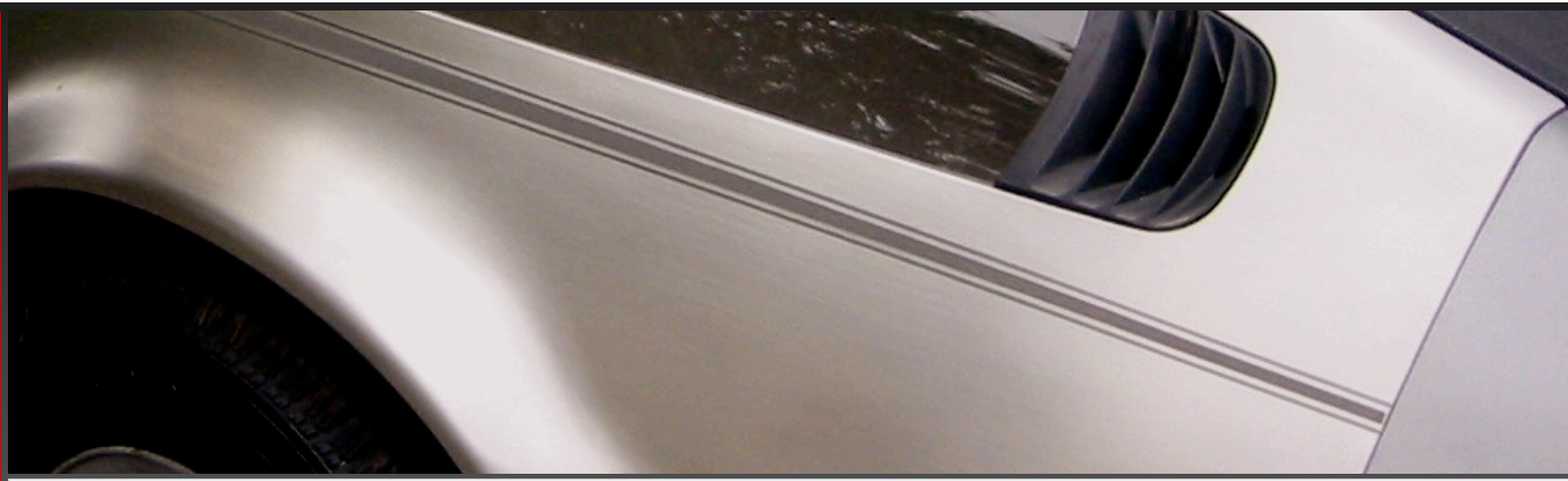


DeLorean:

Recording and Deterministically Replaying Shared Memory Multiprocessor Execution Efficiently

Pablo Montesinos, Luis Ceze* and Josep Torrellas

Department of Computer Science
University of Illinois at Urbana-Champaign

http://iacoma.cs.uiuc.edu

*Department of Computer Science and Engineering
University of Washington

Motivation

- Debugging parallel applications on a multicore is challenging:
  - Bugs can be very timing dependent
  - Their effects might manifest after many instructions
- Need effective debugging techniques for parallel applications on MPs
- One such technique: HW-assisted deterministic replay

HW-Assisted Deterministic Replay

Bacon and Goldstein. WPDD 91

- **Phase I: Initial execution**

- **Phase II: Replay**

Contributions

- Propose **DeLorean**: a new scheme for hw-assisted deterministic replay of parallel programs based on “chunk-based” execution
  - Records initial execution at production-run speeds
  - Requires only small log (0.6% of current schemes)
  - Replays at high-speed
  - Has multiple modes of execution with different speed/logging requirements

