Workload Characterization: Can it save Computer Architecture and Performance Evaluation?

Lizy Kurian John
The Laboratory for Computer Architecture (LCA)
ECE Department, UT Austin
ljohn@ece.utexas.edu
(512)-232-1455
http://www.ece.texas.edu/projects/ece/lca/
http://www.ece.utexas.edu/~ljohn

This presentation includes work by several of my students, especially

- Deepu Talla
- Ramesh Radhakrishnan
- Tao Li
- Yue Luo
- Rob Bell Jr
- Aashish Phansalkar

Basic Belief

- There are bottlenecks that exist in modern computer systems, which if precisely unveiled, will lead to appropriate architectures and architectural enhancements.

Media Workload Characterization Example – Study on effectiveness of MMX- using Discrete Cosine Transform (DCT)

	Pentium II without MMX		*Pentium II with MMX*	
	Clocks	**Eff. Comp.**	**Clocks**	**Eff. Comp.**
Maximum compiler optimizations	3500	0.15	2375	0.24 (6%)
Ideal case	≈512	1	≈128	4 (100%)

Media Workload Characterization Example – Study on effectiveness of MMX- using Discrete Cosine Transform (DCT)

	Pentium II without MMX			*Pentium II with MMX*		
	Clocks	**IPC**	**Eff. Comp.**	**Clocks**	**IPC**	**Eff. Comp.**
Maximum compiler optimizations	3500	1.47	0.15	2375	1.04	0.24 (6%)
Perfect memory System (prefetching)	2737	1.88	0.20	1578	1.56	0.36 (9%)
Ideal case	≈512	-	1	≈128	-	4 (100%)

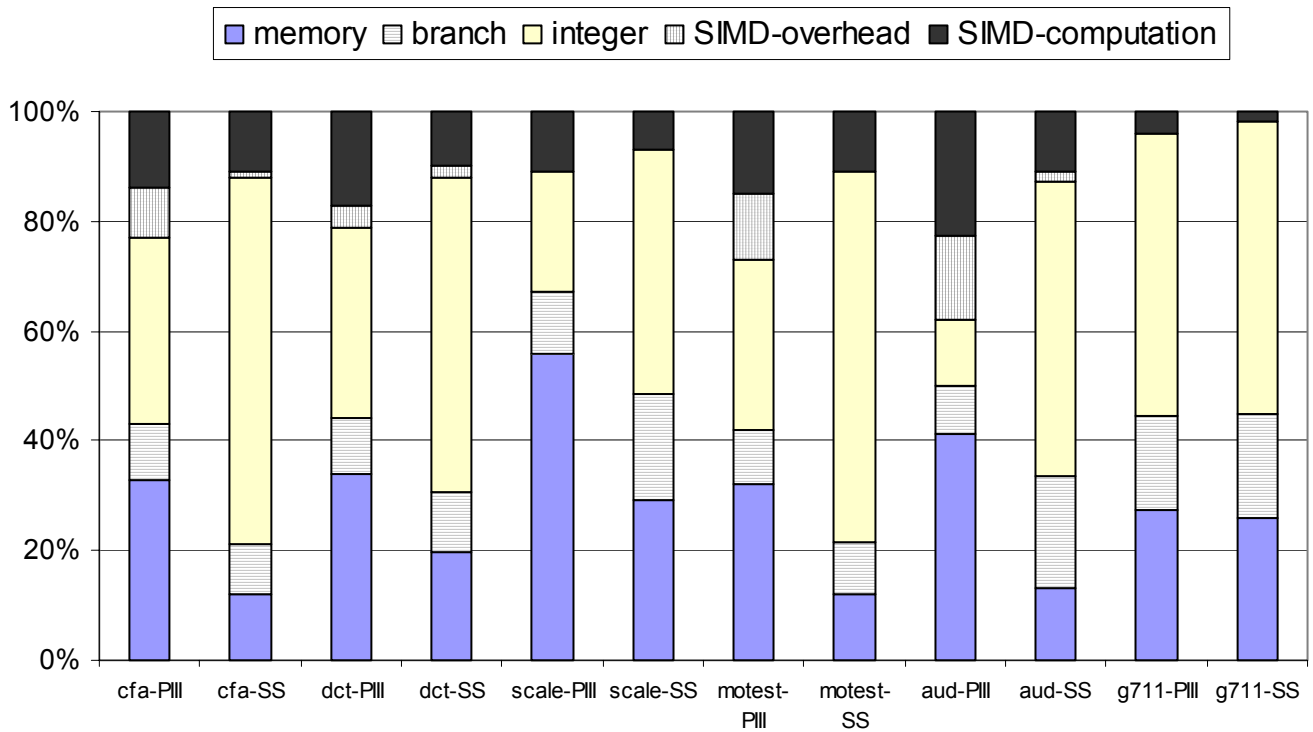
Only the instructions shown in red are MMX computations. All other instructions are simply supporting these computations.

Pentium III – SIMD code for Discrete Cosine Transform (DCT)

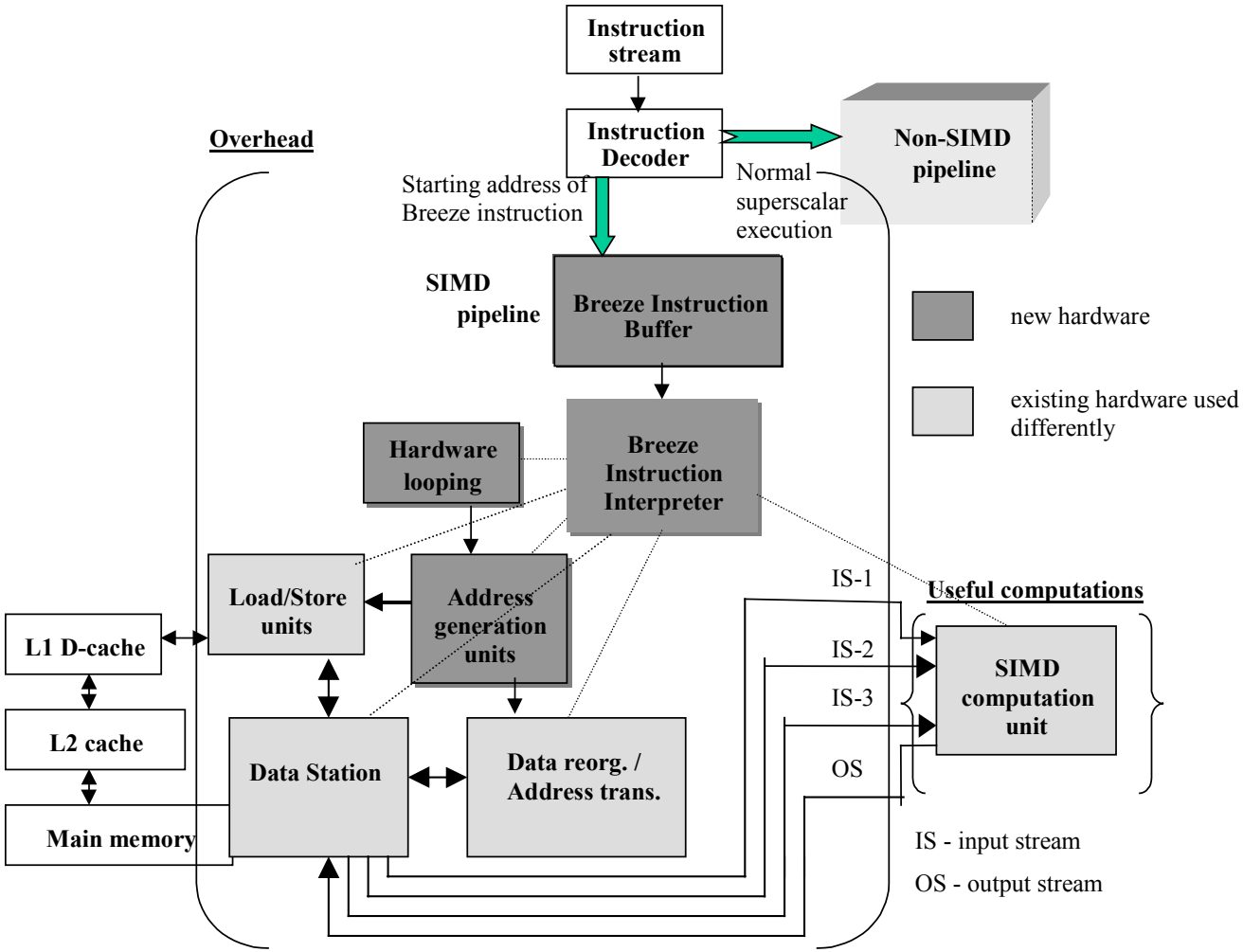
lea	ebx, DWORD PTR [ebp+128]	load/address overhead
mov	DWORD PTR [esp+28], ebx	load/address overhead
$B1$2:		
xor	eax, eax	address overhead
move	dx, ecx	address overhead
lea	edi, DWORD PTR [ecx+16]	load/address overhead
mov	DWORD PTR [esp+24], ecx	load/address overhead
$B1$3:		
movq	mm1, MMWORD PTR [ebp]	load overhead
pxor	mm0, mm0	initialization overhead
pmaddwd	**mm1, MMWORD PTR [eax+esi]**	**True Computation**
movq	mm2, MMWORD PTR [ebp+8]	load overhead
pmaddwd	**mm2, MMWORD PTR [eax+esi+8]**	**True Computation**
add	eax, 16address	overhead
paddw	**mm1, mm0**	**True Computation**
paddw	**mm2, mm1**	**True Computation**
movq	mm0, mm2	load related overhead
psrlq	mm2, 32	SIMD reduction overhead
povd	ecx, mm0	SIMD load overhead
movd	ebx, mm2	SIMD load overhead
add	ecx, ebx	SIMD conversion Overhead
mov	WORD PTR [edx], cx	store overhead
add	edx, 2	address overhead
cmp	edi, edx	branch related overhead
jg	$B1$3	loop branch overhead
$B1$4:		
move	cx, DWORD PTR [esp+24]	load/address overhead
add	ebp, 16	address overhead
add	ecx, 16	address overhead
move	ax, DWORD PTR [esp+28]	load/address overhead
cmp	eax, ebp	branch related overhead
jg	$B1$2	loop branch overhead

