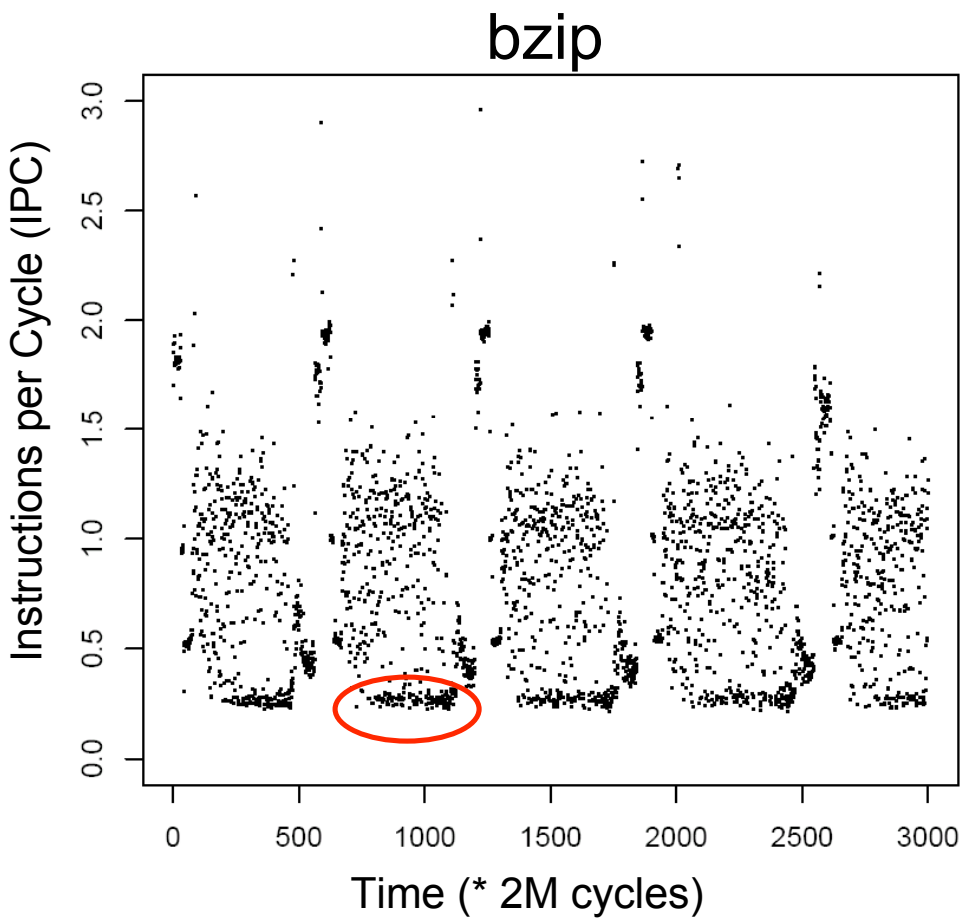
Time Interpolation: So Many Metrics, So Few Registers

Todd Mytkowicz	U of Colorado, Boulder
Peter F. Sweeney	IBM Research
Matthias Hauswirth	U of Lugano, Switzerland
Amer Diwan	U of Colorado, Boulder

Software Performance Analyst



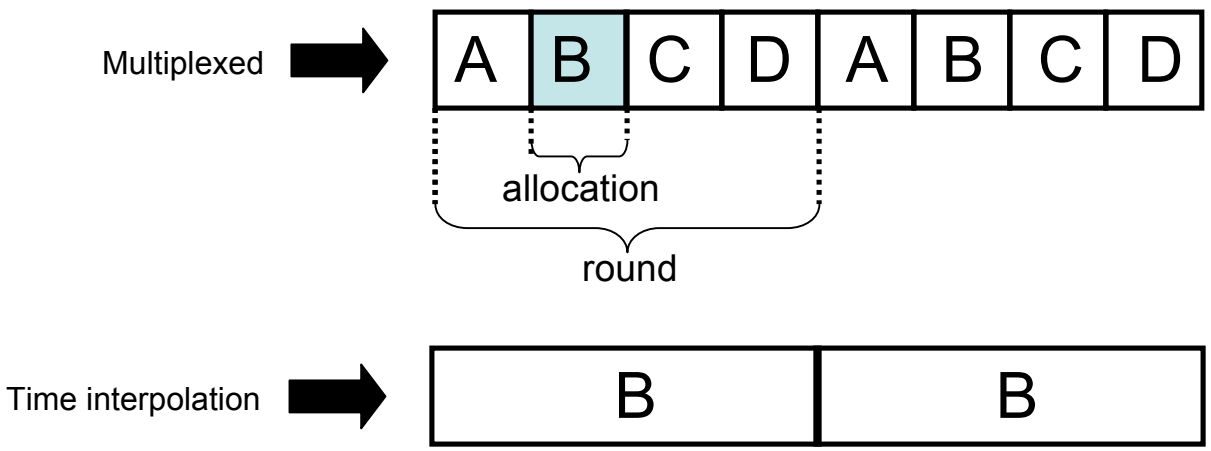
What other metrics correlate with IPC when it is low?

Hardware Performance Monitors

- Available on all microprocessors today
- Lots of metrics: between 32-475
- But can't collect all the metrics at once
  - Only 2-18 registers
  - Restrictions on what can be collected together

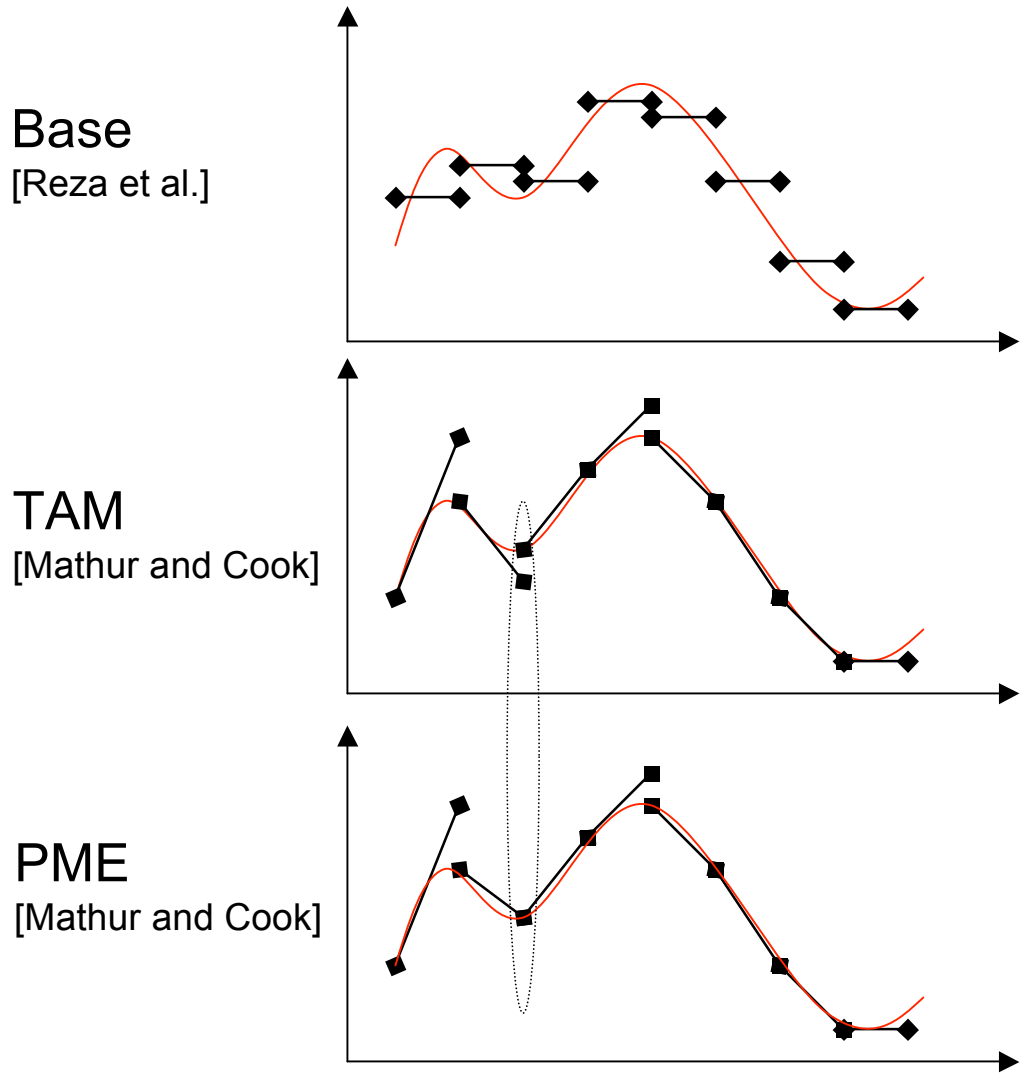
- We need time interpolation:
  - To allow reasoning about the relationships between metrics collected at different times