Outline

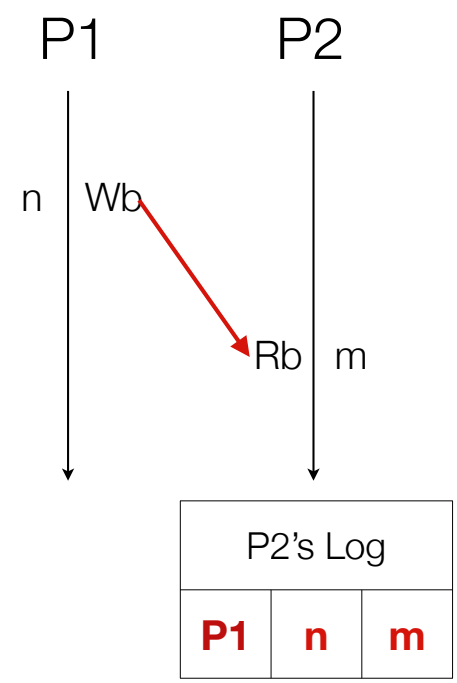
- Background on replay schemes
- DeLorean: a Chunk-based replay scheme
- Advanced DeLorean Execution
- DeLorean Replay
- Evaluation

DeLorean

Point-to-Point Schemes: FDR and RTR

FDR (*ISCA 03*)
RTR (*ASPLOS 06*)

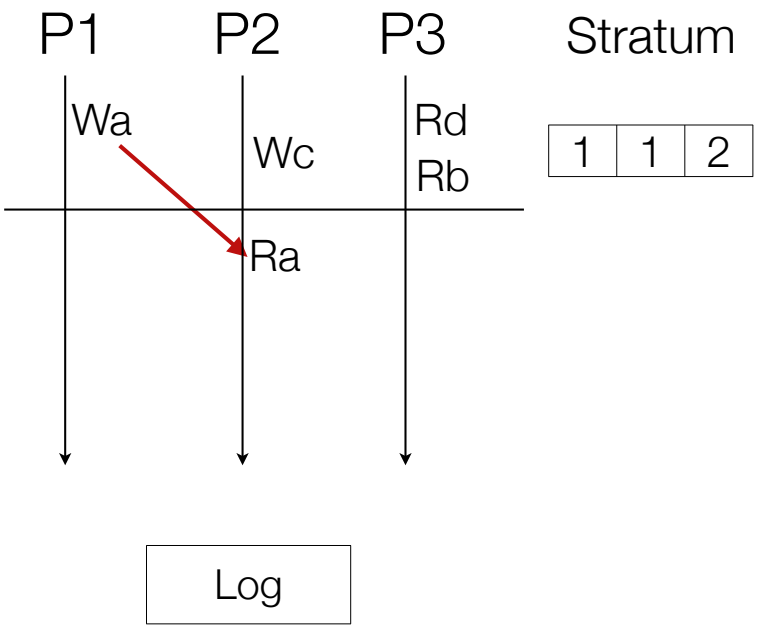
- Record dependences between *individual instructions* of different threads
- Optimizations to reduce log size
  - Transitive reduction
  - Regulated transitive reduction
- Use Sequential Consistency (SC)
- RTR proposes an algorithm to support TSO:
  - Record value of loads that violate SC



Vector-based Schemes: Strata

Strata (*ASPLOS 06*)

- Stratum: vector of as many counters as processors:
  - Count # of memory operations executed
- Log stores sequence of strata
- It also uses SC
- Log size is similar to RTR's