Breakdown of dynamic instructions

- Approximately 75%-85% of dynamic instructions are supporting



The MediaBreeze architecture focuses on the parallelism in the supporting instructions rather than the actual media computations.



Performance of MediaBreeze

- The MediaBreeze architecture gives up to 27X performance on DSP kernels and up to 2X on media applications over the best SIMD performance.

Area, power, and timing implications of MediaBreeze

- Area – 0.31 mm^2 (overall chip area increase is 0.3%)
- Power – 430 mW at 1 GHz (less than 1% of the overall processor power
- Timing – Overall pipeline depth is not increased.

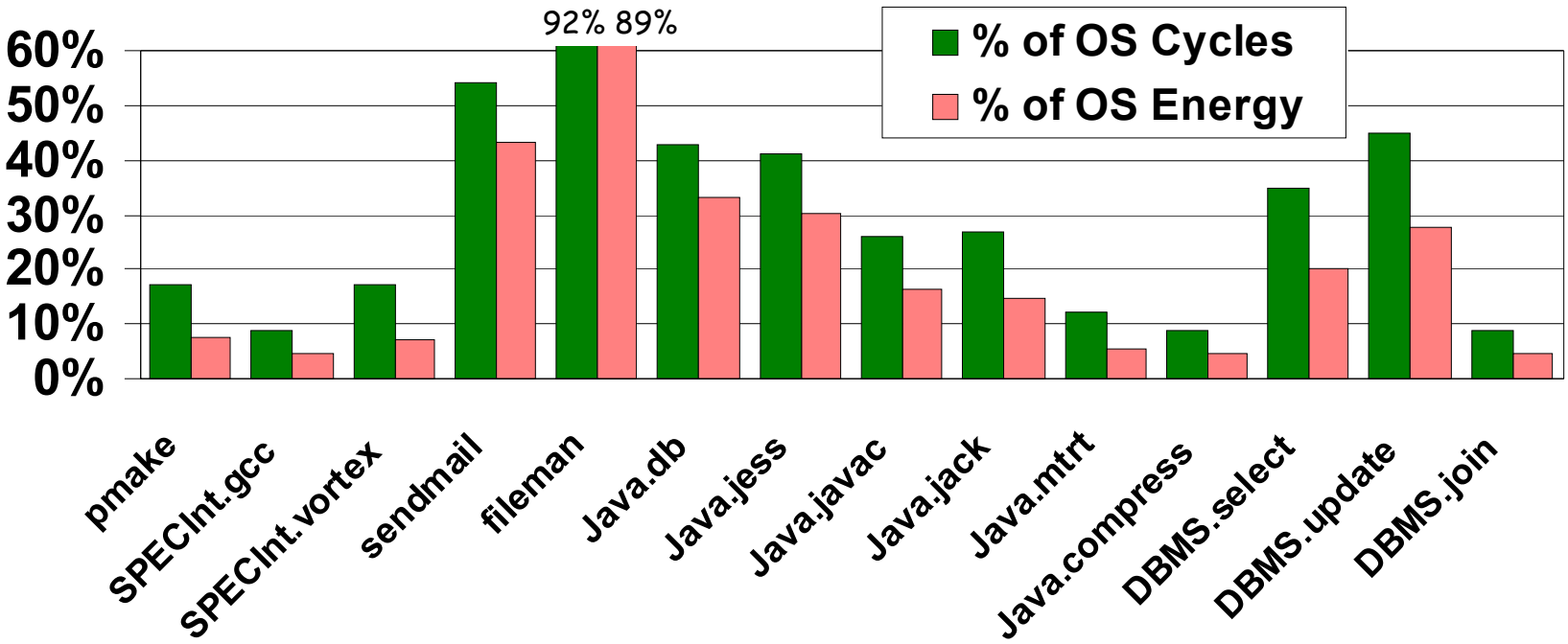
Simple Solutions

Elegant Solutions

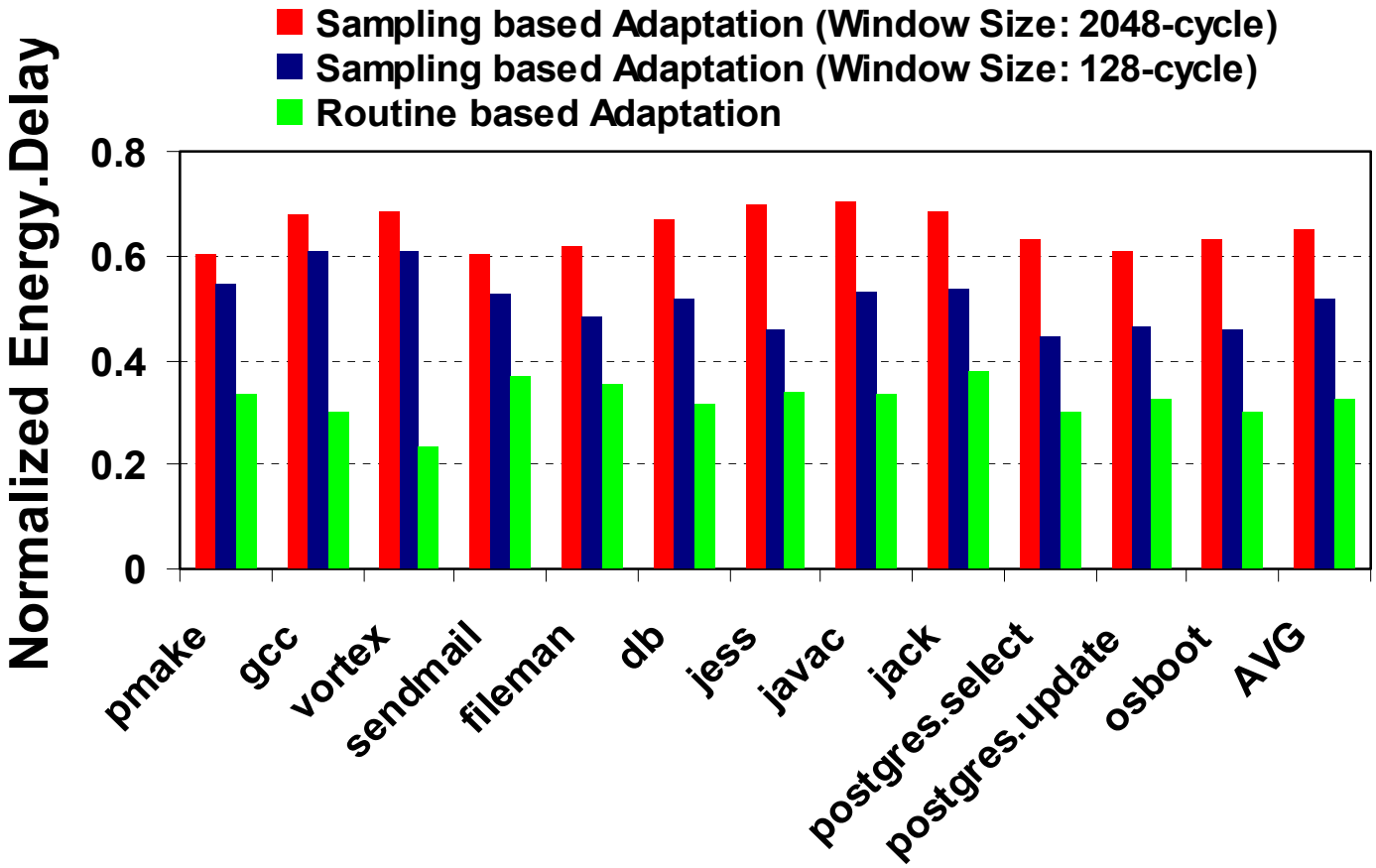
Power Aware Adaptive Architectures

Detects phases during program execution and adapts hardware characteristics to suit features of the phase

