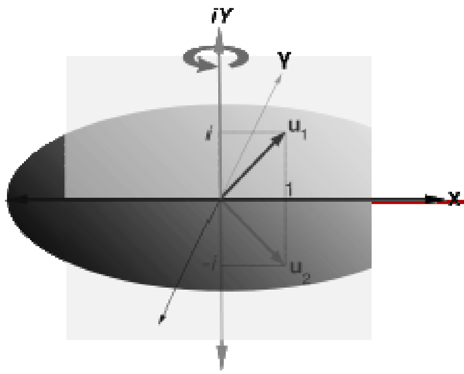


EigenBench: A Simple Exploration Tool for Orthogonal TM Characteristics

Pervasive Parallelism Laboratory, Stanford University

Sungpack Hong
Tayo Oguntebi
Jared Casper
Nathan Bronson
Christos Kozyrakis
Kunle Olukotun

IISWC'10, 2010-Dec-02

Outline

- Yet Another Benchmark for TM?
- Orthogonal Characteristics
- EigenBench
- Orthogonal Analysis
- Application Behavior

TM Benchmarks

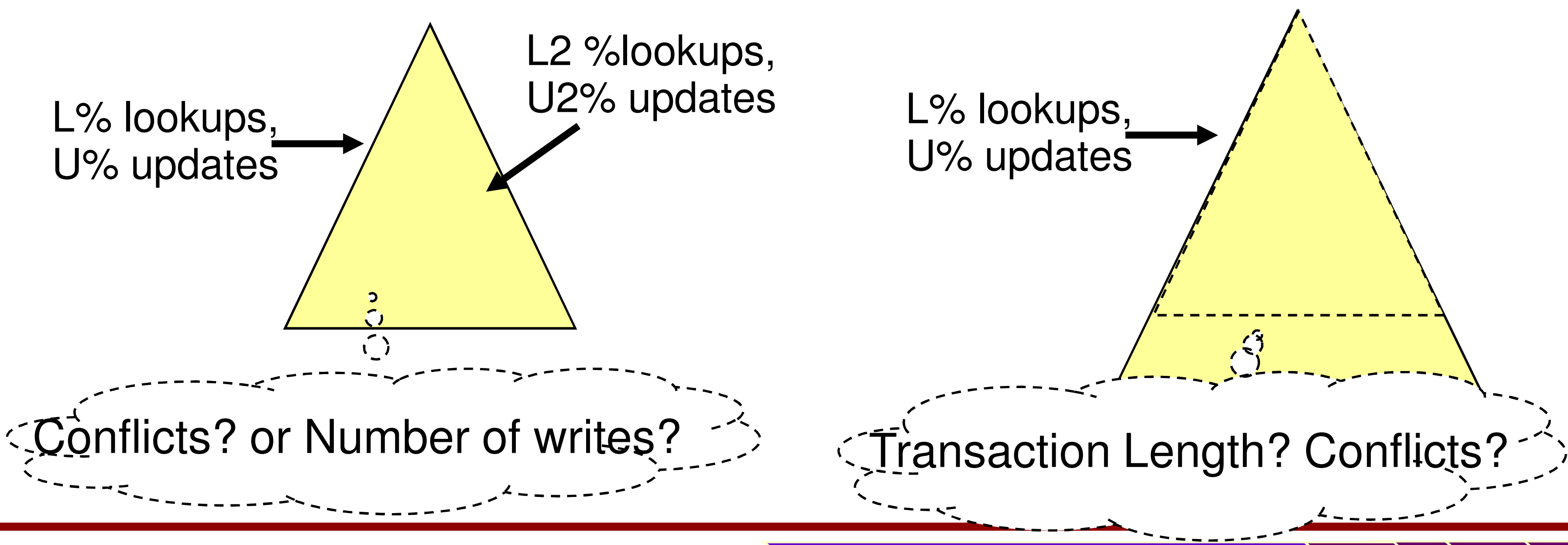
- Transactional Memory (TM)
  - Significant number of TM proposals
    : Hardware TM, Software TM, Hybrid TM …
  - How do we evaluate them?
- Conventional TM Benchmarks
  - Application benchmark (STAMP, …) [Cao Minh et al, IISWC'08]
    - Realistic
  - Synthetic benchmarks (STMBench7, …) [Guerraoui et al, Eurosys'07]
    - Easy to configure and parametrize.
    - Do they reflect realistic application behavior?

    (e.g.) SwissTM outperformed TL2 2x~5x in synthetic bench, but only 20~90% in STAMPs. [Dragojevic et al, PLDI'09]

Conventional Synthetic Benchmarks

- Synthetic Benchmarks (cont'd)
  - Typically based on shared data-structure access (e.g. red-black tree)
  - Degree of freedom for exploration?

(Example)



Knobs Wanted

- Want to observe each TM characteristics, separately
- …. But what are the TM characteristics?
  - People mean different things with one term.

ex> “Large Transactions”

➔ Many TX reads & writes? (STM barrier overhead)