Multiplexing



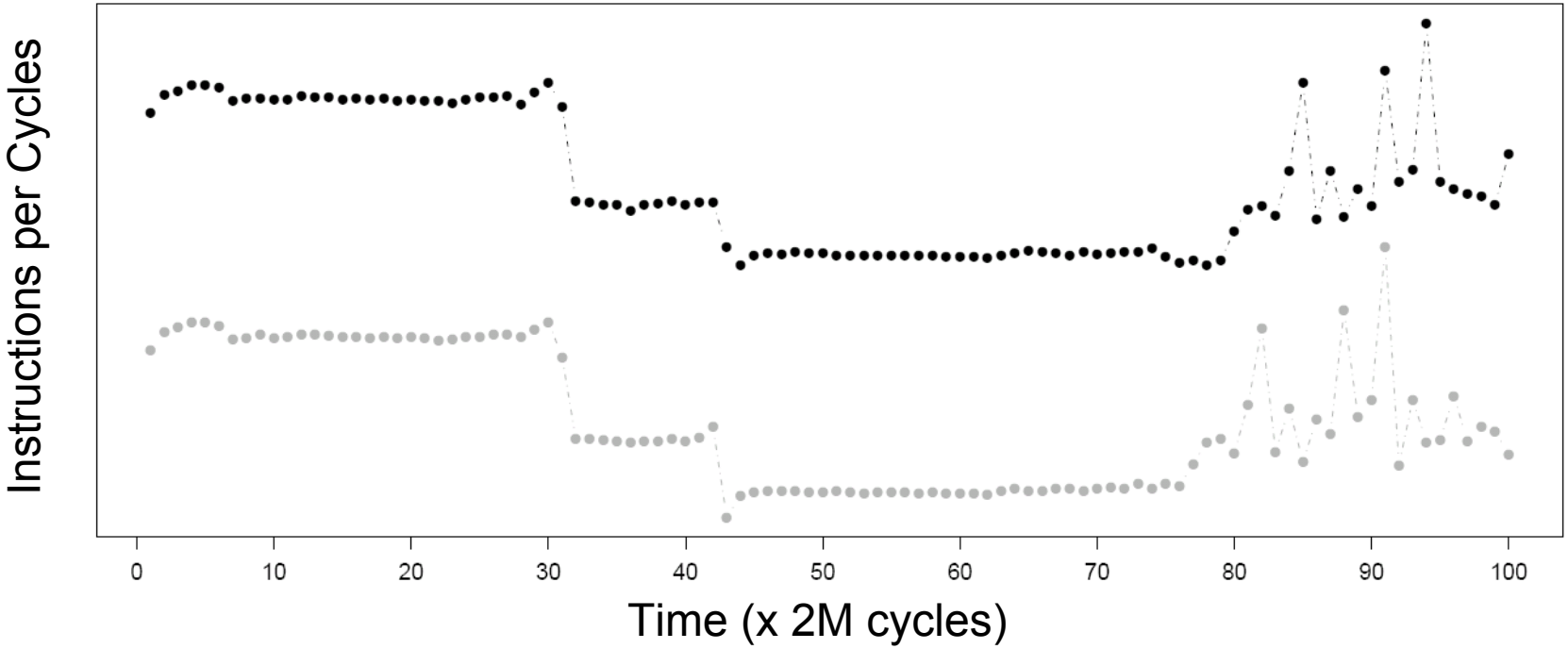
- Cycles per allocation: fixed for each allocation.
- Number of allocations per round; fixed for each round (e.g. 4).
- Time interpolation: derive round count from allocation count.

Three Multiplexing Approaches



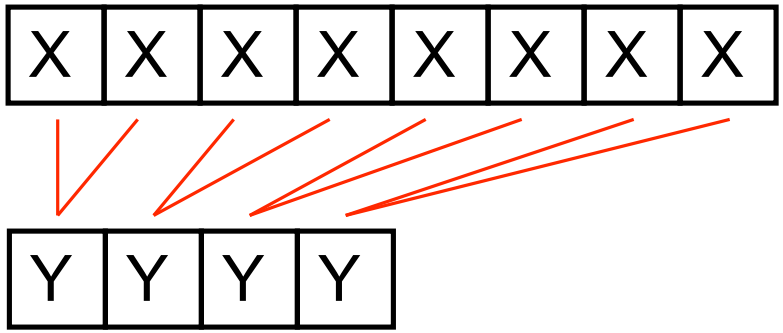
Why Trace Alignment is Hard?

- First 200M cycles of two runs of bzip2 benchmark
- After only 160M cycles, diverge



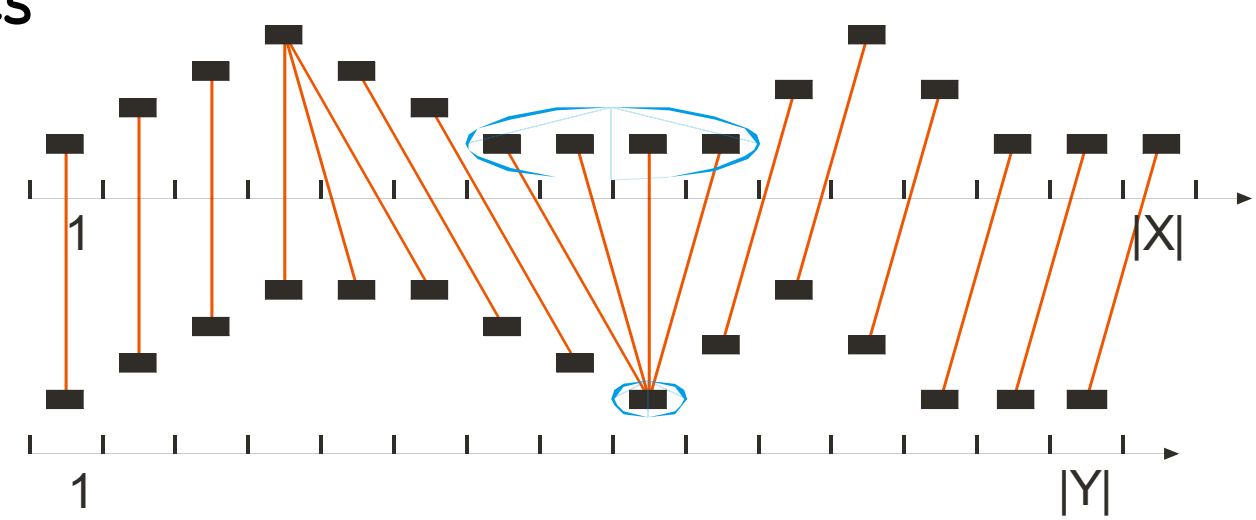
Full Uniform Scaling (FUS)