OS Energy Dissipation



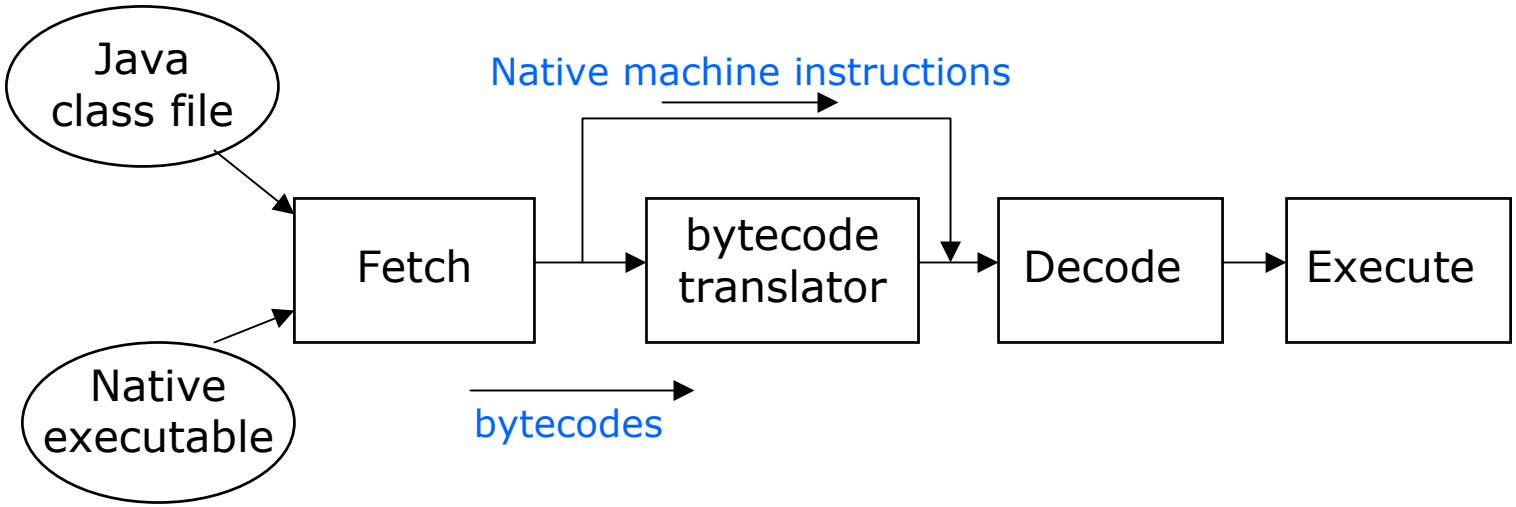
OS Power & Performance Tradeoff



Java Acceleration for General Purpose Processors

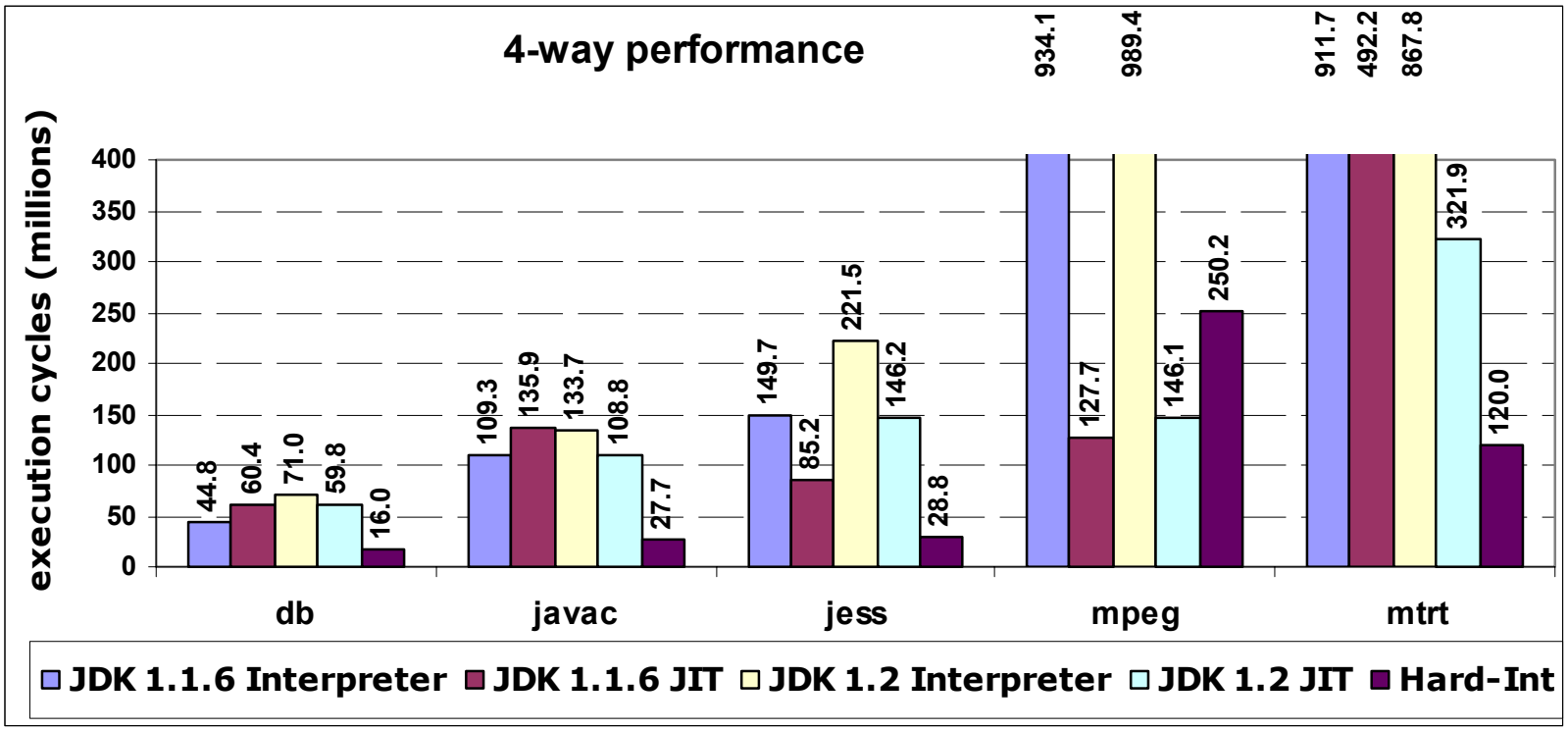
- What's the biggest bottleneck to alleviate?
- Object-oriented nature?
- Translation
- Hardware Translator for Java Bytecodes

The Java Hardware Interpreter



- No changes to processor core
  - light-weight Java run time environment

HardInt Performance



- Hard-Int performs consistently better than the interpreter
- In JIT mode, significant performance boost in 4 of 5 applications.

Simple Solutions

Elegant Solutions

Architect's Treasure Chest

- Perhaps our treasure chest contains simple solutions to most problems we will encounter
- We need to be able to identify which is the right solution for the right problem
- Diagnosing the problem is the issue
- Workload characterization is the key to this diagnosis

Are computer architects becoming like doctors prescribing medicines without diagnosing the disease?

Are we getting too excited with all the wonderful things a new medicine can do?

Performance Evaluation

Compare this vehicle to others by using the FREE FUEL ECONOMY GUIDE available in the dealer showroom.