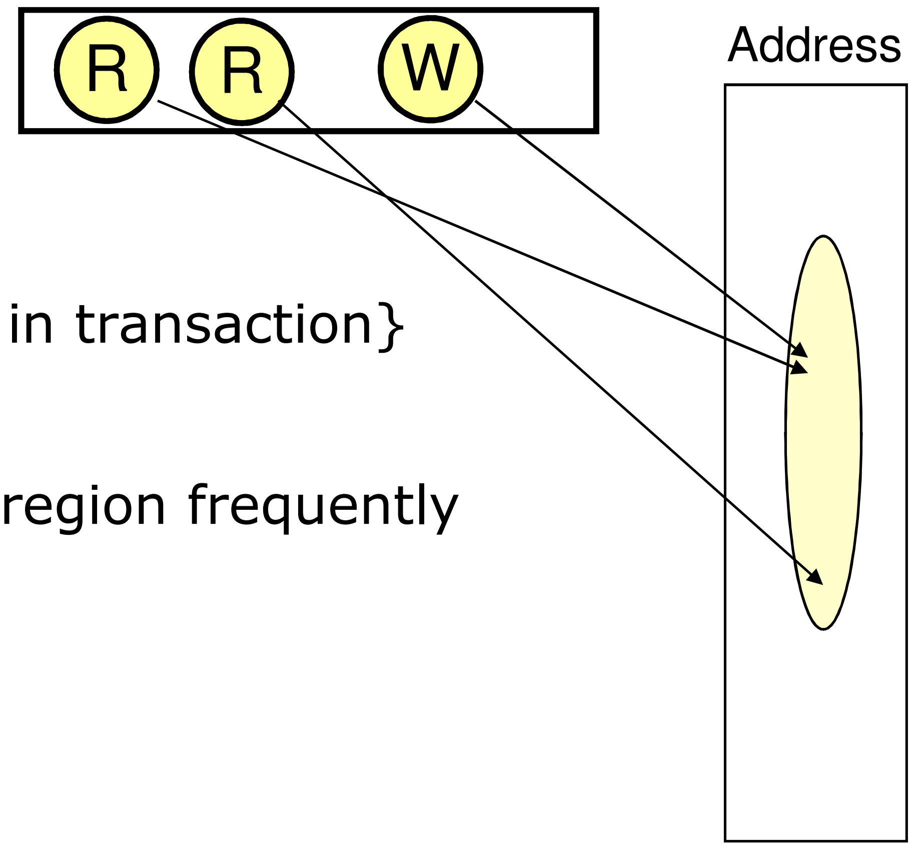
➔ Many different addresses? (HTM overflow)

➔ Many (non-tm) instructions inside TX? (rollback overhead)

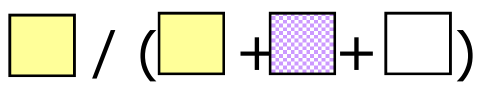
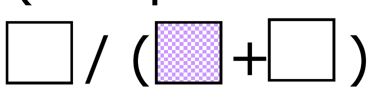
- We propose eight *orthogonal* TM characteristics.

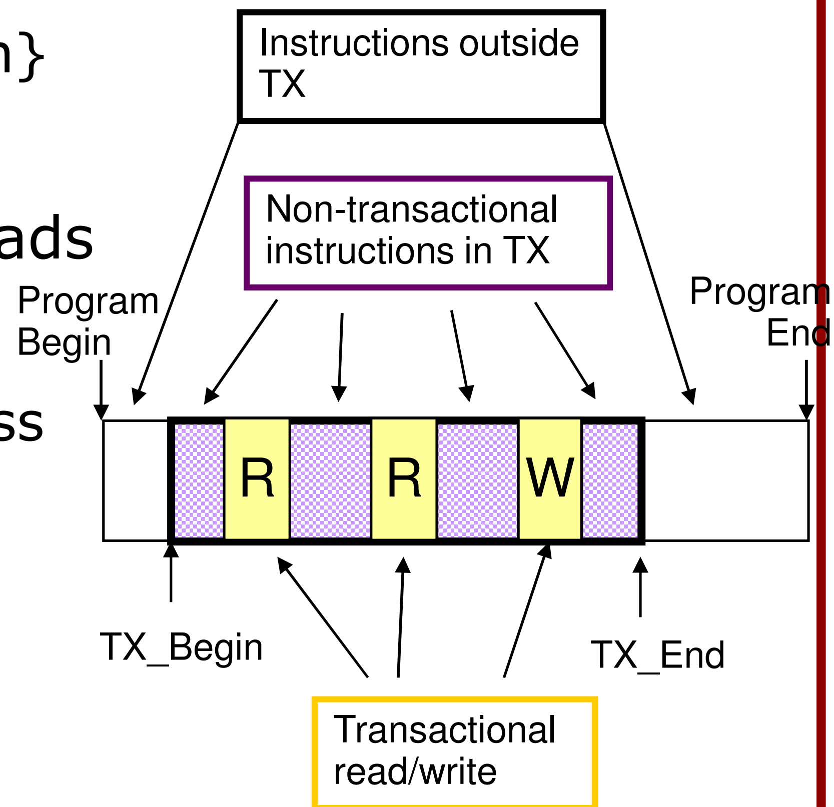
TM Characteristics (1/2)

- Translation Length
  - Number of *Transactional* read,write
- Pollution (0.0 ~ 1.0)
  - (WR) / (RD + WR)
- Locality (0.0 ~ 1.0)
  - Prob {Repeated Address in transaction}
- Working-Set Size
  - Size of memory address region frequently used in application



TM Characteristics (2/2)

- Contention (0.0 ~ 1.0)
  - Prob {Conflict of a transaction}
- Concurrency
  - Number of concurrent threads
- Predominance (0.0 ~ 1.0)
  - Fraction of *transactional* access
  - 
- Density (0.0 ~ 1.0)
  - Fraction of non-tm instr in TX (complementary)
  - 



How do characteristics affect performance?