- Given two sequences X and Y, $|X| > |Y|$, $y_{floor(k*(|Y|/|X|)} \rightarrow x_k$



Dynamic Time Warping (DTW)

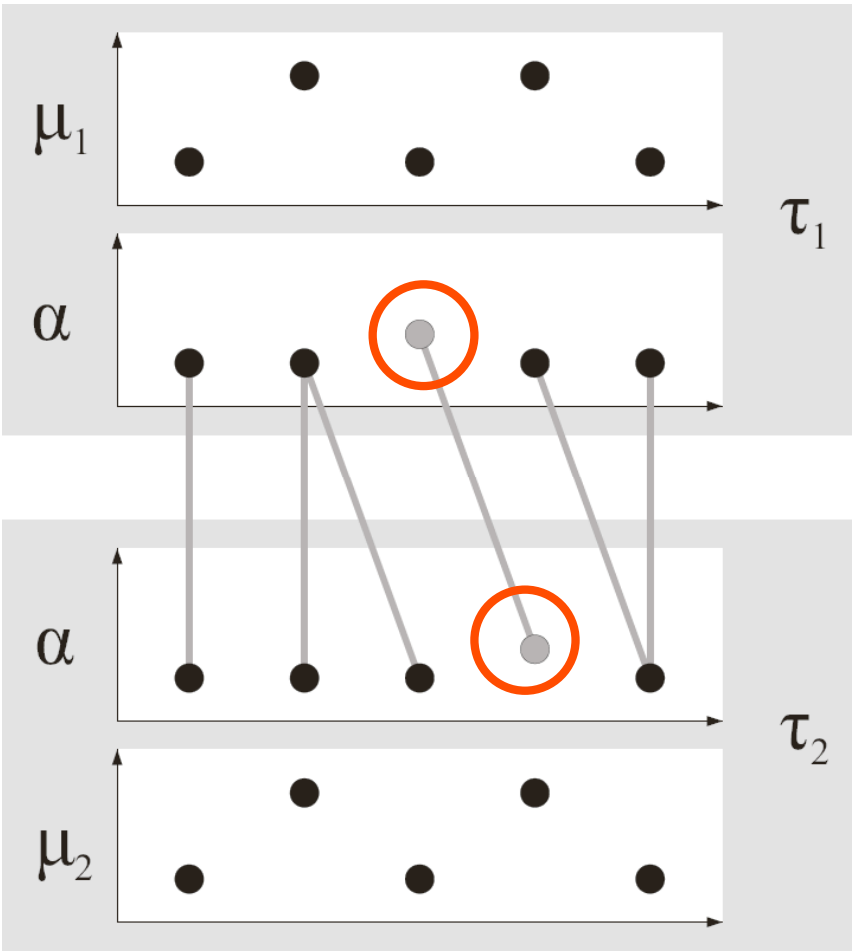
- Needs a common metric, the *alignment metric*, in two traces



- Dynamic Time Warping [Berndt and Clifford 1994]

$$\text{DtwError} = \sum_{k=1}^{|W|} |x_i - y_j| \text{ where } w_k = (i, j)$$

What is a good alignment metric?



Alignment metric should reflect the key transitions in the metrics to be correlated

Picking an alignment metric

- Fixed: Analyst carefully picks a metric
  - e.g., *IPC*
- Variable: Pick a different alignment metric for each pair of metrics to be correlated
  - e.g., *Sum*: pick an alignment metric that is most correlated with both metrics in the pair

Deficiencies of Prior Evaluations

- Multiplexing
  - Evaluated without accounting for time varying behavior
- Trace Alignment
  - FUS
    - None known
  - DTW
    - Visual inspection of time-varying relationship between metrics, but did not quantify effectiveness!

Contributions

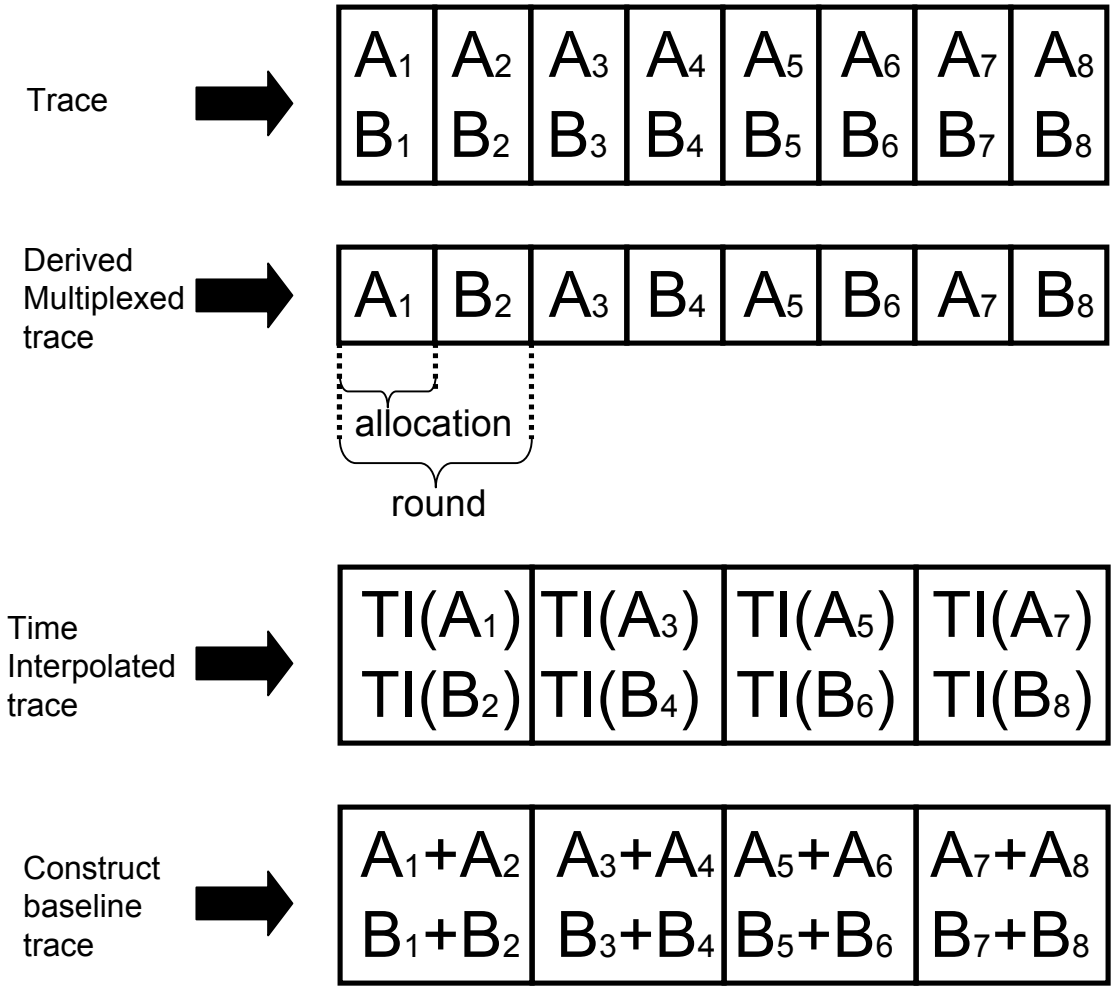
- Invented *correlation criterion* to evaluate different time interpolation techniques
- Used criterion to evaluate multiplexing and trace alignment