CITY MPG
24

HIGHWAY MPG
31

Actual Mileage will vary with options, driving conditions, driving habits and vehicle's condition. Results reported to EPA indicate that the majority of vehicles with these estimates will achieve between

20 and 28 mpg in the city

and between

26 and 36 mpg on the highway.

2004 GREEN CAR 2WD, 4 CYL, 2.0 LITER, MULTIPOINT FUEL INJECTION, 4-SPEED AUTO TRANS, CATALYST.

Estimated Annual Fuel Cost: $777

For Comparison Shopping
All vehicles classified as COMPACT CARS have been issued mileage ratings ranging from

13 to 48 mpg city

and

19 to 51 mpg highway.

See www.fueleconomy.gov

If cars were benchmarked like computers

- Mileage chart might have looked like

I-10	27.2 mpg
I-20	28.1 mpg
I-80	25.3 mpg
I-90	24.2 mpg
I-35	26.6 mpg
I-95	27.5 mpg
I-75	28.3 mpg

International Routes

I-10	27 mpg
I-20	28 mpg
I-80	25 mpg
I-90	24 mpg
Madrid's M-30	24 mpg
AutoBahn	30 mpg
Japan's Hwy 142	28 mpg

And we would have asked questions like

- Did you drive on I-80 in the summer or winter?
- Was it night or day?
- When you drove through Austin on I-35, was a Longhorn football game going on?

And in our car conferences, we would have accepted papers that

- Benchmarking results on most number of highways
- Benchmarked from end to end
- Benchmarked on highways in multiple weather conditions

Imagine running cars on

I-10 (CA to FL)	2460 miles
I-20 (TX to SC)	1540 miles
I-80 (CA to NJ)	2900 miles
I-90 (WA to MA)	3020 miles
I-35 (TX to MN)	1570 miles
I-95 (Maine-FL)	1920 miles
I-75 (FL to MI)	1790 miles

Abstracting these long roads

- CITY
- HIGHWAY

If we look at our computer performance evaluation trends, aren't we simply adding roads to the list?

Reducing Redundancy in Benchmarking

- SimPoint [Sherwood et. al]
- SMARTS [Wunderlich et. al]
- Benchmark clustering using PCA Analysis [Eeckhout et. al]

SimPoint

- Sherwood, et al. ASPLOS 2002
- Sample selection:
  - Clustering analysis of Basic Block Vectors to identify representative chunks of instructions
- Sampling unit size: 100 million instructions
- Sample size: 3-10
- Warm up: No explicit warm-up

SMARTS

- Wunderlich, et al. ISCA 2003
- Sample selection:
  - Selecting chunks evenly distributed in the instruction stream (systematic sampling)
- Sampling unit size: 1,000 instructions
- Sample size:
  - Depends on confidence interval requirement
  - Thousands to tens of thousands

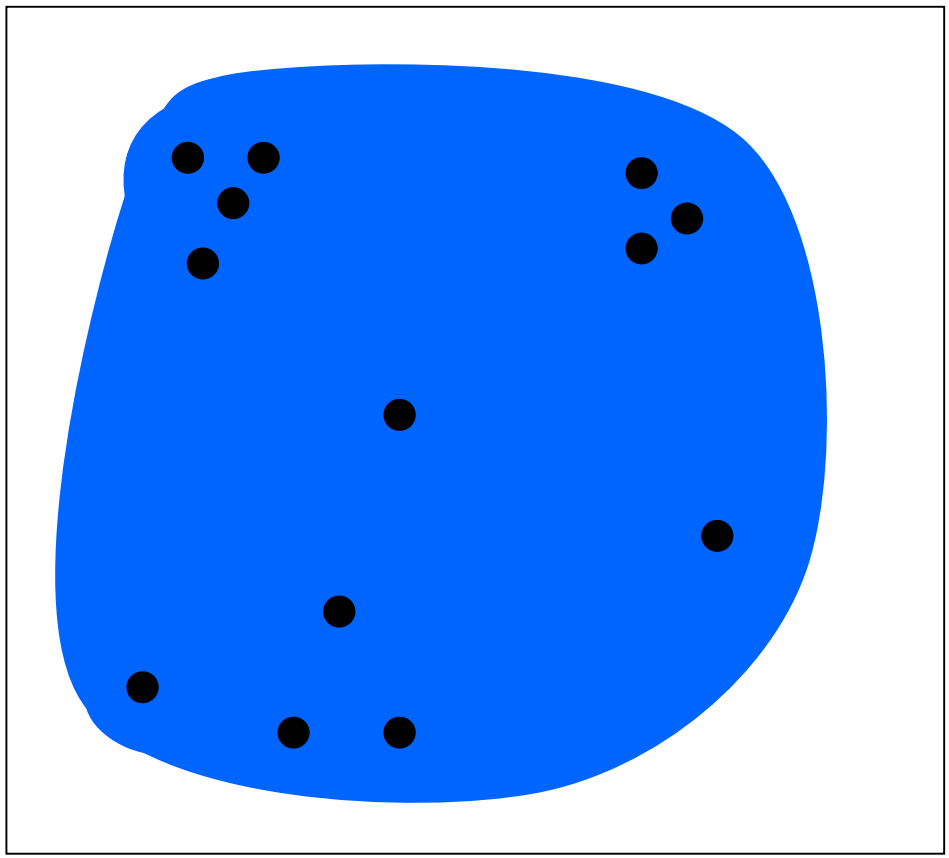
SIMPOINT

- Analogous to realizing that no need to go all over I-10 from California to Florida, if 10 miles around Phoenix, and 10 miles from San Antonio and 10 miles from El Paso are 10 miles from the New Mexico desert are taken, that's sufficient.

SMARTS

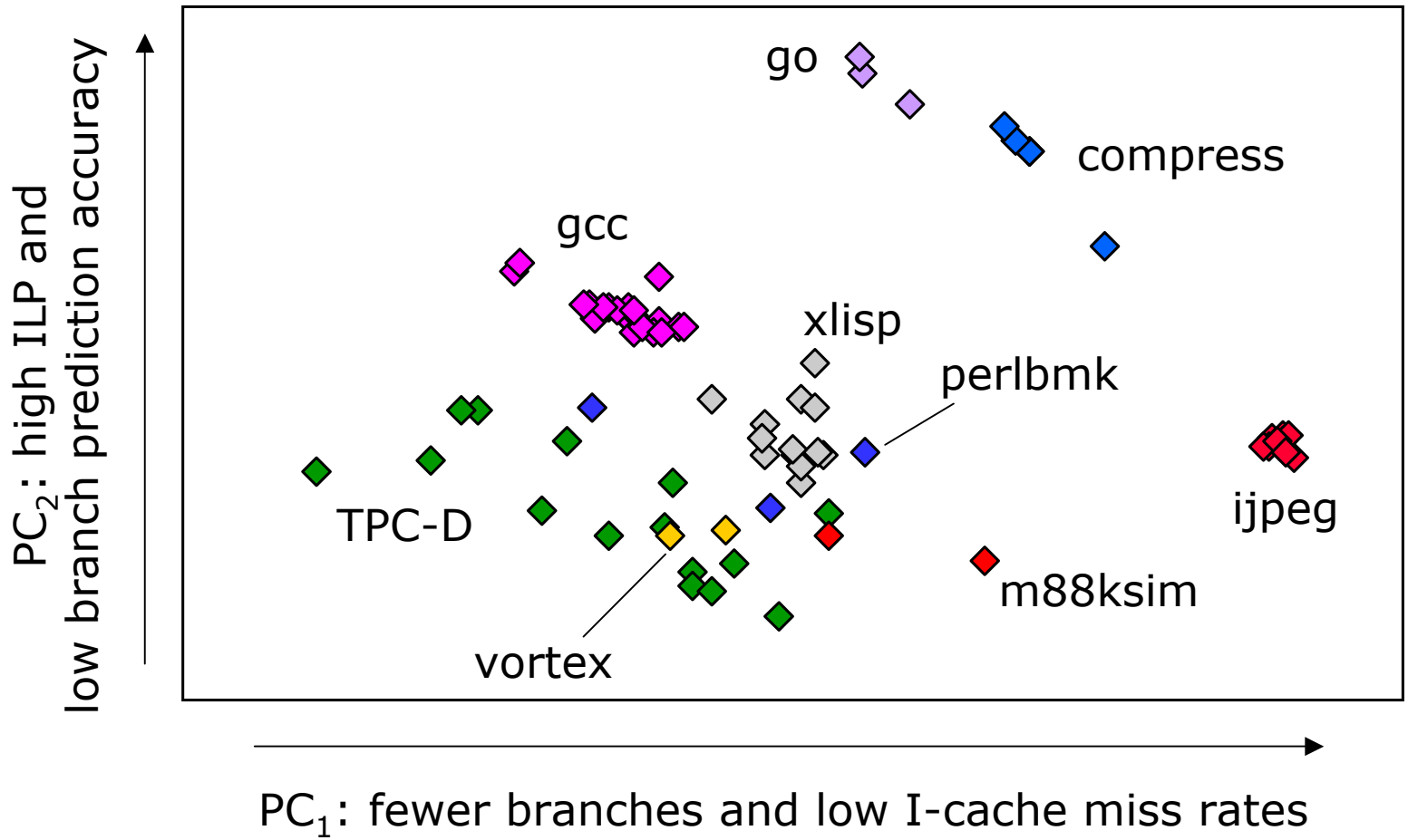
- Randomly picking some miles from anywhere will do.
- No need for representative sampling