Episode-Based Schemes: ReRun

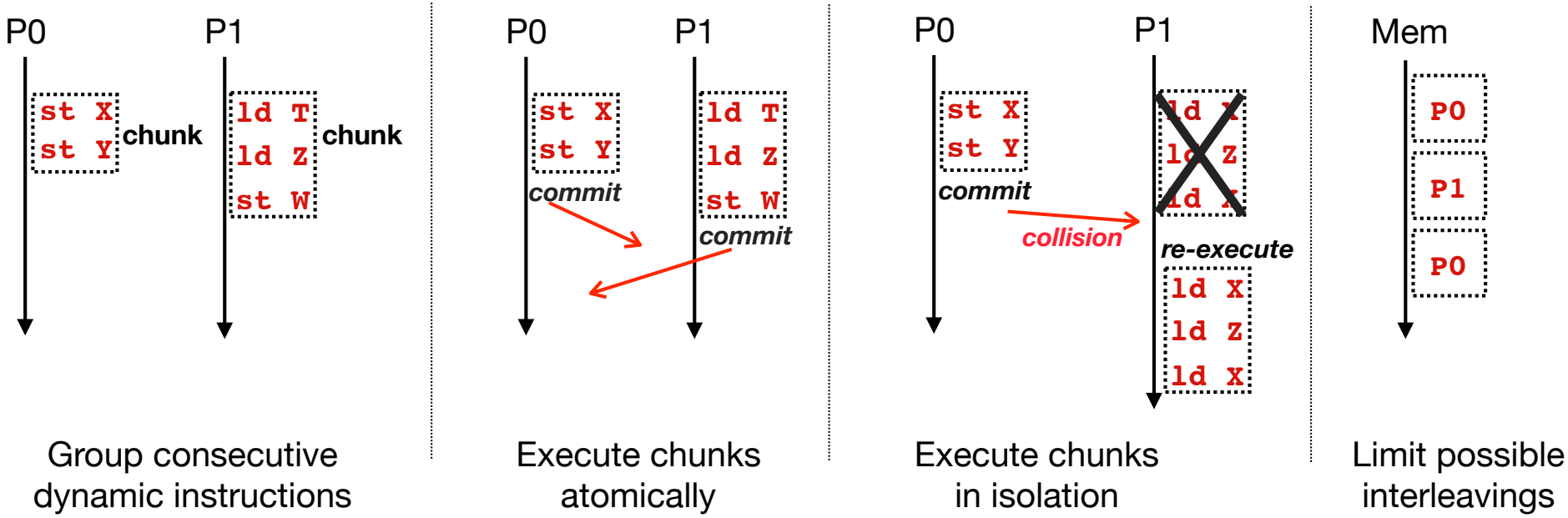
ReRun (*ISCA 08*)

Outline

- Background on replay schemes
- DeLorean: a Chunk-based replay scheme
- Advanced DeLorean Execution
- DeLorean Replay
- Evaluation

DeLorean

Chunk-Based Execution



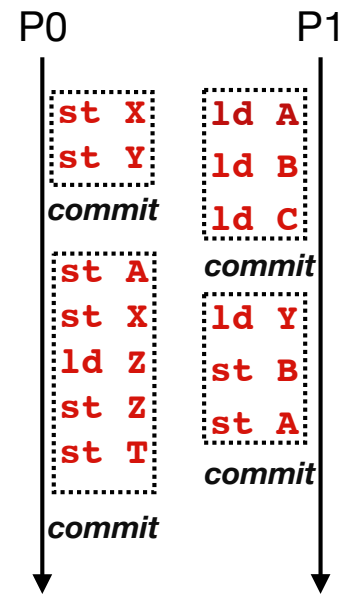
- Memory ops can be reordered and overlapped within and across chunks
- Memory access interleaving happens at chunk boundaries
- Chunk-based systems: TCC (*ISCA 04*), IT (*ICPS 05*), BulkSC (*ISCA 07*)

A Great Substrate for MP Replay

- Overlapping/reordering enables high-speed execution and replay
  - Performance similar to a system with Release Consistency (RC)
- Limited memory interleaving minimizes logging requirements
  - No need to record individual dependences