Evaluation Criterion

- How well does time interpolation preserve correlation between metrics?
- Compare correlation with and without time interpolation

- *Correlation Error* = | baseline – outer |
  - *baseline correlation*
    - metrics in same non-multiplexed trace
  - *outer correlation*
    - Trace alignment: metrics in different traces
    - Mulitplexing: collected metrics for only part of the time

Post-Mortem Multiplexing



Evaluation Technique

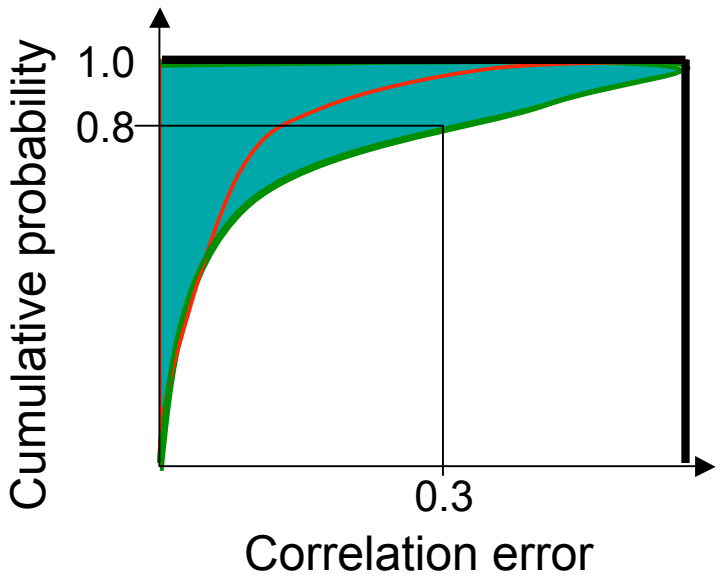
- Correlation error is a score for a pair of metrics.
- Collect 15 metrics, 15 choose 2 pairs or 105 scores!

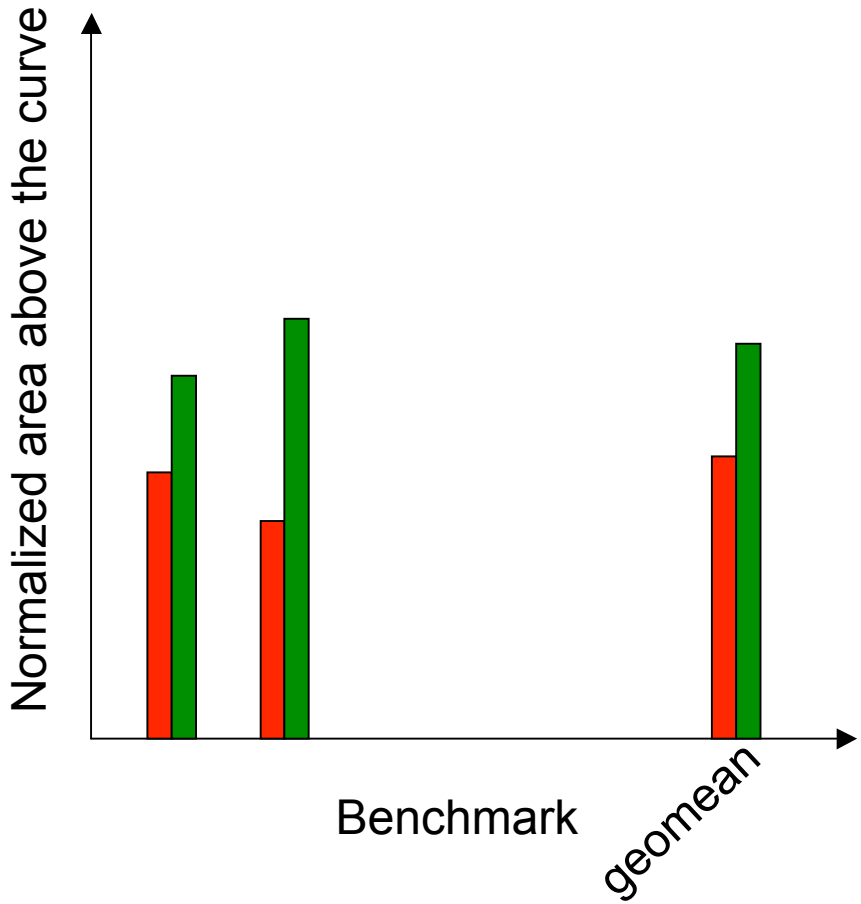
How to present this data?

