The Relative Impact of Memory Latency, Bandwidth, and Branch Limits to Microprocessor Performance

Parthasarathy Ranganathan
Rice University

with

Norman Jouppi
DEC Western Research Lab

Motivation

Processor performance increasing
60% a year (100X per decade!)

- 1986: 16MHz single-issue in-order R2000
- 1997: 500MHz quad-issue out-of-order EV6
- 2008: 4GHz 16-issue 8-thread 4-processor SMP?

What are limits to performance?

Limits studies

- Points out potential bottlenecks, given current trends
- Realistic and balanced assumptions crucial

Possible Limits

- Memory Latency *(Memory latency wall)*
  Future processors $\Rightarrow$ longer latencies
- Memory Bandwidth *(Memory bandwidth wall)*
  Future processors $\Rightarrow$ more bandwidth
- Branch Predictability *(Branch wall)*
  Future processors $\Rightarrow$ wider issue, deeper pipelines
- Other limits
  - Design complexity
  - Fabrication process
  - Economics

Results

- The branch wall is more significant than either memory wall
- Memory bandwidth solvable by good engineering and money
- For many applications, memory latencies of 100 cycles or more are tolerable
- Future improvement in branch prediction may be limited

Overview

⇒ Background
  - Memory and branch wall
  - Technology trends
- Evaluation methodology
- Results
- Conclusions

The Memory Wall [WulfMcKee1996]

Increasing difference between processor and memory speeds

Memory wall

- $$ExecutionTime = \sum nonMemInstr * AvgNonMemExecTime \\ + \sum memInstr * AvgMemAccessTime$$
- Assertion: No program can execute faster than this

BUT

- Above assumes non-pipelined memory
- Bandwidth set to (Bus Width/Latency)

The “Memory Speedbump”

Systems use pipelined memory

- Even PC’s have it now
- Requires exploitation of large ILP for deep memory pipelines
- Memory latency and bandwidth largely independent limits
  - “Memory Latency Wall”
  - “Memory Bandwidth Wall”

The "Branch Wall"

- Branches lead to loss of instruction-level parallelism
- Problem exacerbated with wider issue and deeper pipelines
- Solution: *Speculate!* (Branch prediction)
  - Bimodal branch predictors
  - Global branch predictors
  - Local branch predictors
  - Hybrid branch predictors

  Branch prediction increasing in aggressiveness

Branch prediction strategies (1 of 2)

- Bimodal branch predictor – two bit counters
- Global branch predictor – use correlation between different branches

```
if(x<1){                ← Branch 1
  Do something
}

...

if(x>1){                ← Branch 2 correlated to branch 1
  Do something else
}
```

Branch prediction strategies (2 of 2)

- Local branch predictor – use correlation between executions of same branch

```
for(j=0;j<100;j++){      ← Branch 1

  for(i=0;i<3;i++){      ← Branch 2 pattern repeats!

    ...
  }
}
```

- Hybrid branch predictors
  - different predictors perform best in different circumstances
  - dynamically choose between different predictors

Technology Trends - Semiconductor

- History
  - Transistors on a microprocessor: +40% per year
  - Clock frequency: +30% per year
  - Microprocessor performance: +60% per year

$\Rightarrow$ Microprocessor performance improved 100X per decade

- On-chip memory has increased by $>$ 50X over last decade

- SIA roadmap projects current scaling rates next 15 years
- On-chip memory should increase by $>$50X over next decade

Technology Trends - Interconnect

- History
  - Device pincount has been increasing at only 16% per year
  - Ceramic PGA pins present large impedance discontinuity
  - Most PC's have bus frequencies stuck at <= 66 MHz (1/4 CPU)
  - Bus widths have been narrow and bidirectional (P5 and P6 8B)

Is current growth in packaging and interface technology good enough?