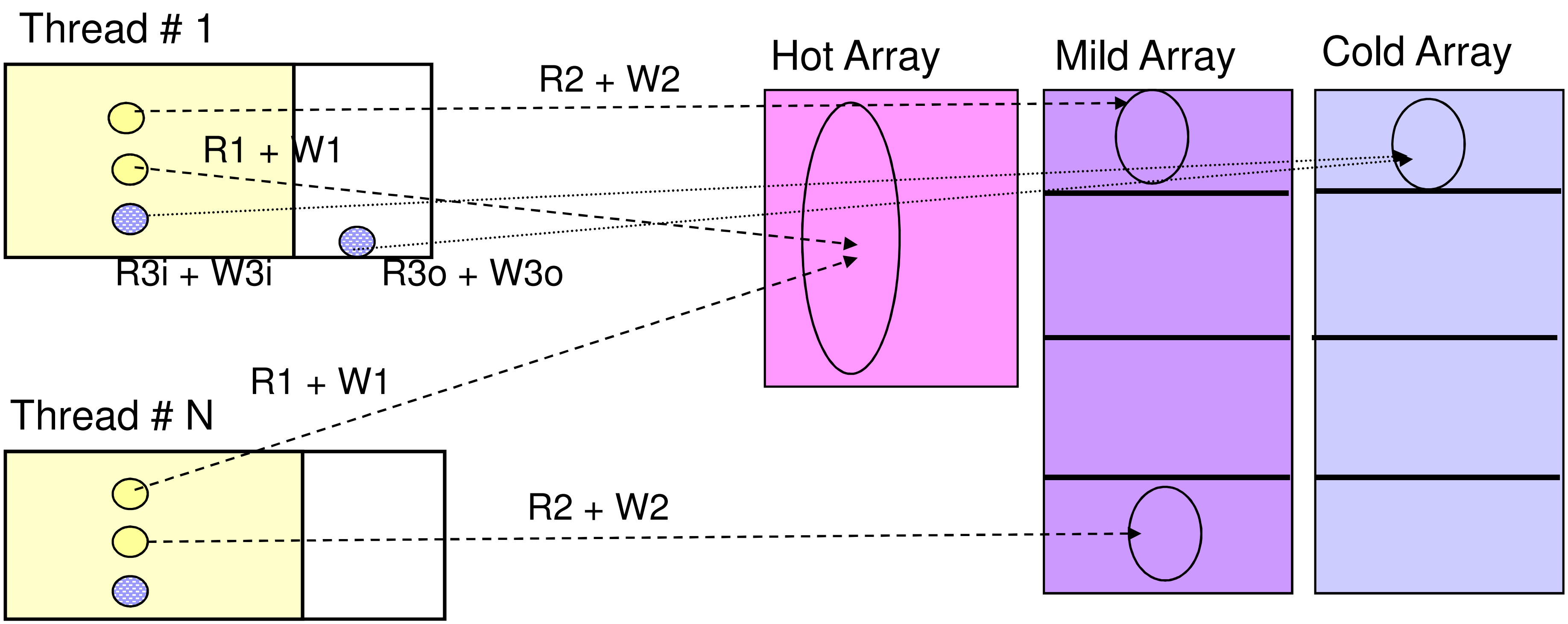
	HTM	STM
Tx Length	Overflow	TX-Barrier overhead
Pollution	Overflow ; conflict detection	Write-buffer manage; conflict detection
Locality	Overflow	Write-buffer searching
Working-Set Size	Conflict detection; cache miss latency	
Conflict	Conflict detection	
Concurrency	Scalability	
Density	Cost of re-execution	
Predom.	TM impact on overall performance	

Orthogonal Characteristics

(*) Write-set size = (TX Length) * (Pollution) * (1 - Locality)

EigenBench

- How to explore each characteristic one by one?
- EigenBench – a simple exploration tool



EigenBench (Cont'd)

- Implementation is very simple (randomized memory accesses)
- EigenBench can induce each TM characteristic orthogonally.

Characteristic	Eigenbench Parameters	Characteristic	Eigenbench Parameters
Concurrency	N	Working-set size	$A_1 + A_2 + A3$
Transaction length	$R_1 + R_2 + W_1 + W_2 = (T_{len})$	Pollution	$(W_1 + W_2)/T_{len}$
Temporal locality	lct	Contention	*see Equation (1)*
Predominance	$T_{len} * \alpha / (T_{len} * \alpha + C_{in} + C_{out})$	Density	$C_{out}/(C_{in} + C_{out})$
Read set Size*	$(R_1 + R_2) * (1 - lct)$	Write set Size*	$(W_1 + W_2) * (1 - lct)$

$N_{in} = ((R_{3i} + W_{3i}) * \alpha + Nop_i) * T_{len}/K_i, \quad O_{in} = \beta * T_{len} * (1 + (R_{3i} + W_{3i})/K_i), \quad C_{in} = N_{in} + O_{in}$

$N_{out} = ((R_{3o} + W_{3o}) * \alpha + Nop_o)/K_o, \quad O_{out} = \beta * (R_{3o} + W_{3o})/K_o, \quad C_{out} = N_{out} + O_{out}$

α: the average memory access latency β: overhead of random address generation and action decision in CPU cycles†

Detailed explanation available in the paper

Orthogonal Analysis: How-to

- Our approach
  - Start from a *typical* transaction; explore each characteristic.
  - Non-conflicting transactions ➔ *overhead*
  - Conflicting transactions ➔ *detection precision*
- Example Analysis
  - TL2 vs. SwissTM
  - Default Transaction;