Cluster analysis

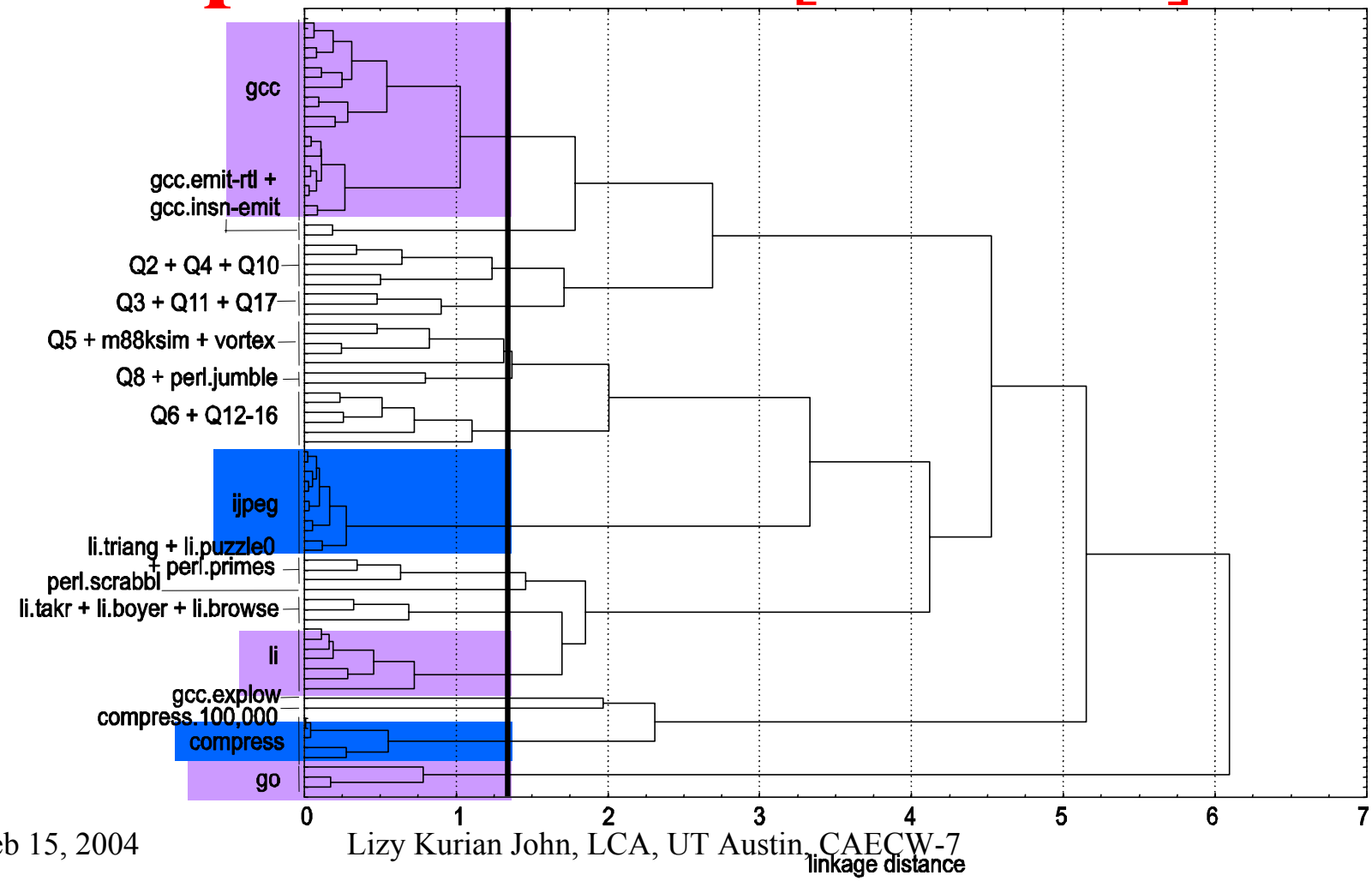


- **Linkage clustering**
- K-means clustering
- Iterative algorithm
- Based on distance between program-input pairs = linkage distance

Rescaled PCA space: PC_1 vs PC_2 [Eeckhout]



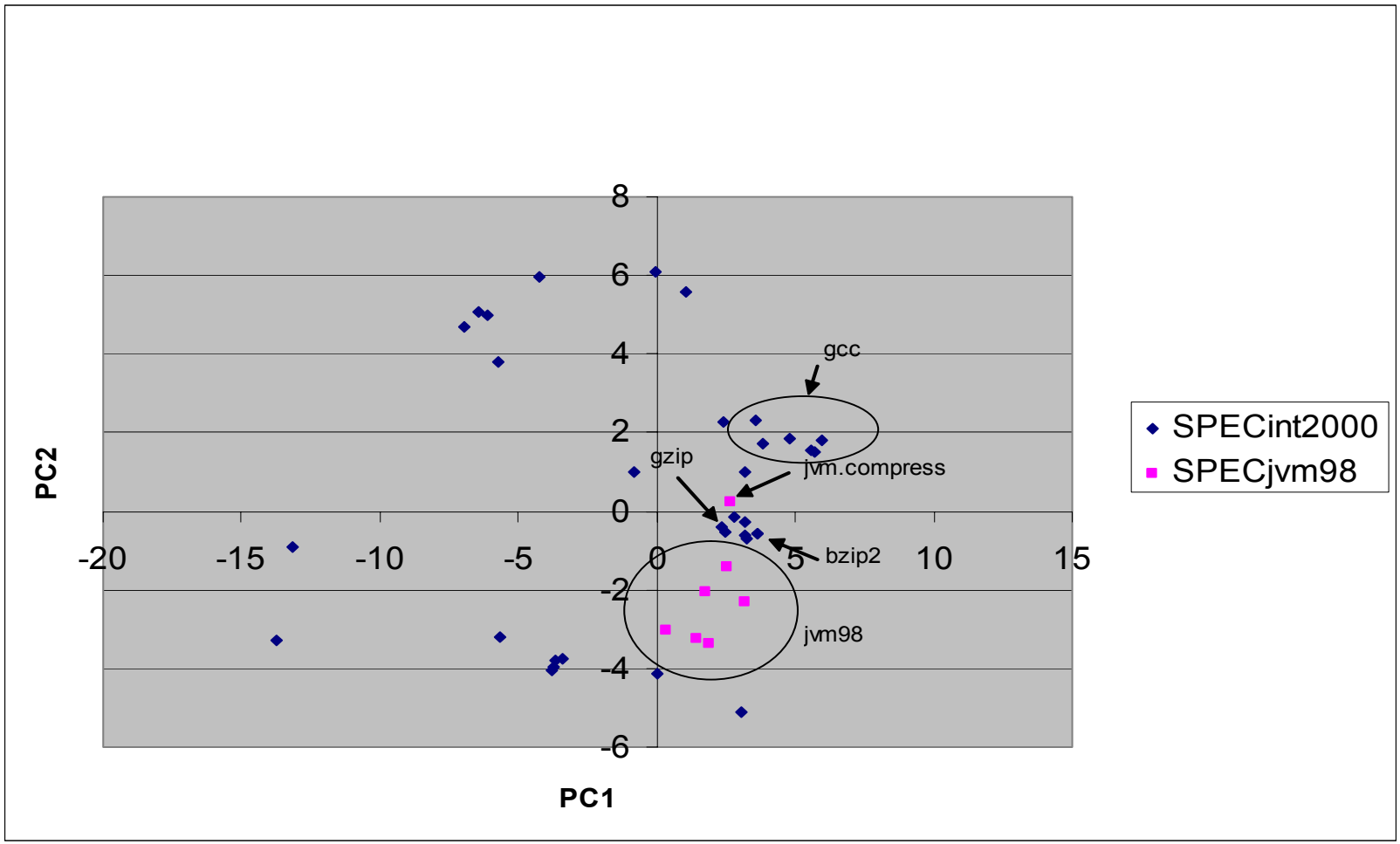
Dendrogram to select representatives [Eeckhout]



Cluster Analysis and PCA [Eeckhout]

- Analogous to realizing that I-10 and I-20 are very similar kinds of roads. Similarly I-80 and I-90 are very similar.