Technology Trends - Interconnect

- But
  - Rambus has shown interchip signaling at 2X processor clock
  - Communication circuits can approach 50GHz today
  - Recently work begun on 4GHz differential signaling in 0.5um CMOS
  - Vendors have announced ball grid arrays with 1500 bumps (for I/O connections)
  - Small bumps have been used in microwave circuits up to 70GHz

Simulation Framework

- Trace-driven simulation
- ATOM to instrument executable
- Processor model
  - Unlimited resources except for memory and branches
  - Register renaming to unlimited number of registers
  - Single cycle functional units
  - 16536 entry instruction window
  - Issue width up to 16536 per cycle
  - Perfect jump prediction
  - All memory references checked at run time for aliases
- Goal is to include benefits of large-scale code motions
- Focus is on upper bounds (limits study)

Large Scale Code Motions

- Loop interchange
- Loop fusion

  Example:

```
for(i=0;i<10000;i++)
        work1(i);

for(i=0;i<10000;i++)
        work2(i);
```

```
for(i=0;i<10000;i++)
        work1(i);
        work2(i);
```

Compiler optimizations enable large-scale code motions

Memory Models and Benchmarks

Ideally

Simulate benchmarks of 2010 on memory systems of 2010

Problems

Benchmarks of 2010 not available yet!

Memory systems of 2010 $\approx$ 10MB on-chip memory

Large application/large cache simulation time infeasible

Solution: Use scaled memory systems and benchmarks

(SPEC92 and small caches)

Applications

- SPEC92 benchmarks

Application	Input	Load %	Branch %
Integer Applications			
Compress	ref	26.2	13.34
Espresso	bca	22.5	17.81
Eqntott	ref	12.9	10.91
Floating point applications			
Doduc	short	31.5	6.07
Tomcatv	ref	29.1	3.32
Mdljsp2	short	29.5	8.59
Swm256	short	36.0	1.85
Hydro2d	short	21.7	7.83
Fpppp	short	34.9	1.21

Memory Models

- Perfect instruction cache
- Single-level data cache
  - 4-way set associative, 64KB
  - Chosen to approximate future demanding applications on *on-chip* 10 MB caches
  - Types modeled
    * Perfect memory system
    * Non-pipelined memory system
    * Pipelined memory system
  - Load latency and bandwidth varied
  - Store bandwidth and latency assumed not critical

Branch models

- Prediction strategies of increasing aggressiveness

Branch predictor	Notation	Size
bimodal (2c)	2c(1K)	2Kbits
gshare (Gsh)	gshare(12,12)	8Kbits
bimodal/gshare (2c/g)	2c(1K)/gshare(12,12)/sel(4K)	18Kbits
local/gshare (pa/g)	PA(4K,12,1)/gshare(12,12)/sel(4K)	72Kbits
big local/gshare (PA/G)	PA(32K,15,1)/gshare(15,15)/sel(32K)	672Kbits
Futuristic (Futr)	25% PA/G, 75% perfect	NA
Perfect (Perf)	Oracle-based	NA

- Issue stalled on misprediction
- Recovery from misprediction – one cycle penalty
- No issue restrictions on conditional branches

Overview

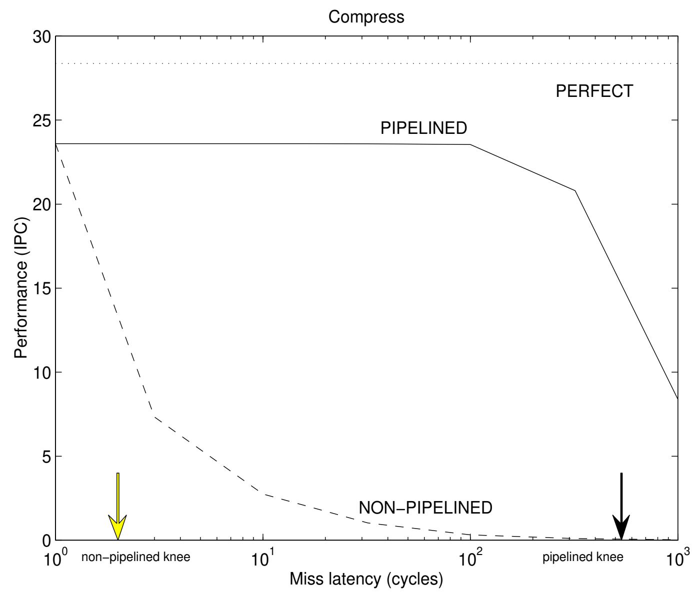
- Background
- Evaluation methodology

⇒ Results

  - Performance vs. Memory Latency
  - Performance vs. Memory Bandwidth
  - Performance vs. Branch Prediction
  - Performance vs. Latency, Bandwidth, Prediction