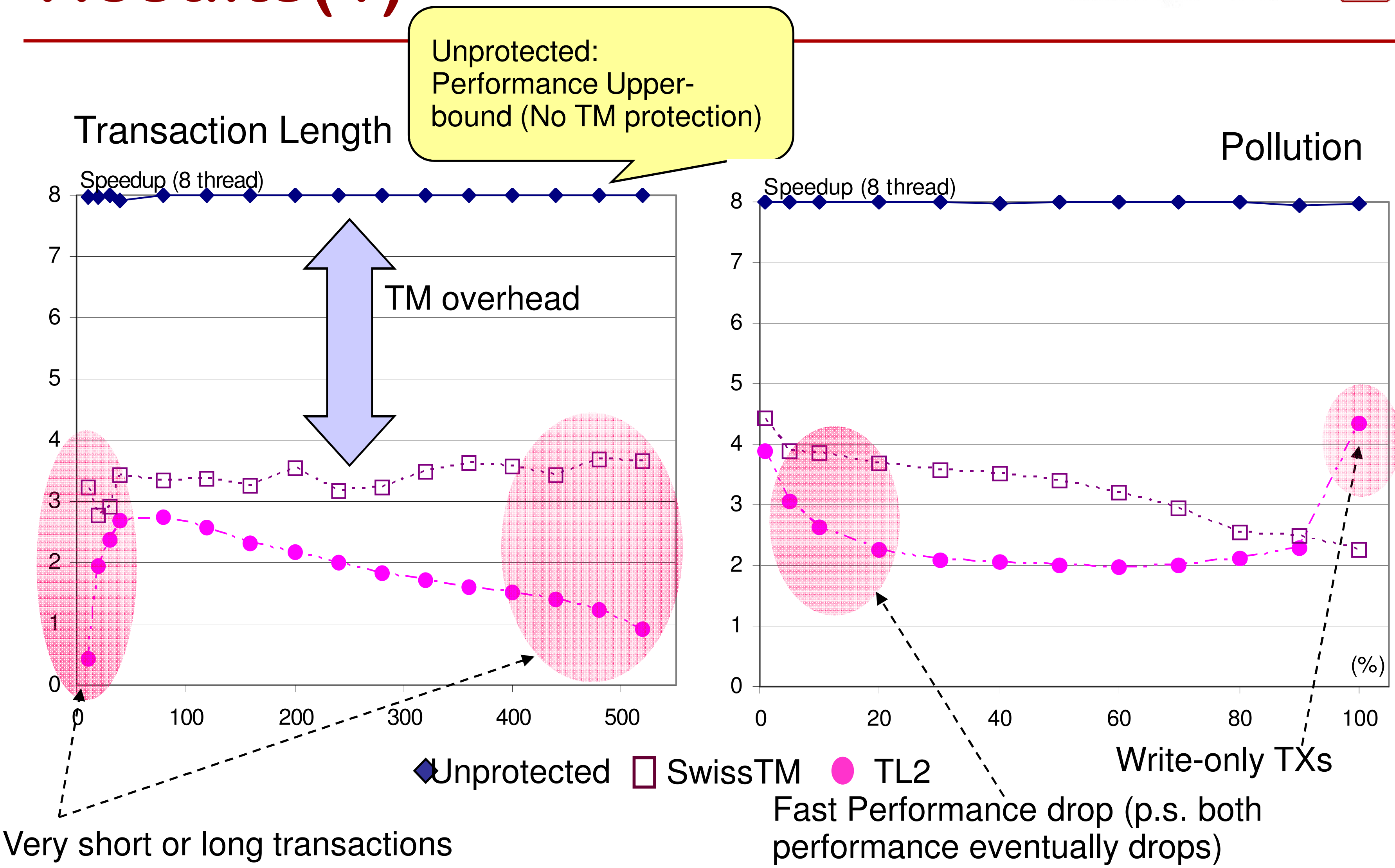
    Length:100, Pollution:0.1,

    Conflict: 0.0, Working-set:256kB (per thread),

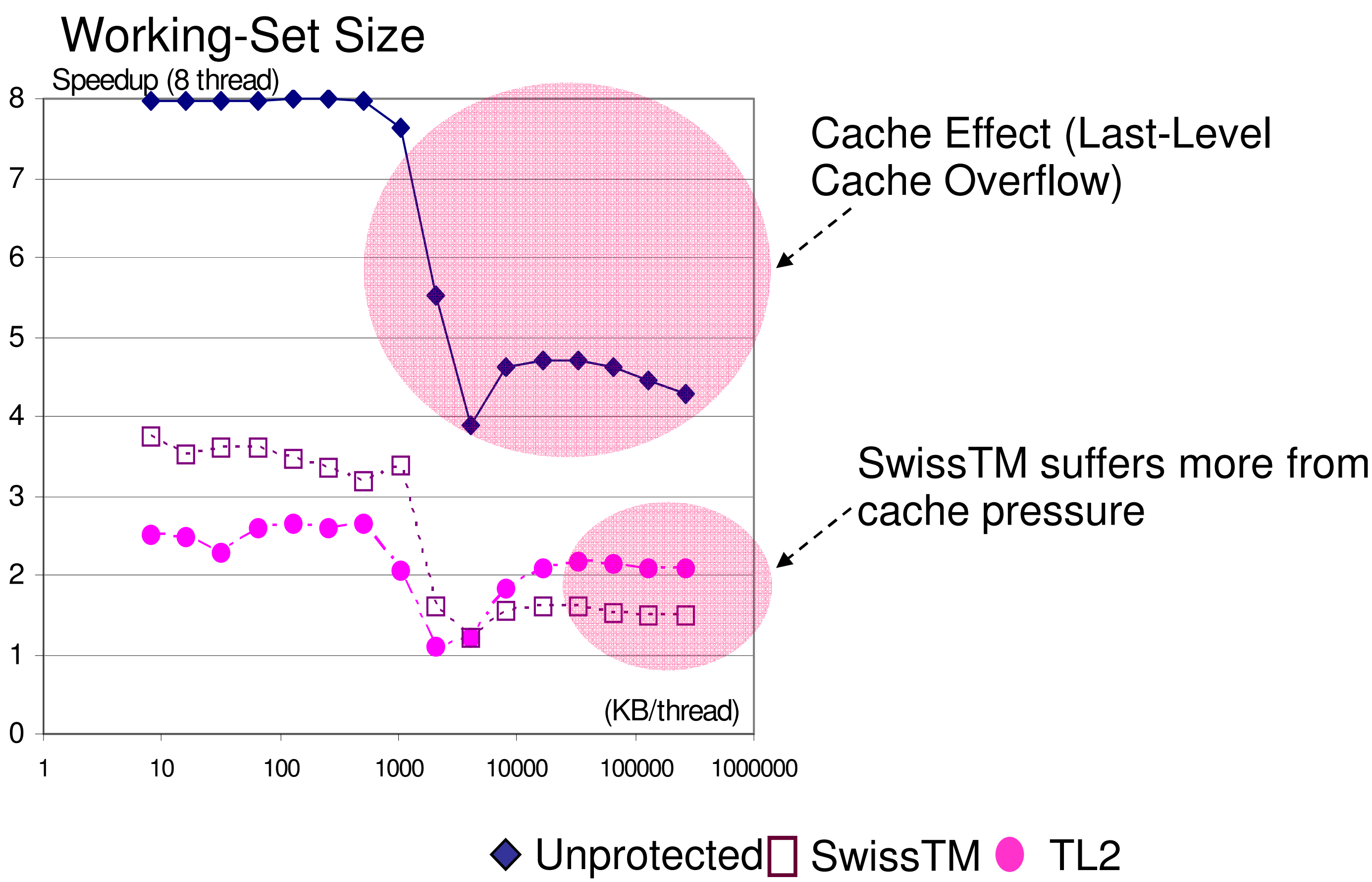