- For one benchmark
- For all benchmarks

Presentation of Data

We use cumulative distribution functions to present our data





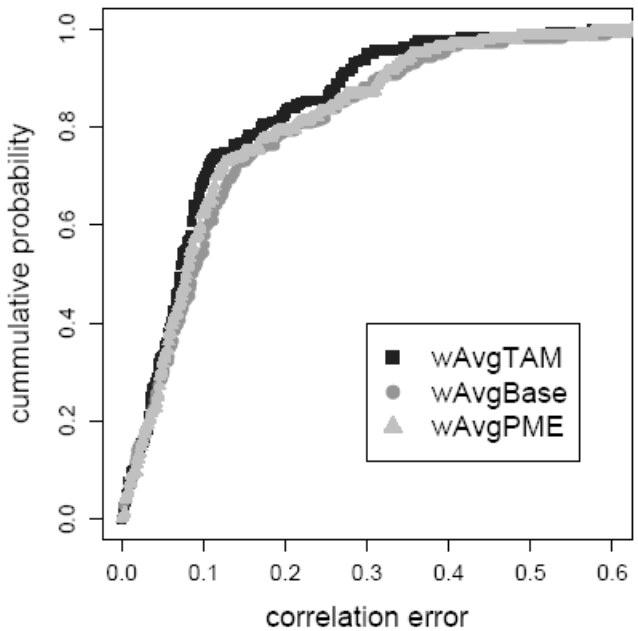
Experimental Methodology

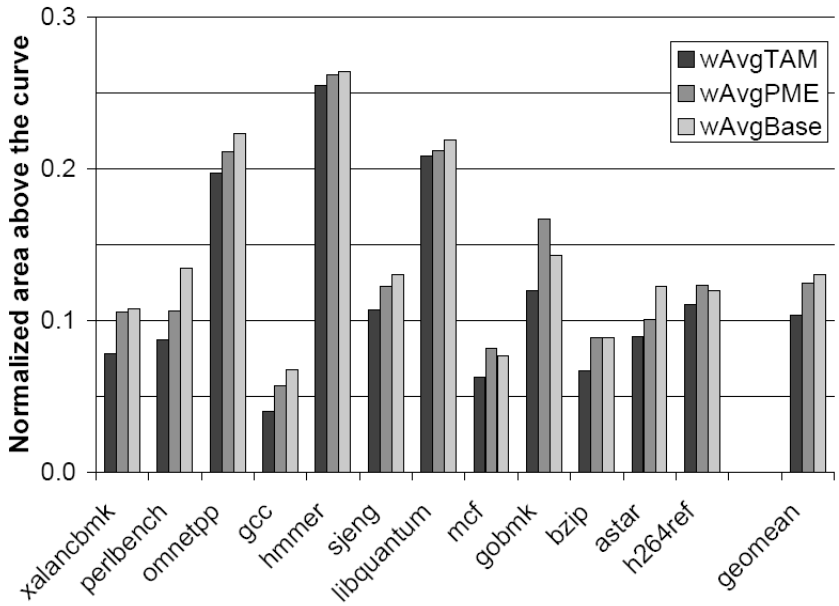
- Platform: 2.0 GHz Intel Pentium 4 & Linux
  - 18 HPM registers
- Trace Infrastructure
  - Built on top of PAPI
  - Generated record every 200,000 cycles
- Benchmarks
  - SPEC CPU2006 benchmarks
    - Use input that yielded runtime < 150 seconds.
- Metric selection
  - Collected 15 metrics
  - Varied hardware functionality
  - Spectrum of correlation scores

Multiplexing Variations
(Pentium 4, C Spec Benchmarks)

Configuration is same as Reza et al.: 10 allocations per round, 200K cycles per allocation

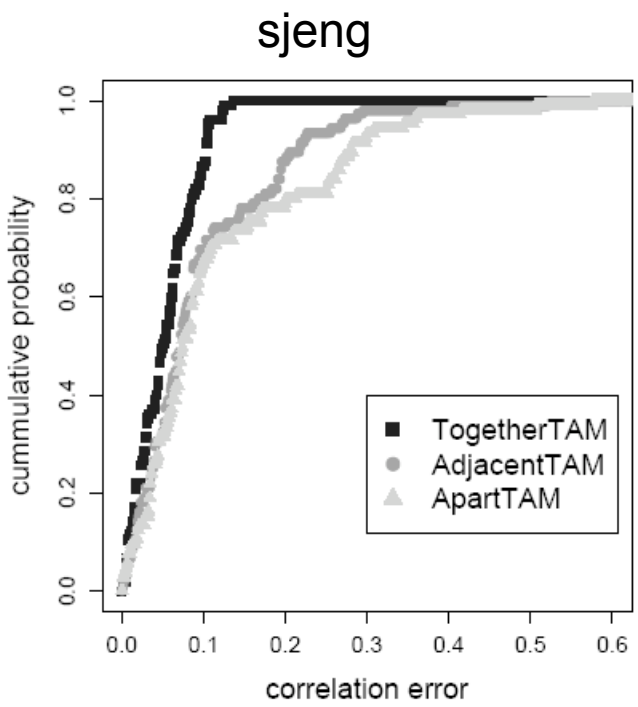
sjeng

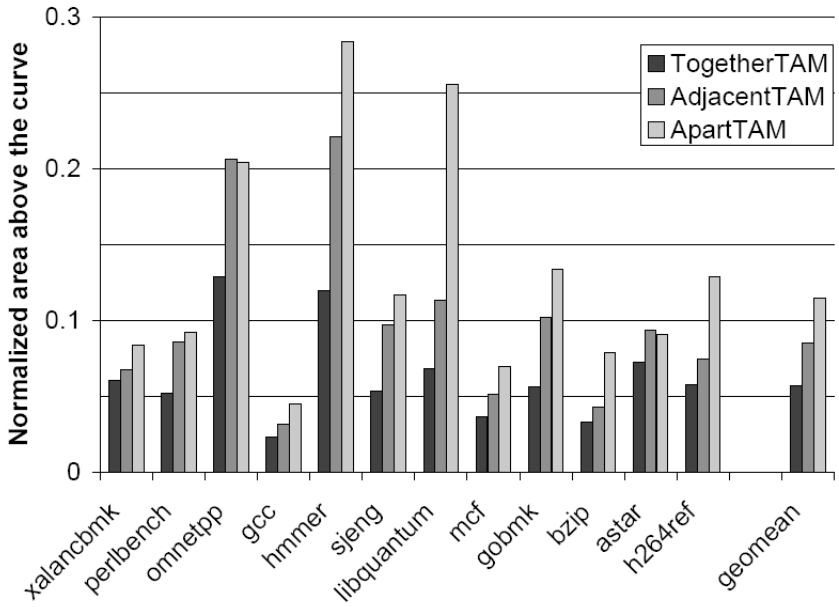




TAM performs best (but not great)

Trustworthiness of Multiplexing
(Pentium 4, C Spec Benchmarks)

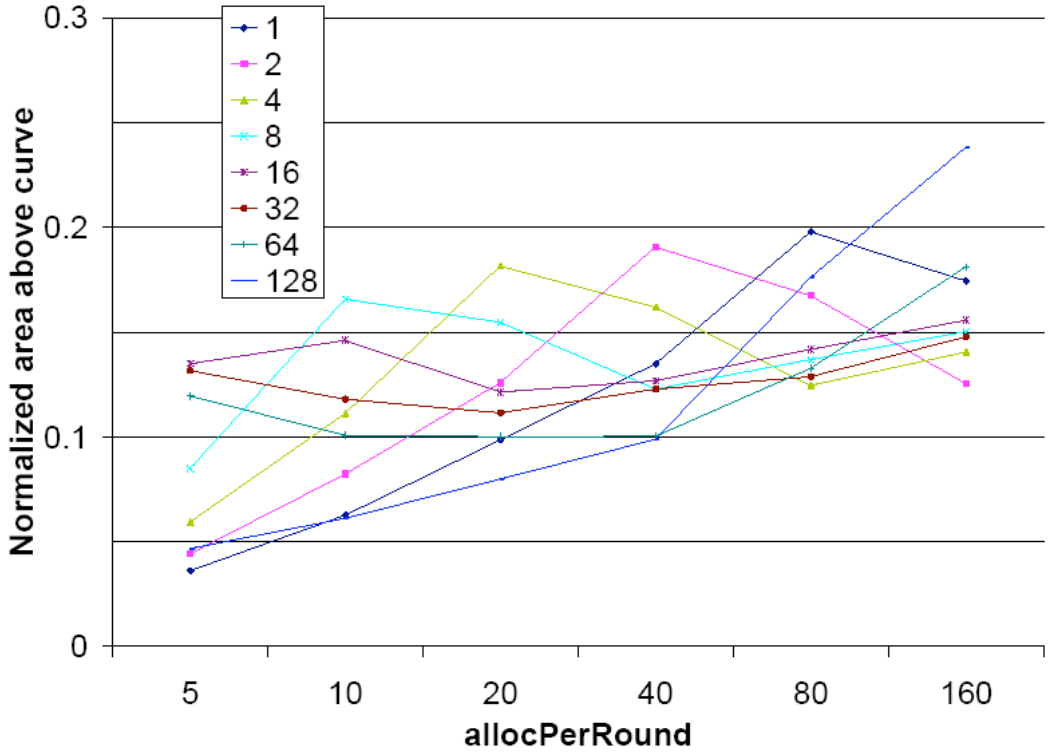




If the metrics of interest are in the same allocation then multiplexing is most likely trustworthy