- Conclusions

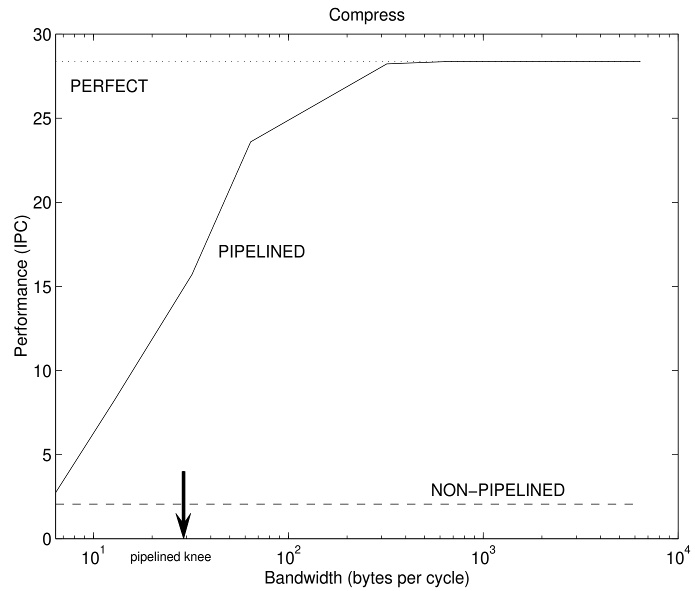
Performance vs. Memory Latency – Compress



(perfect branch, bandwidth 64 bytes per cycle)

- Low knees for non-pipelined system
- High knees for pipelined system

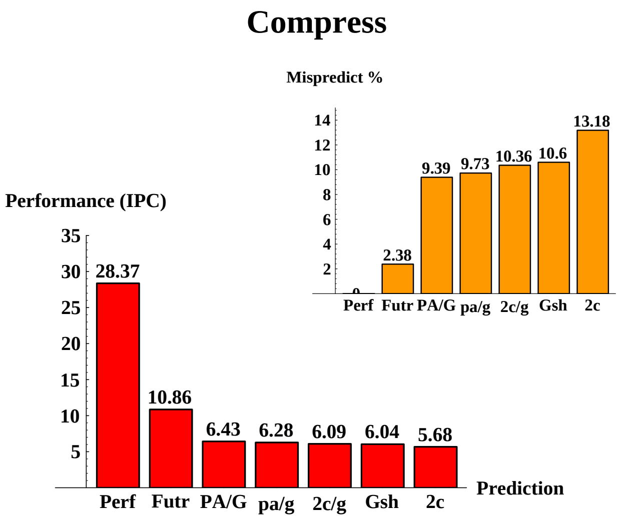
Performance vs. Memory Bandwidth – Compress



(perfect branch, 16 cycle miss latency)

- Most applications have low knee values
- Systems targeting IPC of 100 need <100 bytes per cycle

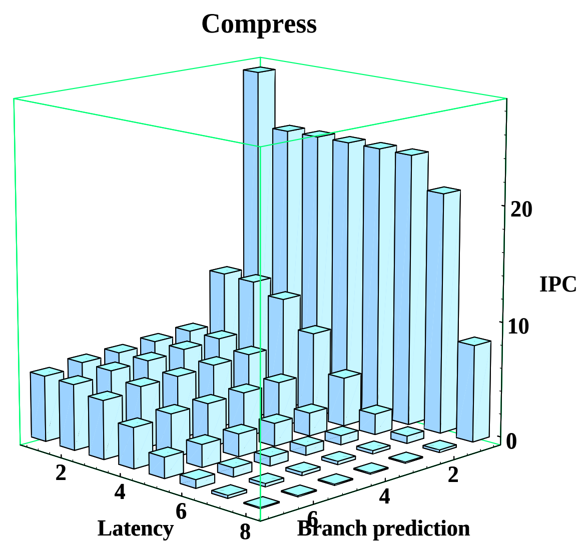
Performance vs. Branch Predictor – Compress



(perfect memory system)

- Significant loss in performance due to branch mispredictions.
- Even futuristic model does not get close to perfect.