Plot for the scores of L1 cache access behavior of SPECint2000 and SPECjvm98 benchmark suites



The Return of Synthetic Benchmarks?

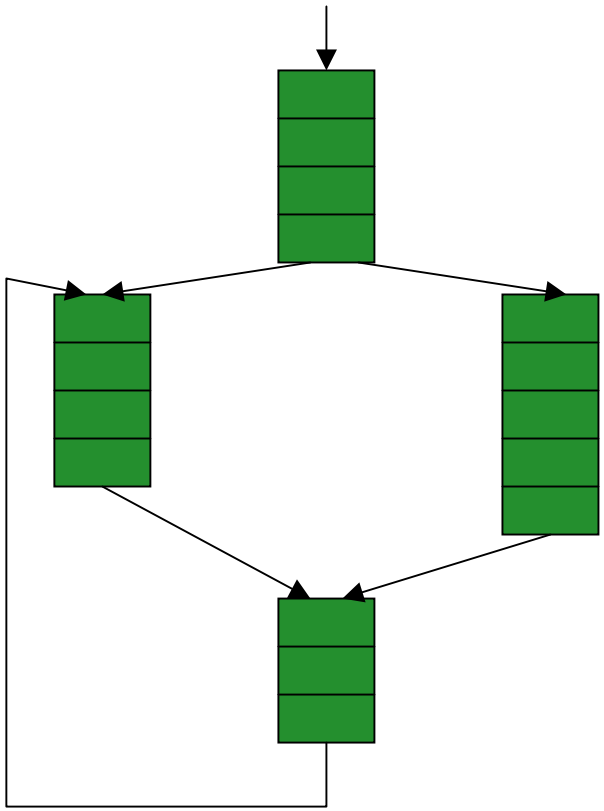
A framework to generate synthetic benchmarks that are:

- Representative of applications or user specifications
- Automatically generated
- Generated and executed using user parameters
- Source code and executables
- Portable to multiple hardware & simulation systems

Statistical Simulation

- Statistical Simulation using Synthetic Traces
  - Carl and Smith
  - Nussbaum and Smith
  - Oskin et al.: HLS
  - Eeckhout et al.
- Executable code built from the workload characterization of well-correlated statistical simulation systems
  - Automatic Benchmark Synthesis

Synthetic Traces in Statistical Simulation



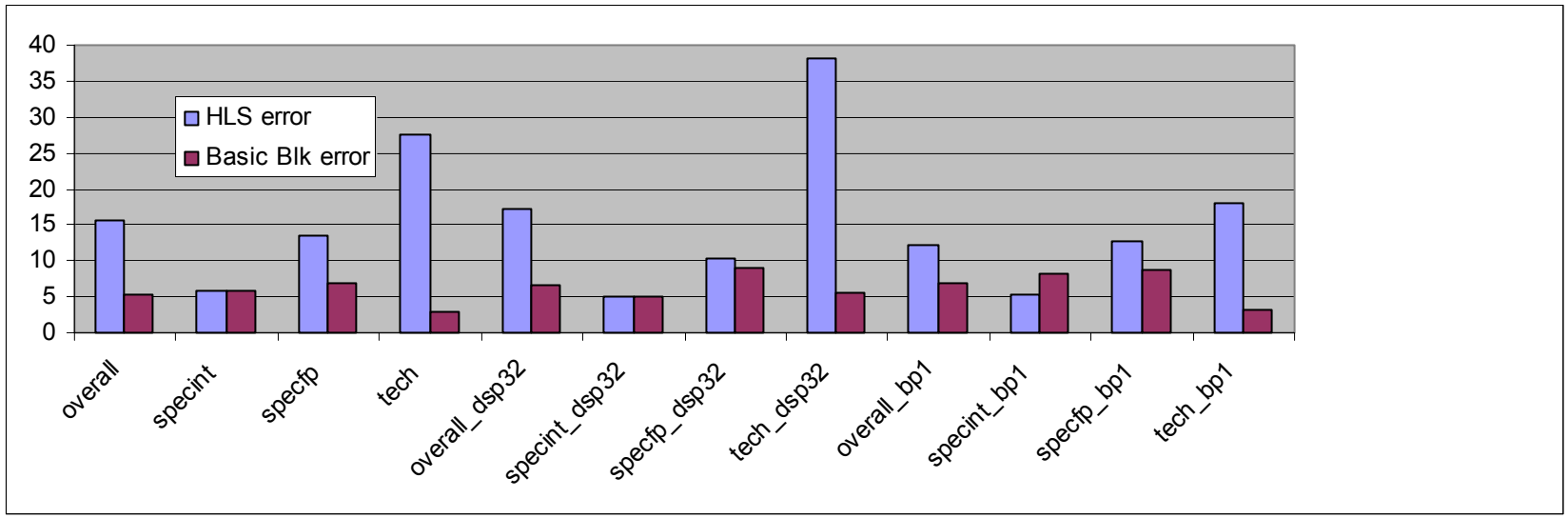
1. Collect global statistics
   - Basic block size
   - Instruction Mix
   - Instruction Dependencies
   - Branch predictability
   - L1/L2 cache statistics
2. Generate basic blocks
3. Connect them together into a graph (HLS) or generate a trace
4. Execute in order, simulating cache misses and branch mispredicts

Improving Correlation

Track information on a per basic block basis

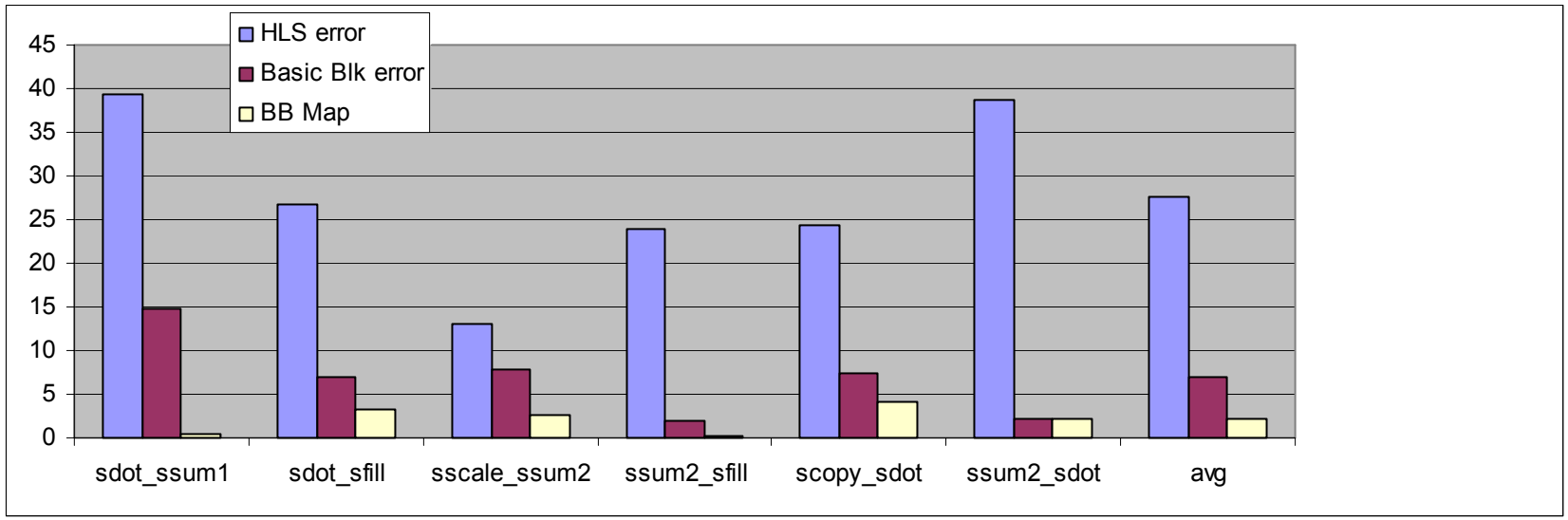
- Basic block size
- Instruction sequences
- Merged dependency information
- Cache hit/miss information
- Branch predictability