Granularity and Multiplexing
(Pentium 4, C Spec Benchmarks)

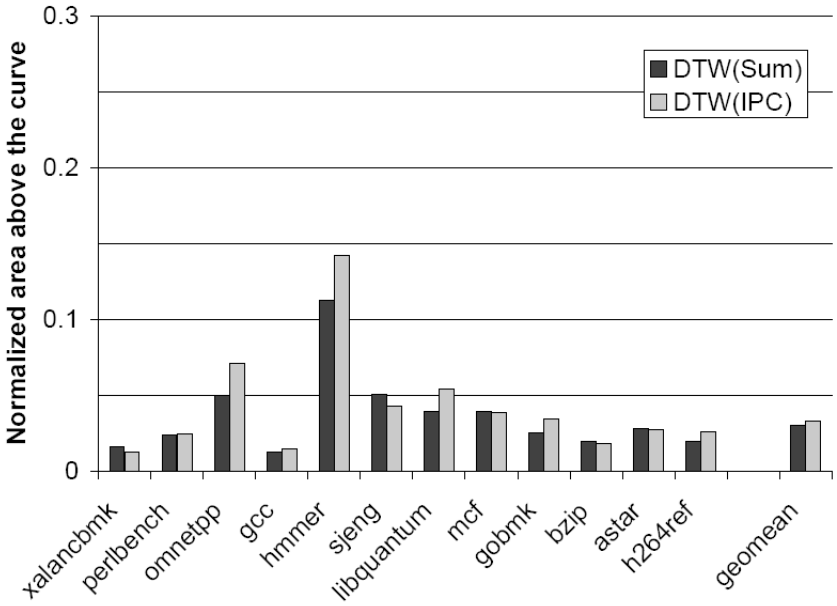
What happens if we collect more metrics?



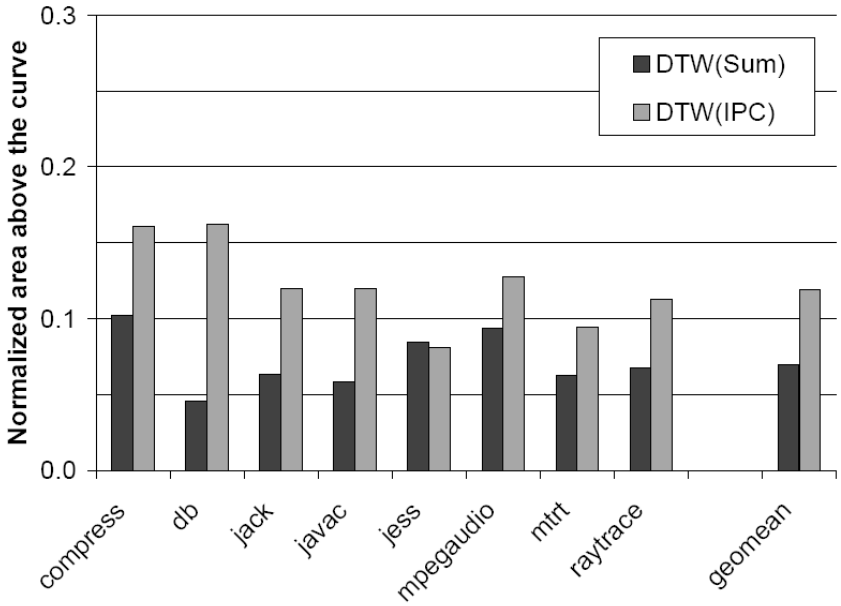
Multiplexing degrades as we collect more metrics