    Locality:0.0, Predom:1.0,

    Density:1.0, Concurrency: 8

Orthogonal Analysis: Results(1)

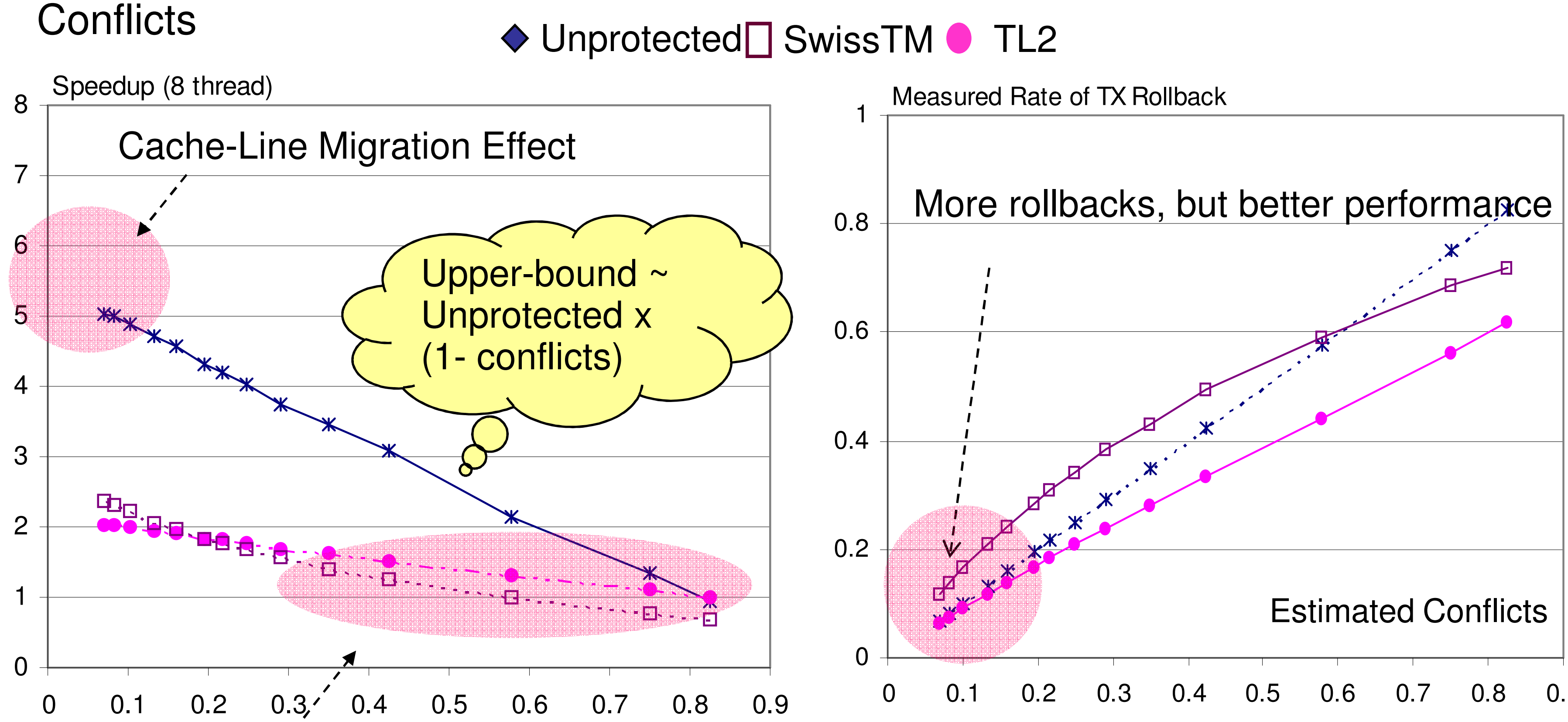


Orthogonal Analysis: Results(2)