Summary by Benchmark Class



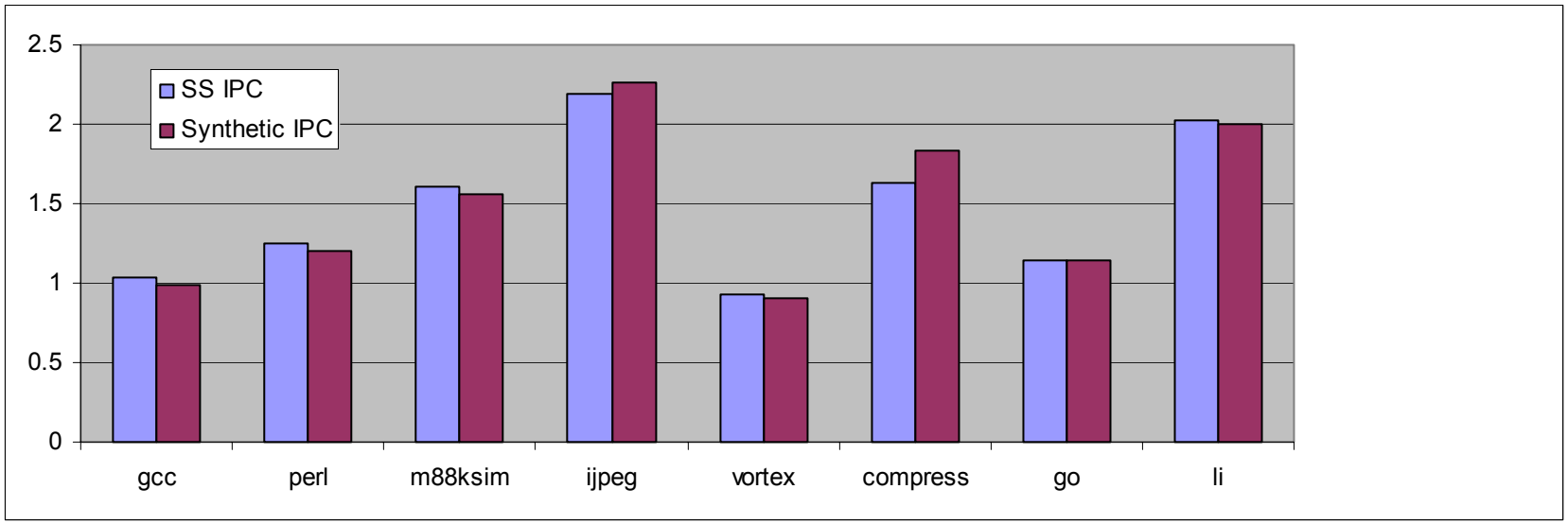
Tracking characteristics on a basic block granularity reduces error

Adding Structure: BB Maps



- Multi-phase programs can benefit from simulation using a graph of basic block frequencies
- Programs are back-to-back two-loop combinations of the technical loops

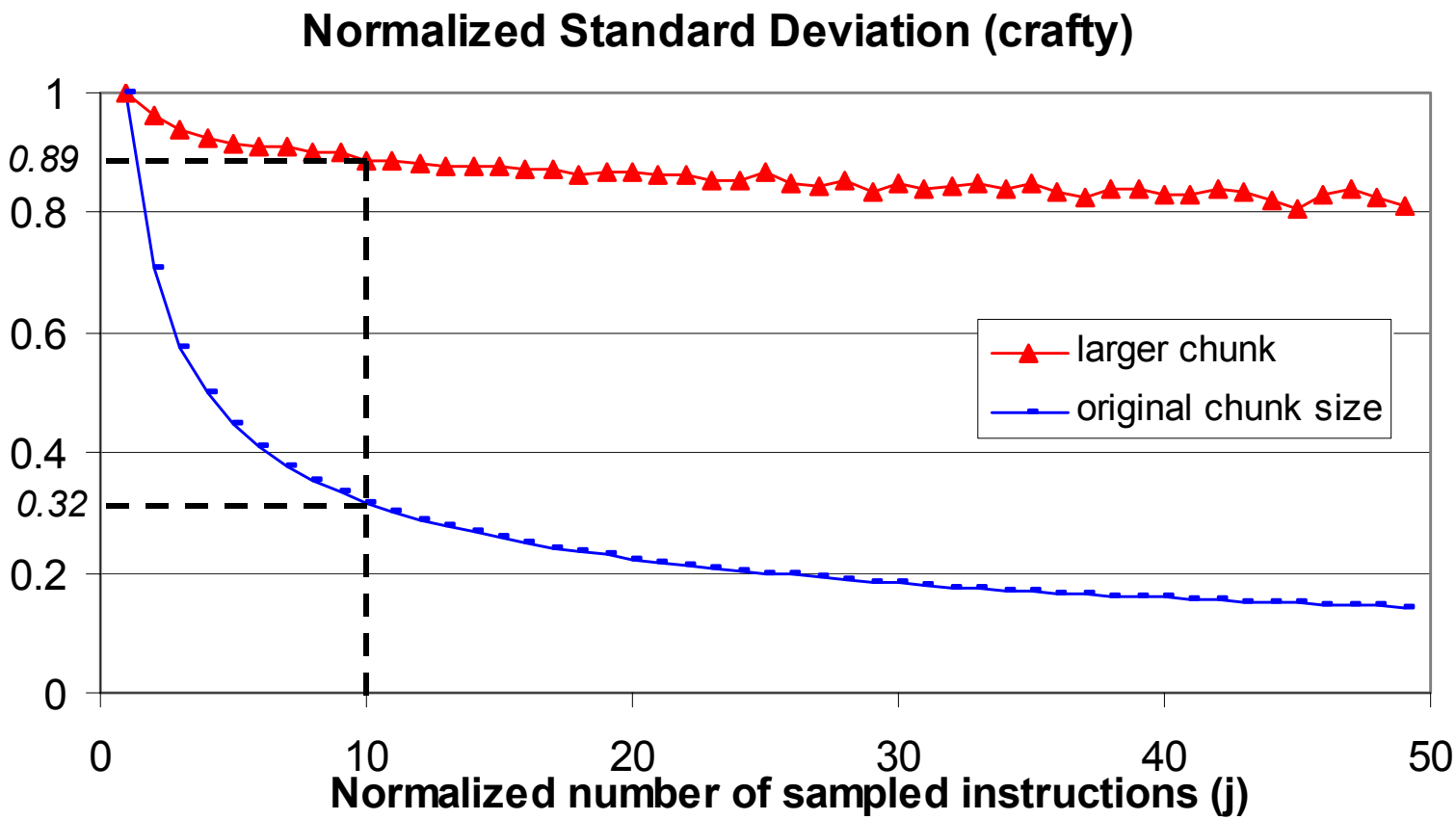
IPC from Synthetic Trace



Converted workload characteristics to C-code/ASM statements and run through SimpleScalar

Use of statistical theory?

We architects – are we unwilling to use statistical theory?