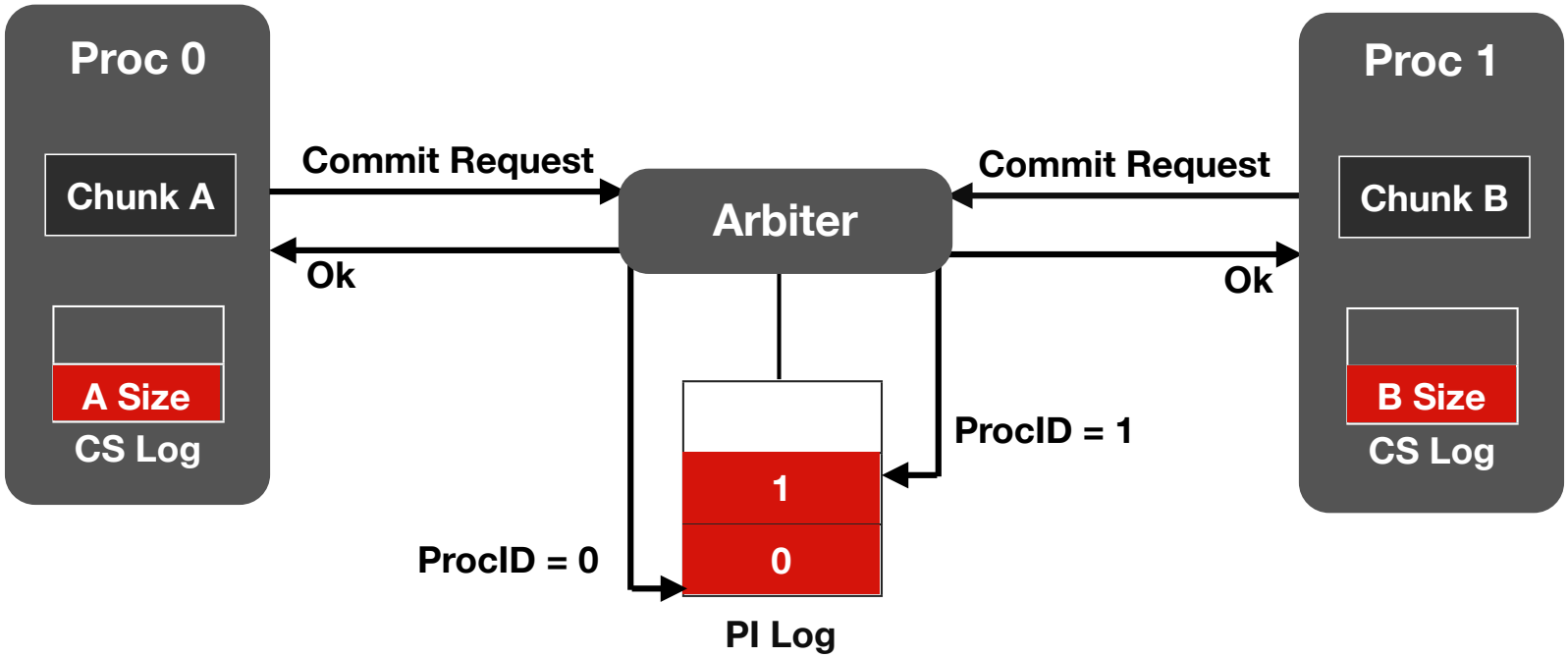
Chunk-Based Replay: DeLorean

- Replaying a chunk-based system means:
  - Generate same chunks during initial execution and replay
  - And commit them in the same order
- DeLorean's Log stores total order of chunk commits and their size
- DeLorean's log is very small:
  - Each entry is short: (ProcID, ChunkSize)
  - Log updated infrequently
  - Aggressive designs can make it even smaller



Log	
P0	2
P1	3
P1	3
P0	5

Basic DeLorean Operation



- Log implemented as two different structures:
  - Chunk Size (CS) Log
  - Processor Interleaving (PI) Log

Outline

- Background on replay schemes
- DeLorean: a Chunk-Based replay scheme
- Advanced DeLorean Execution
- DeLorean Replay
- Evaluation

DeLorean

Reducing DeLorean's Log

1. Use large chunks
2. Fixed-size chunking
3. Predefined interleaving

PI Log	P0 CS Log	P1 CS Log
0	2000	2000
1	2000	2000
0	2000	2000
1	2000	500
0	100	500
1	400	75
0	325	
1		
0		
0		
0		
1		
1		