Orthogonal Analysis: Results(3)

Conflicts

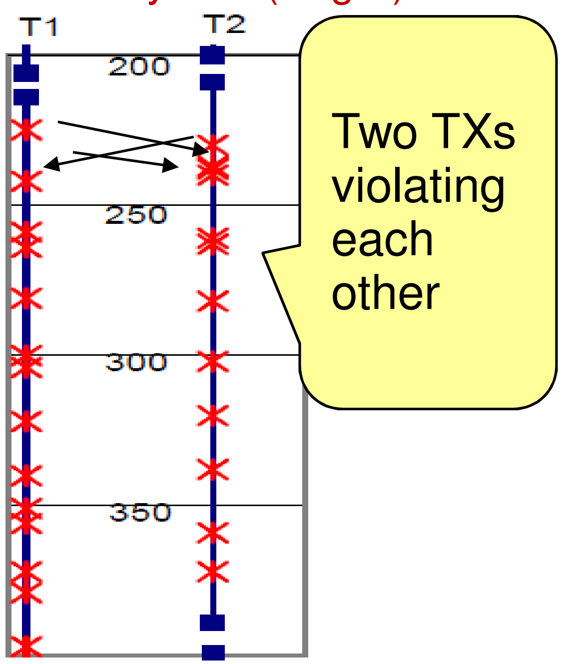


$$P_{conf} = 1 - \left(1 - \min\left\{1, \frac{(N-1)W_1'(1-lct)}{A_1}\right\}\right)^{W_1'+R_1'} \quad (1)$$

Pathology Generation

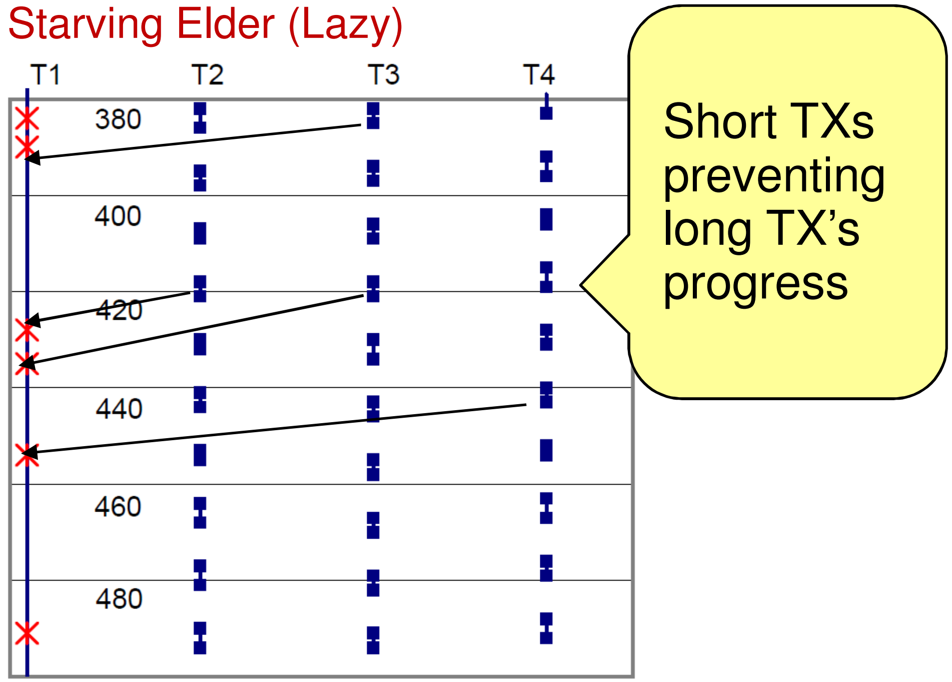
- TM Pathology [Bobba et al, ISCA 2007]
  - memory access patterns causing low performance
  - Can we generate pathologies from EigenBench? Yes

Friendly Fire (Eager)



TX trace via timestamp

Starving Elder (Lazy)



TX trace via timestamp

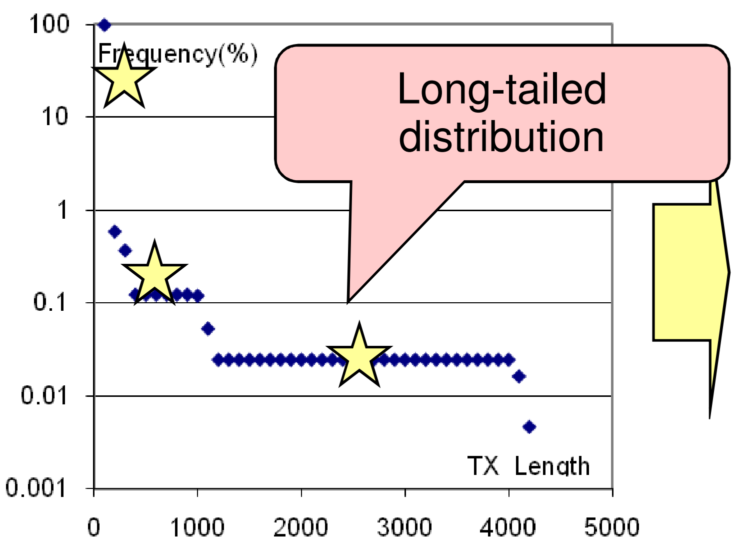
Application Characteristics

- Questions
  - What are TM characteristics of real applications?
  - Can we explain application performance via TM characteristics?
- Example Study: STAMP applications mimicry
  - To demonstrate relationship between characteristics and application performance
  - Instrumentation/Profiling ➔ statistics for TM characterisitcs ➔ Replay with EigenBench