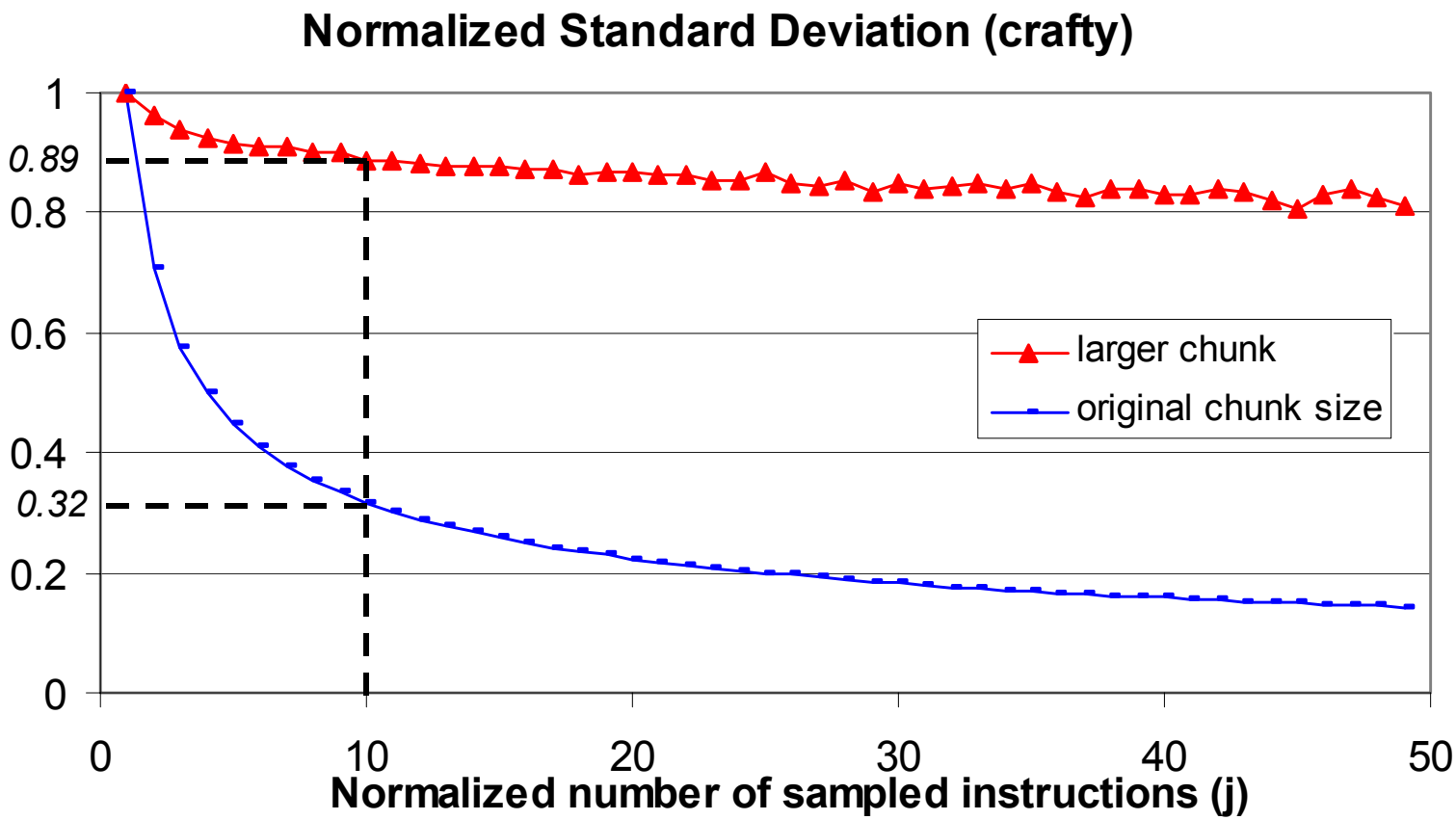

Normalized Standard Deviation (crafty)
1
0.89
0.8
0.6
0.4
0.32
0.2
0
0
10
20
30
40
50
Normalized number of sampled instructions (j)
larger chunk
original chunk size

SimPoint

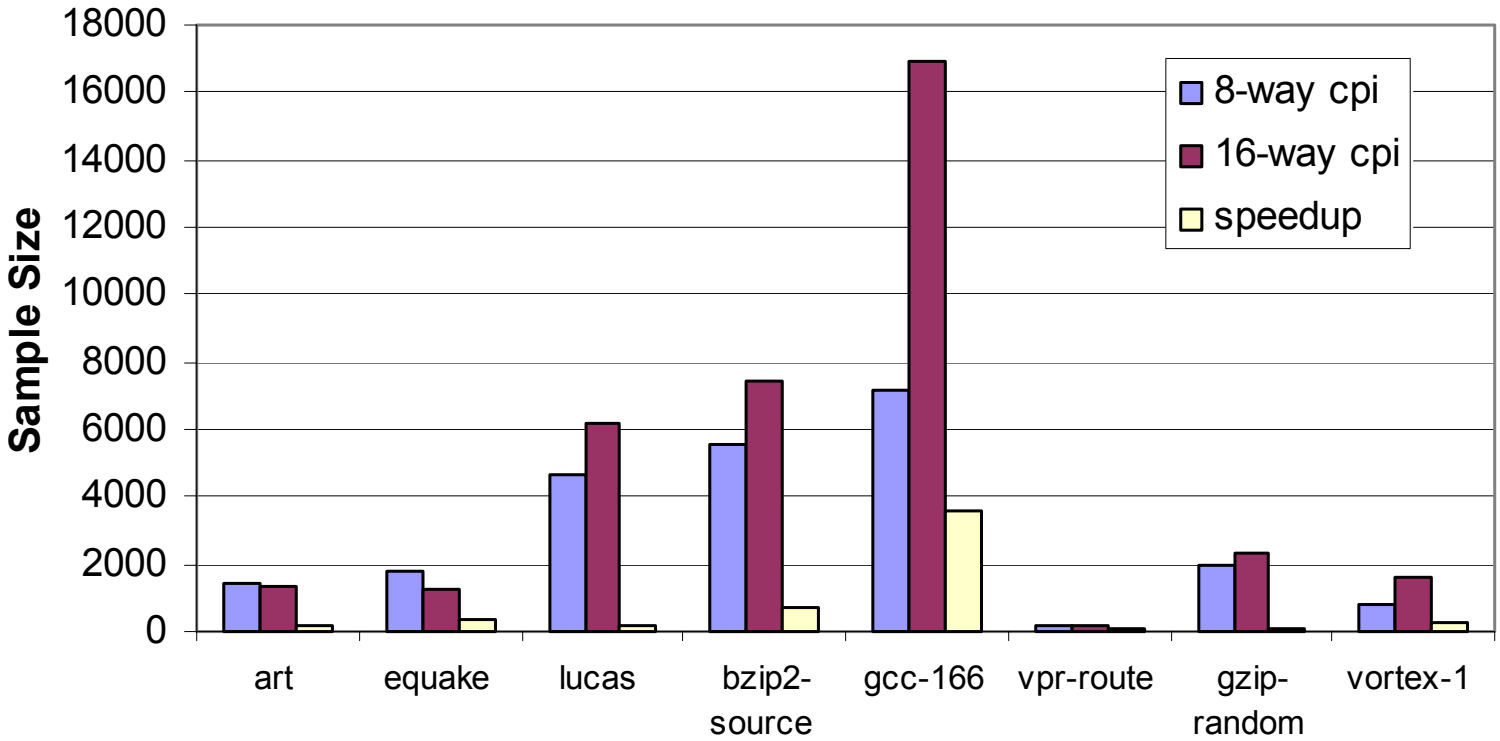
- Sherwood, et al. ASPLOS 2002
- Sample selection:
  - Clustering analysis of Basic Block Vectors to identify representative chunks of instructions
- Sampling unit size: 100 million instructions
- Sample size: 3-10
- Warm up: No explicit warm-up

SMARTS

- Wunderlich, et al. ISCA 2003
- Sample selection:
  - Selecting chunks evenly distributed in the instruction stream (systematic sampling)
- Sampling unit size: 1,000 instructions
- Sample size:
  - Depends on confidence interval requirement
  - Thousands to tens of thousands


Normalized Standard Deviation (crafty)
1
0.89
0.8
0.6
0.4
0.32
0.2
0
0
10
20
30
40
50
Normalized number of sampled instructions (j)
larger chunk
original chunk size

Estimating speedup Required sample size



Sample size required to achieve 2% relative error at 95% confidence level

(Comparing reduced data set: Test of population median

- If the populations are the same, the population median/mean should be the same
- Wilcoxon signed rank test

Metrics	Reduced data set	*p*-value
CPI on 8-way machine	Test	0.06445
	Train	0.02734
	MinneSPEC	0.04883
CPI on 16-way machine	Test	0.03711
	Train	0.01953
	MinneSPEC	0.03711
Speedup (16-way vs. 8-way)	Test	1
	Train	0.375
	MinneSPEC	0.6953

Comparing reduced data set: Test result

- None of the reduced data sets have the same population median CPI with reference data set
- All reduced data sets show same median speedup as the reference data set

Future Workloads

Life, Death and Games

Future Workloads – Life, Death and Games

- Life Science - Pharmaceuticals, Drug Discovery
- Death Science – Military Applications, Weapon Simulation, Crash Analysis, Scientific Computing
- Games – Multiplayer, Natural Language Recognition/Semantic Analysis

Risk of missing truly innovative architectures?

Inventions in Search of Applications eg: Laser

If workload characterization can lead to synthetic workloads of the future, endless future workloads can be synthesized that may lead to truly innovative architectures

Workload Characterization: Can it save Computer Architecture and Performance Evaluation?

Possibly Yes.

Nothing else possibly can.

Task is on us, the workload characterization community

We need to abstract program behavior into essential attributes which can help new architectures and performance evaluation