Trace Alignment Variations

C on Pentium 4

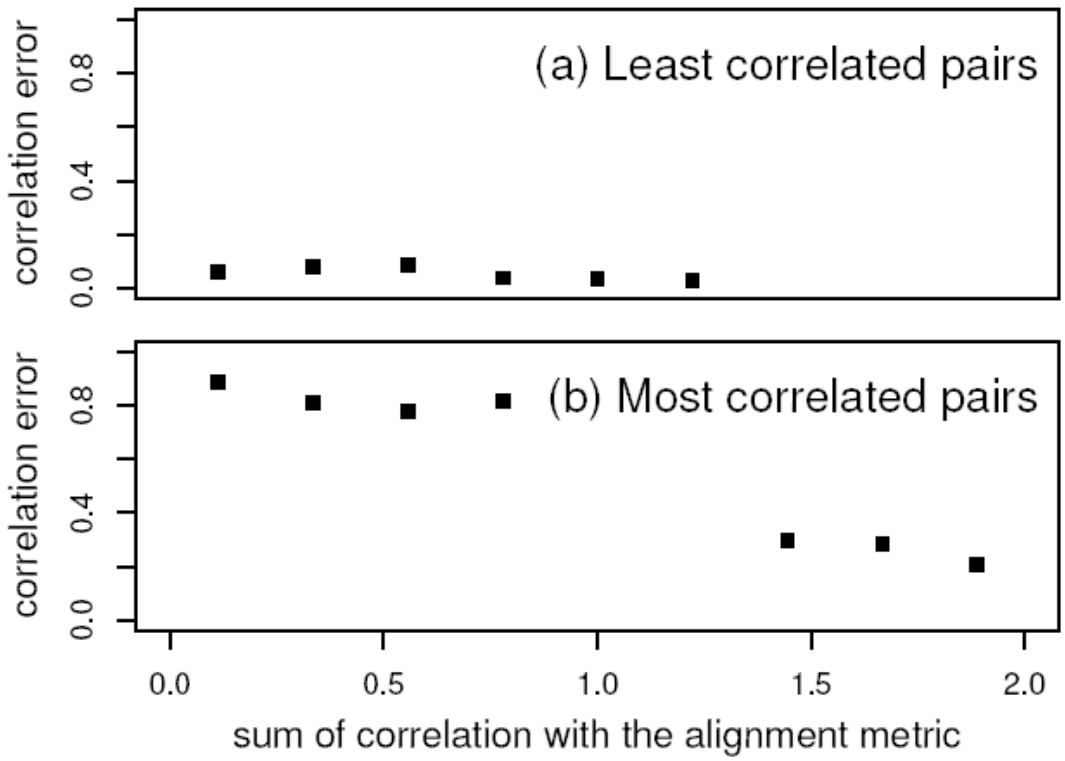


Java on Power4+



DTW(Sum) outperforms DTW(IPC) especially for Java

DTW: trustworthiness

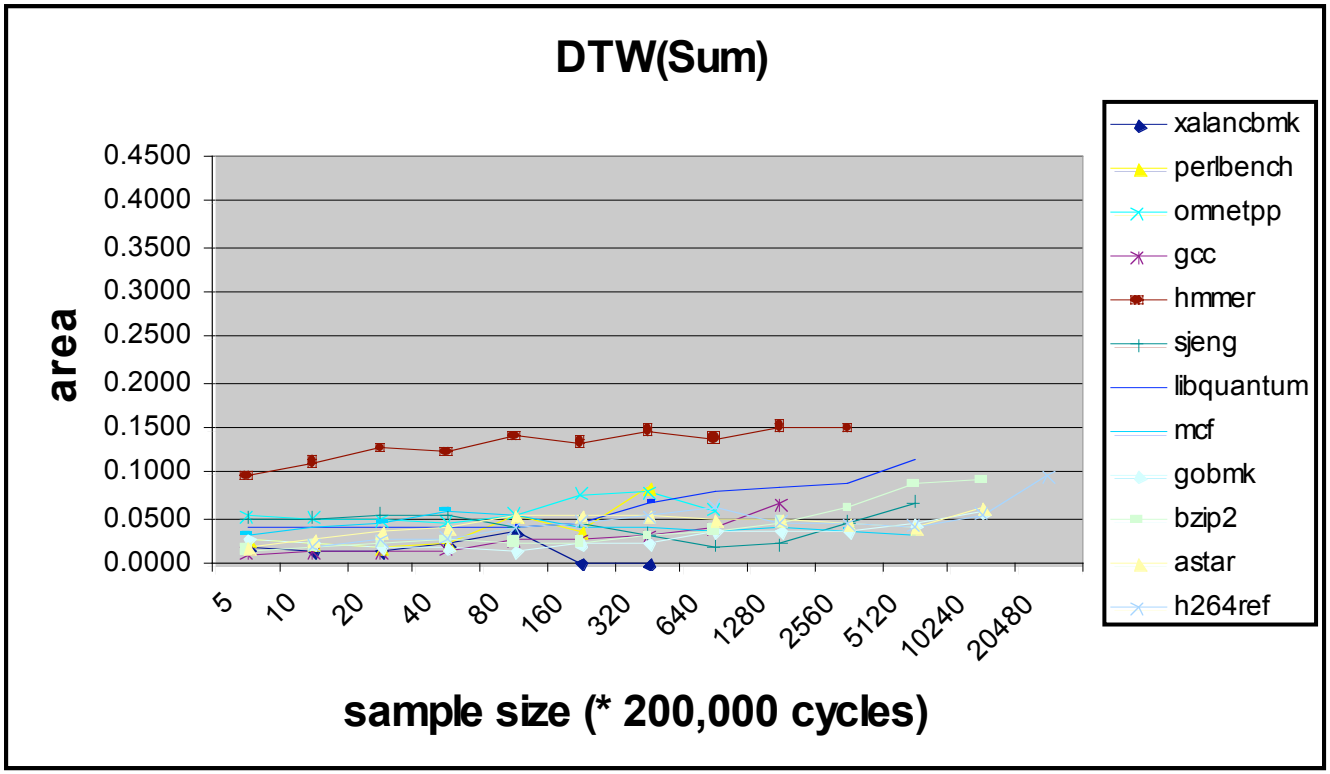


Low correlation
(838 triples)

High correlation
(386 triples)

If the metrics of interest and alignment metric are highly correlated then alignment is most likely trustworthy.

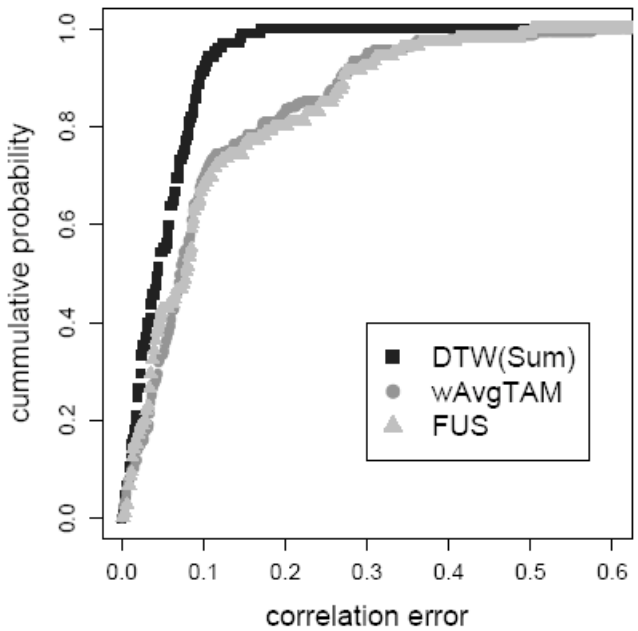
DTW: granularity

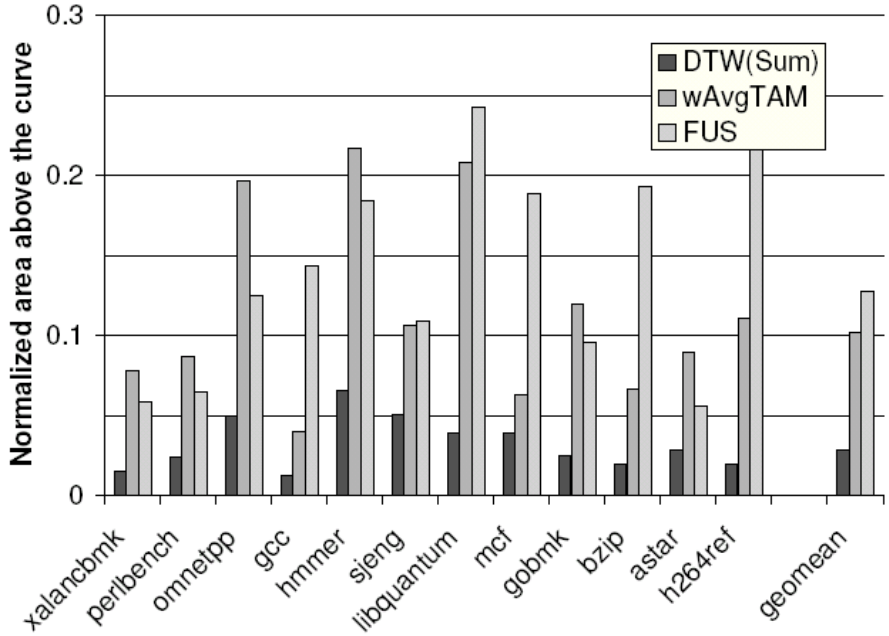


DTW(Sum) not effected by change in sample size.