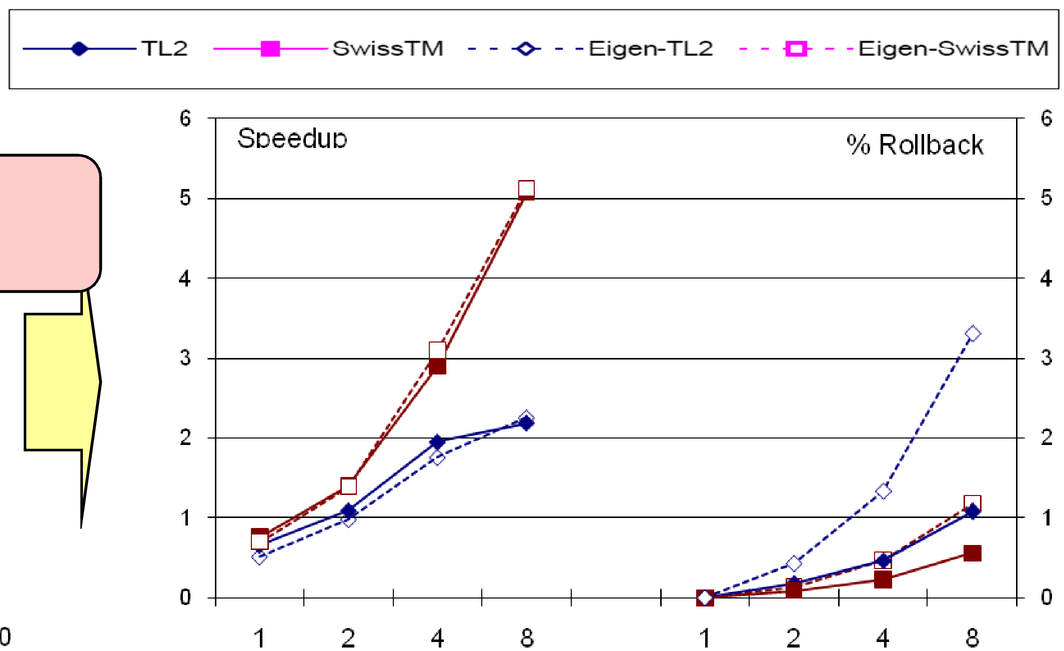
STAMP Application (1)

- Observations
  - Different *distributions* of characteristics
  - Single average may not be enough ➔ Mix of discrete characteristics.

Genome

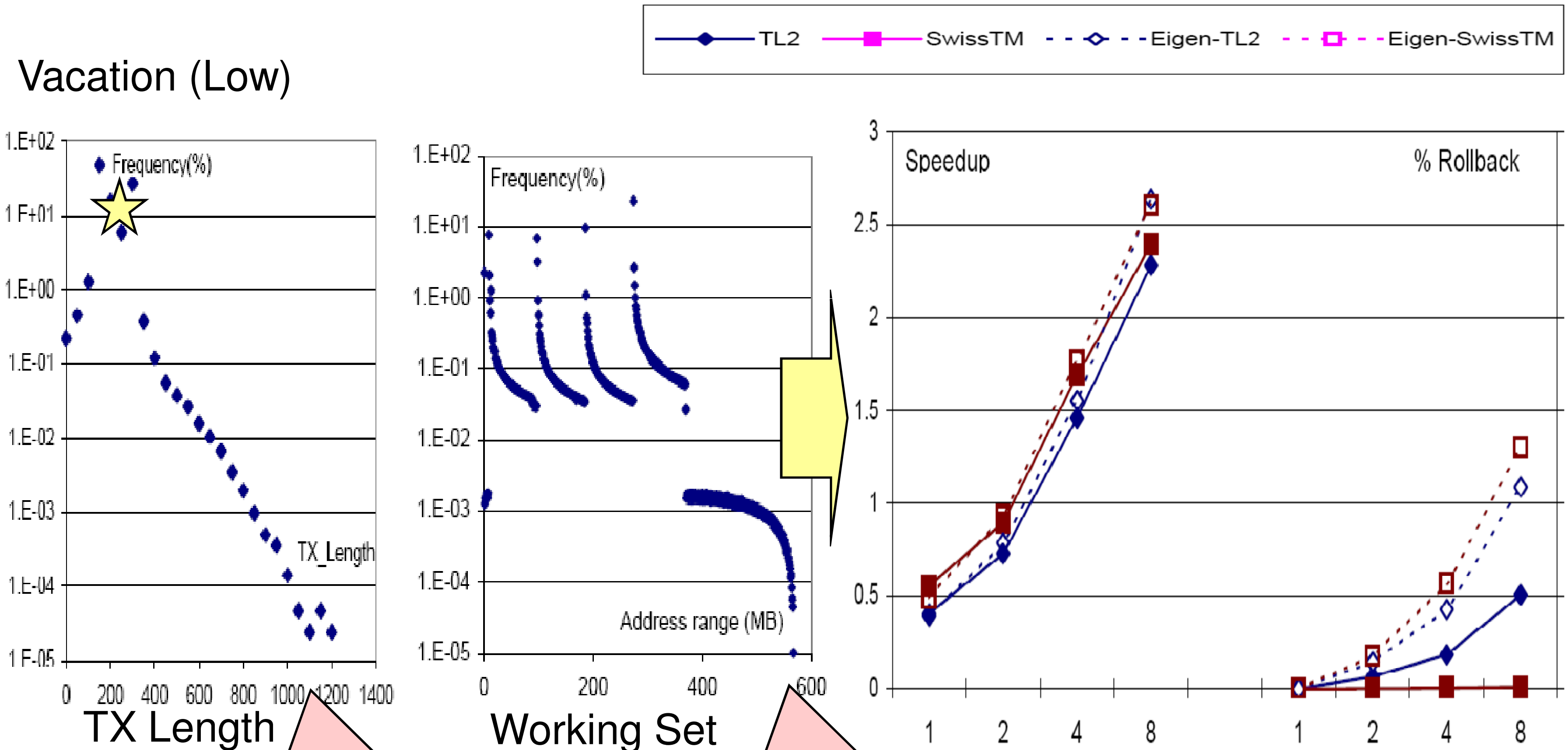


TX Length



STAMP Application (2)