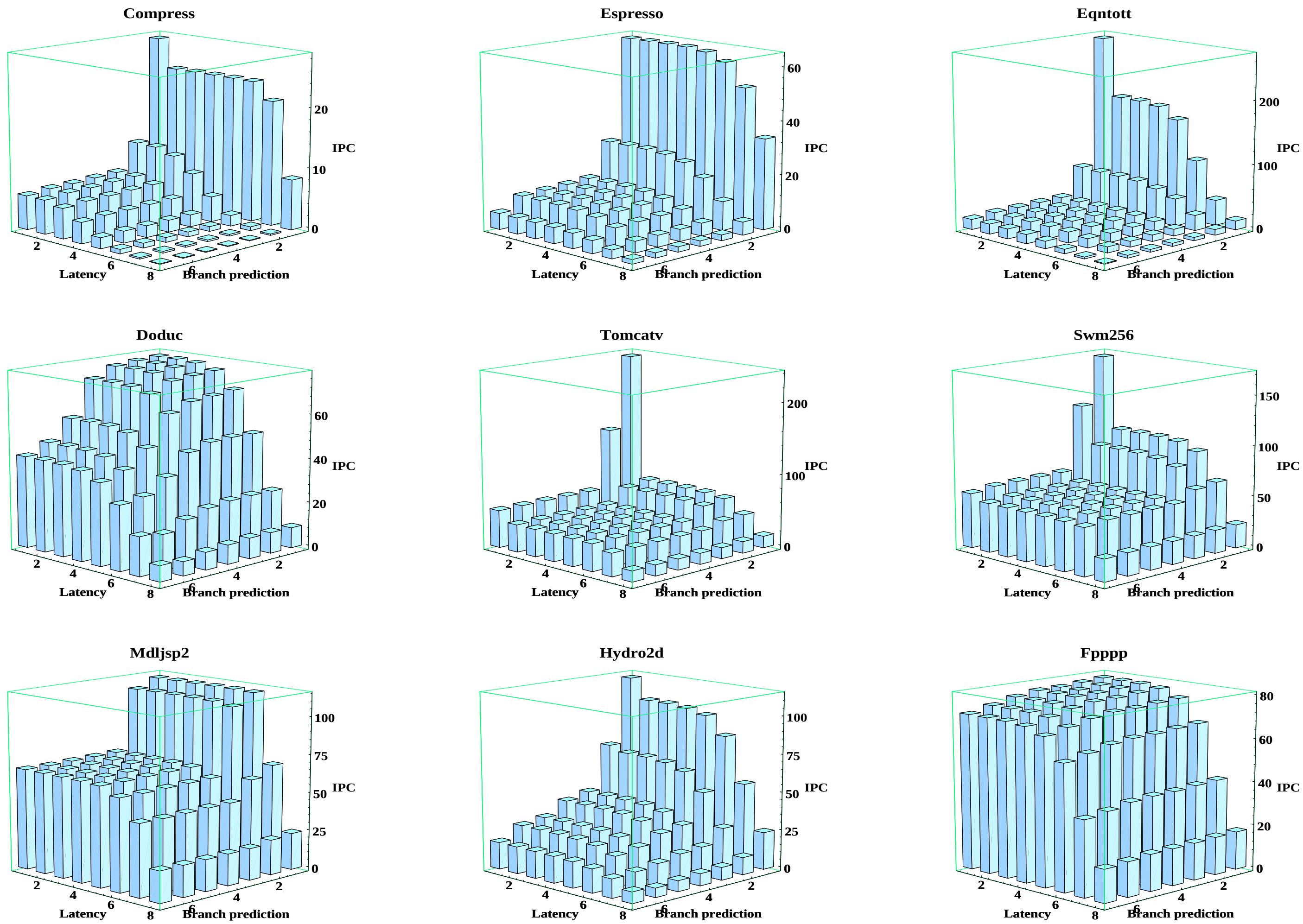
Performance vs. Latency and Prediction



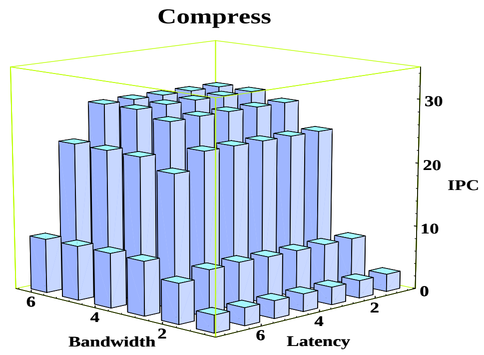
	0	1	2	3	4	5	6
Latency (cycles)	Perf	1	3	10	32	100	320
Branch prediction	Perf	Futr	PA/G	pa/g	2c/g	gsh	2c

- Steeper branch wall as compared to memory wall
  - With perfect branch prediction, need latency $<=$ 100
  - With current branch predictors, need latency $<=$ 10

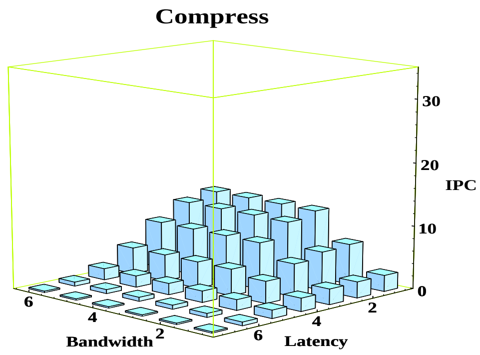
Performance vs. Latency and Prediction



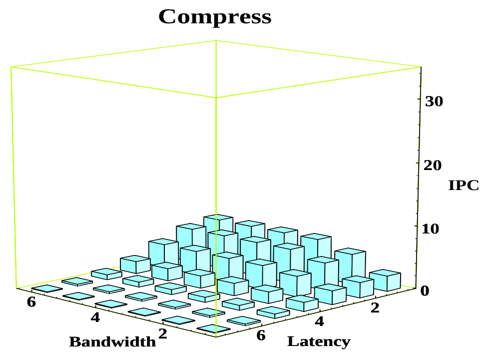
Compress vs. Latency, Bandwidth, and Prediction



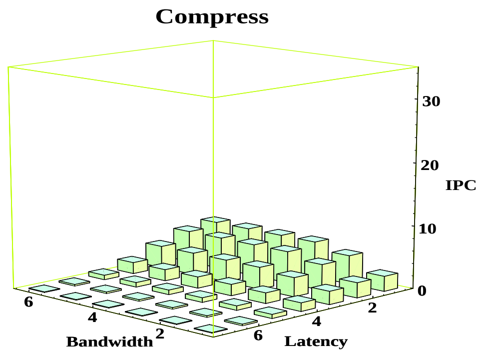
Perfect prediction



Futuristic prediction



Bimodal/gshare prediction



Bimodal prediction

Simulation Conclusions

- The branch wall is more significant than either memory wall
- Memory bandwidth solvable by good engineering and money
- For many applications memory latencies of 100 cycles or more are tolerable
- Future improvement in branch prediction may be limited

Attractive Future Research Areas

- Better branch prediction
  - Prediction-Accuracy-Based Branch Classification
  - Impact-Based Branch Classification
  - Limits of Predictability
- Exploit large-grain flow predictability
  - Multiprocessors on a die
  - Simultaneous multithreading
  - Multiscalar
- Better quantification of implementation tradeoffs

Prediction Accuracy Based Branch Classification

- Most branches have fairly high prediction accuracy
- Most mispredicts are caused by “good” branches
- Inherent limitation to branch prediction?

Impact-Based Branch Classification

- Branch mispredicts can have varying impacts
- Higher impact mispredicts cause greater performance loss
- Motivation for *impact-based* branch prediction strategy?