The Bakeoff
(Pentium 4, C Spec Benchmarks)

Comparing trace alignment and multiplexing



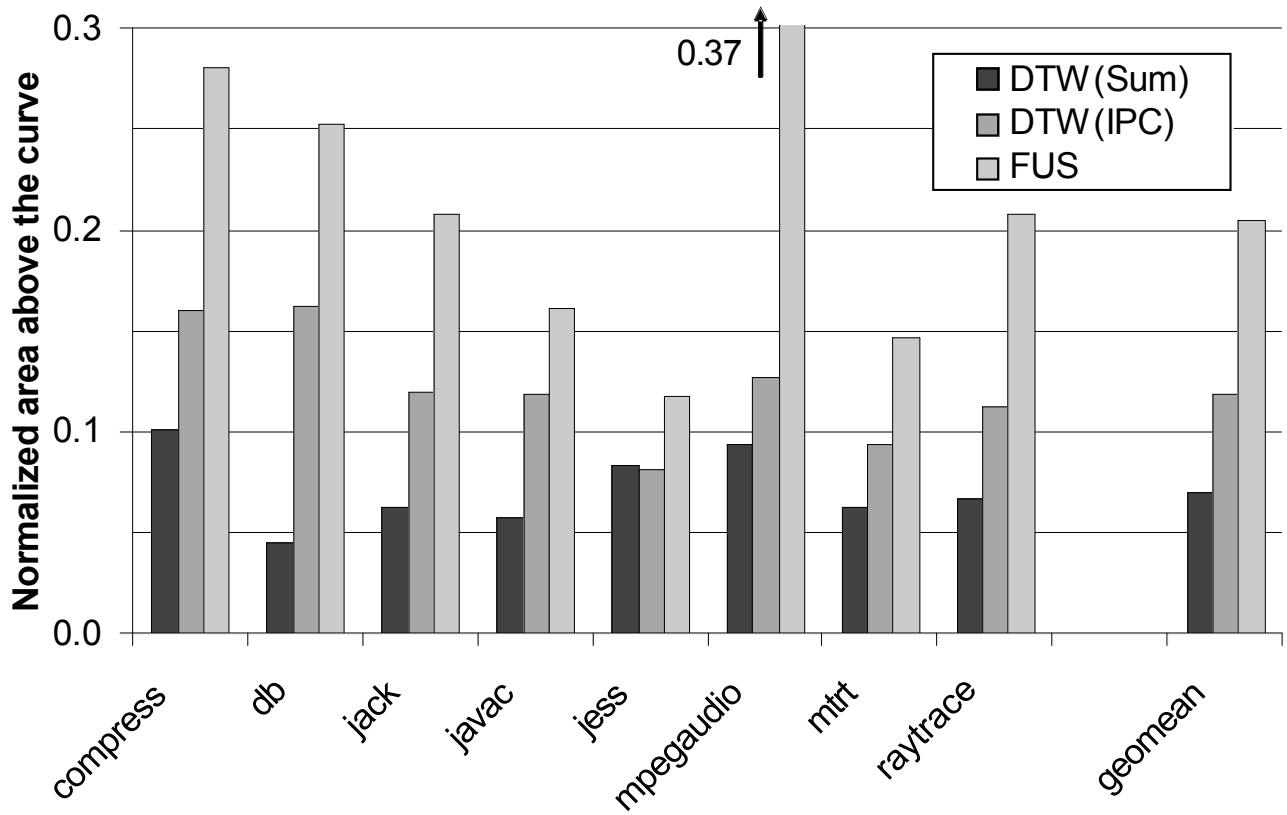


DTW(Sum) is the technique of choice

Why not always use DTW?

- Multiplexing advantages
  - Only one run
    - Handles non-deterministic programs
- DTW advantages
  - Better, more consistent results across different granularities
  - Less perturbation, samples once per round, whereas mulitplexing samples once per allocation

Beyond C
(Power4+/AIX, Java SPECjvm98 Jikes RVM)



Summary: DTW(Sum) effectiveness not limited to C, HPMs, or one target architecture. JVM introduces more correlation error.

Related Work

- Multiplexing hardware events
  - R. Azimi, M. Stumm, and R. W. Wisniewski. *Online performance analysis by statistical sampling of microprocessor performance counter*
  - Wiplove Mathur and Jeanine Cook. *Toward accurate performance evaluation using hardware counters*
- Dynamic Time warping
  - Hundreds of papers in speech recognition, time-series data mining, bioinformatics…
- Full Uniform Scaling
  - E. Keogh. *Efficiently finding arbitrarily scaled patterns in massive time series databases*

Conclusions

- Applications exhibit time-varying behavior
  - Need to reason about time-varying relations between metrics
  - Need time interpolation to reason about metrics collected at different times.
- Introduced novel evaluation technique
- Evaluated multiplexing and trace alignment
  - For each technique, evaluated multiple approaches
  - Explored sensitivity of parameter space
  - Multiple languages, platforms and OS
- DTW(SUM) performs well enough to enable reasoning across a large number of metrics

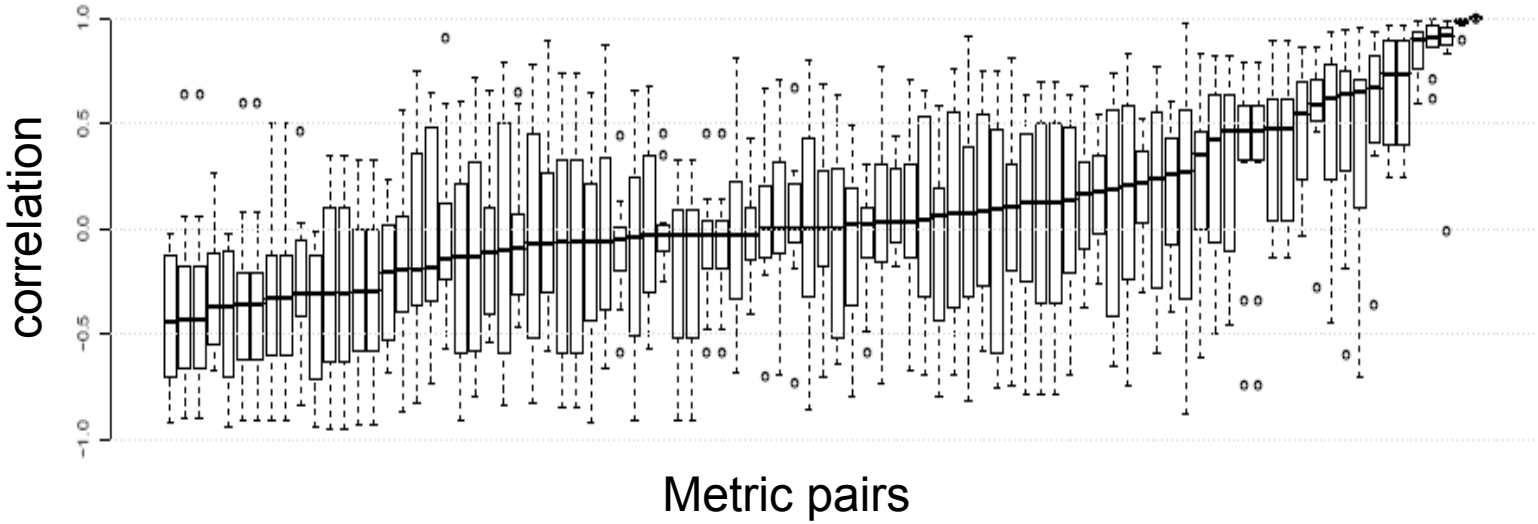
Questions

?

Milestone-based Alignment

- Why not use regular deterministic events (milestones) to align two traces?
  - Align the nth occurrence in one trace with the nth occurrence in another trace
- Hard to find good milestones
  - Uniformly and frequently distributed,
    - Otherwise requires alignment between them
- Java applications
  - Although application thread deterministic, JVM threads not, but interested in application and JVM interaction.
- Deterministic C programs (no runtime!)
  - Use instruction count as milestone!
  - Sample on every n^{th} instructions completed.
  - As good as DTW(Sum) when applicable.

Correlation Scores



Box plot of correlation scores for all pairs of metrics across all benchmarks

- Each column is a pair of metrics for all benchmarks
- Bold line in middle is median
- Large spectrum of correlation scores: good basis for evaluation