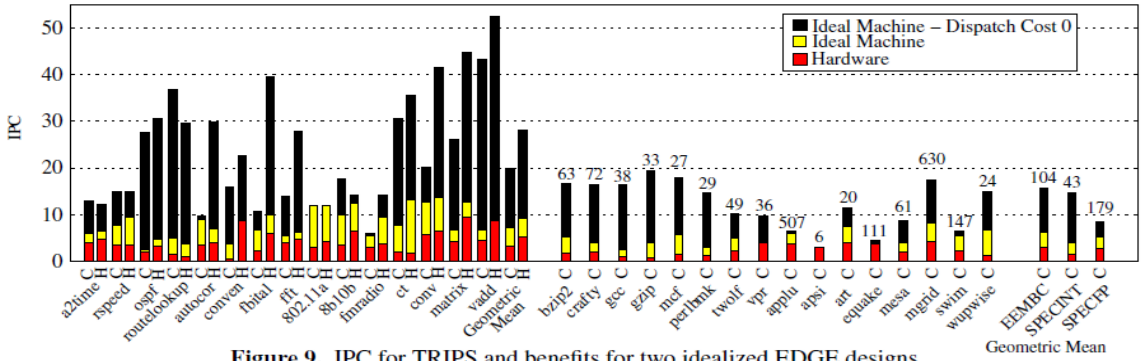


Figure 9. IPC for TRIPS and benefits for two idealized EDGE designs.

Microarchitecture Evaluation : v/s Commercial

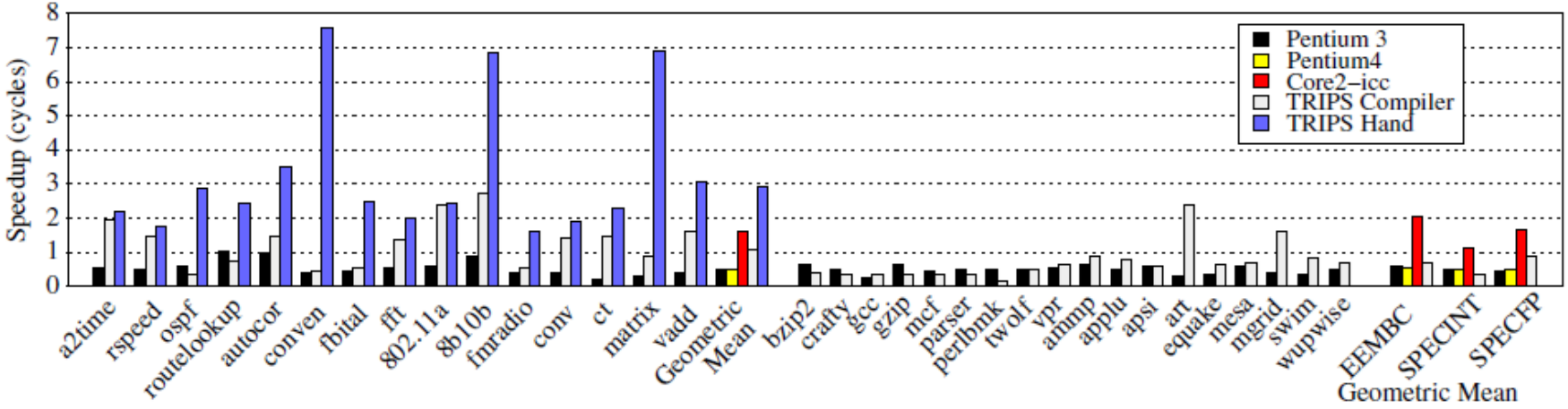


Figure 10. Speedup of TRIPS relative to the Core 2 on simple, EEMBC, and SPEC benchmarks.

Conclusion

“the performance and potential energy efficiency of EDGE designs may be sufficiently large to justify adoption in mobile systems or data centers, where high performance at low power is essential.”

Polymorphic processor - every task can run on every unit

Discussion

WaveScalar vs TRIPS:

- Data-flow type of execution in both
- Inter-block communication:
  - TRIPS - register file
  - WaveScalar - WaveCache
- TRIPS is a Von Neumann architecture
- WaveScalar has 2.5 times more speedup as more parallelism