Vacation (Low)

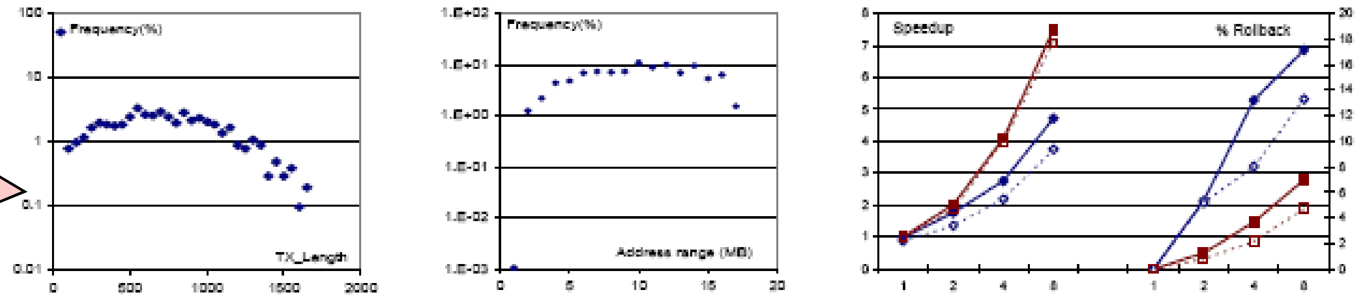


Normal distribution

Wide memory access

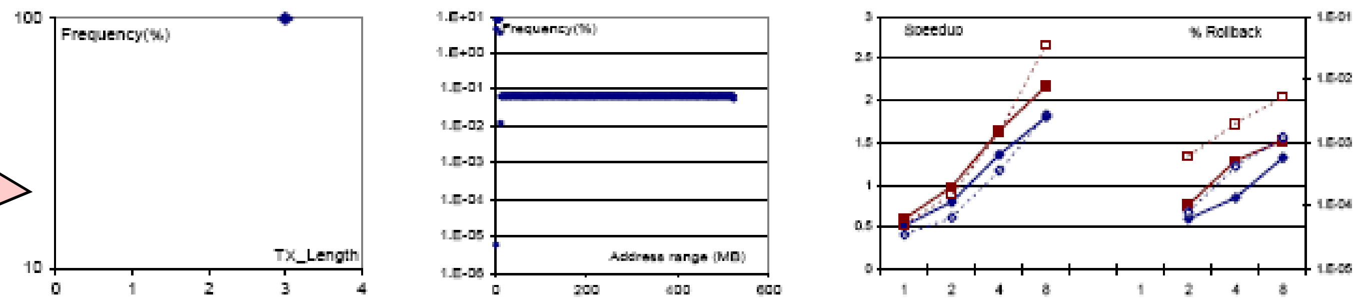
STAMP Application (3)

Long TX
Low Density



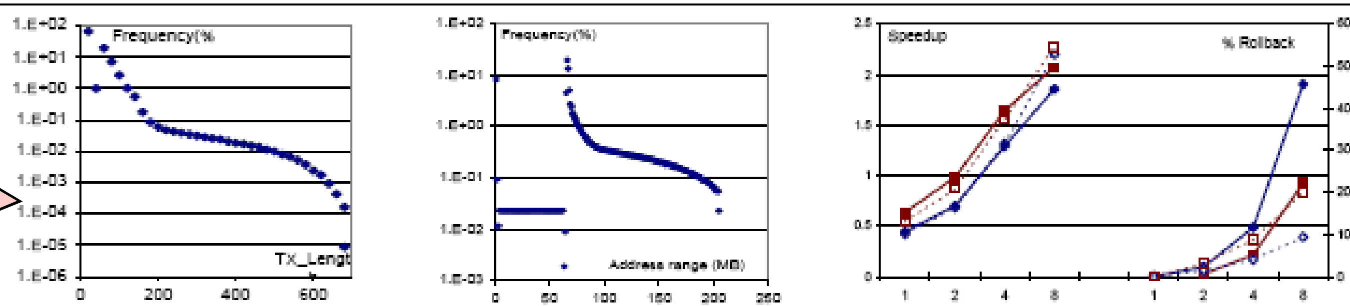
(c) Labyrinth

Short TX
Large
Working-Set



(d) SSCA2

Short TX
High Conflicts



(e) Intruder

EigenBench Use-cases

- How to use EigenBench?
  1. Orthogonal Analysis

     : Length, Working-Set, Pollution, Conflicts, Concurrency, (locality, density, predom)
     - Non-conflicting
     - Conflicting
  2. Explain application behavior
  3. (Optional) Check if it can survive pathologies.

Summary

- Orthogonal TM Characteristics
- EigenBench
- Orthogonal Analysis
- Application Performance Explanation

- Subsidiary Lessons for STM designers
  - Cache effect should be considered
  - Trade-off barrier overhead vs. conflict resolution
    - Restart penalty can be small

- Download: http://ppl.stanford.edu/eigenbench
- E-mail: eigenbench_manager@lists.stanford.edu