Reducing DeLorean's Log

1. Use large chunks
   - \+ Fewer entries in log
   - − More collisions and cache overflows

2. Fixed-size chunking
   - \+ *Almost* no
   - − Need to handle cache overflows

3. Predefined interleaving
   - \+ No PI log
   - − Performance degradation

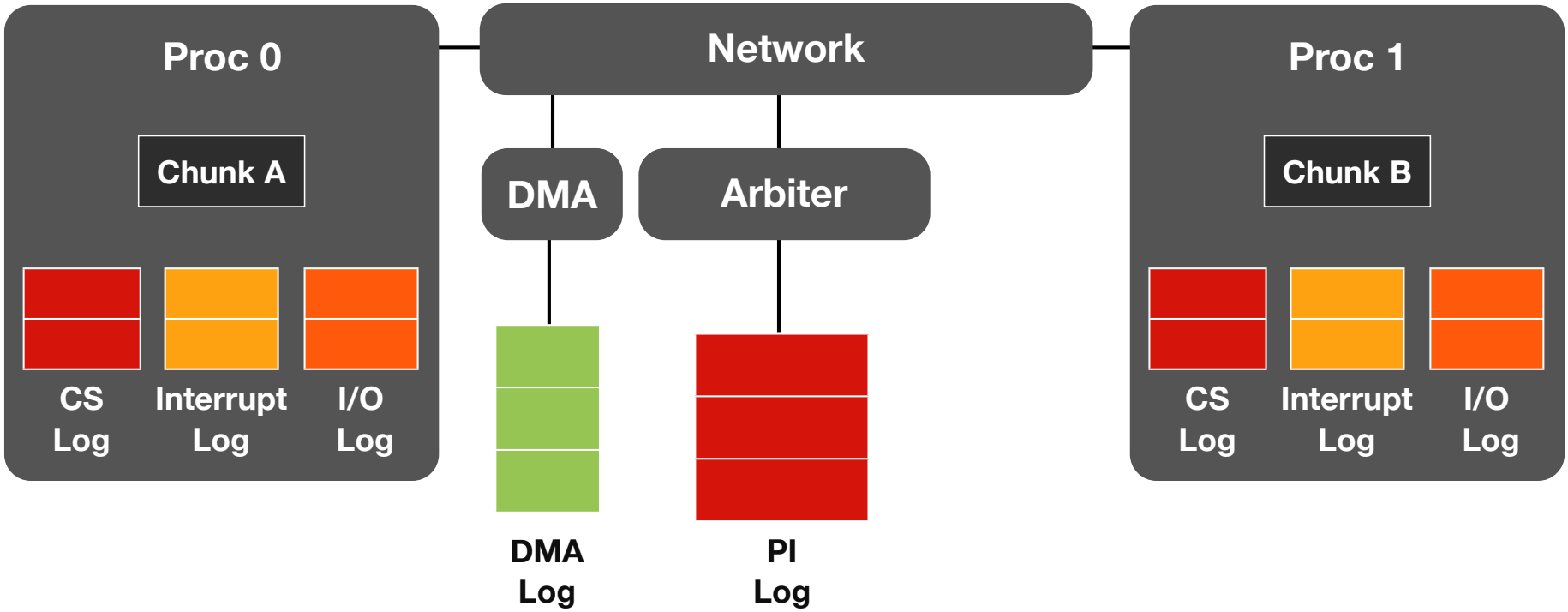
DeLorean Execution Modes

Nano	Pico
1 Use large chunks	1 Use large chunks
2 Fixed-size chunking (tiny CS log)	2 Fixed-size chunking (tiny CS log)
	3 Predefined interleaving (no PI Log)

Outline

- Background on replay schemes
- DeLorean: Chunk-Based replay scheme
- Advanced DeLorean Execution
- DeLorean Replay
- Evaluation

Overall System



- Interrupt, I/O and DMA logs also appear in all past replay schemes

DeLorean Replay

- Virtual machine monitor loads initial system checkpoint
  - Interrupt Log provides initial-execution interrupts
  - I/O Log provides initial-execution I/O inputs
- Processors execute normally (they execute in parallel):
  - Use CS Log to generate chunks with exceptional sizes (from cache overflows)
- Arbiter uses PI Log to enforce initial-execution commit order (or predefined order in *Pico*)

Also in the paper

- Rules for deterministic chunking
- How to deal with interrupts, traps, cache overflows, I/O ops
- Scalability of *Pico*
- A detailed view of DeLorean's architecture
- Proof of why DeLorean's replay is deterministic
- Combination with other HW replay schemes

Outline

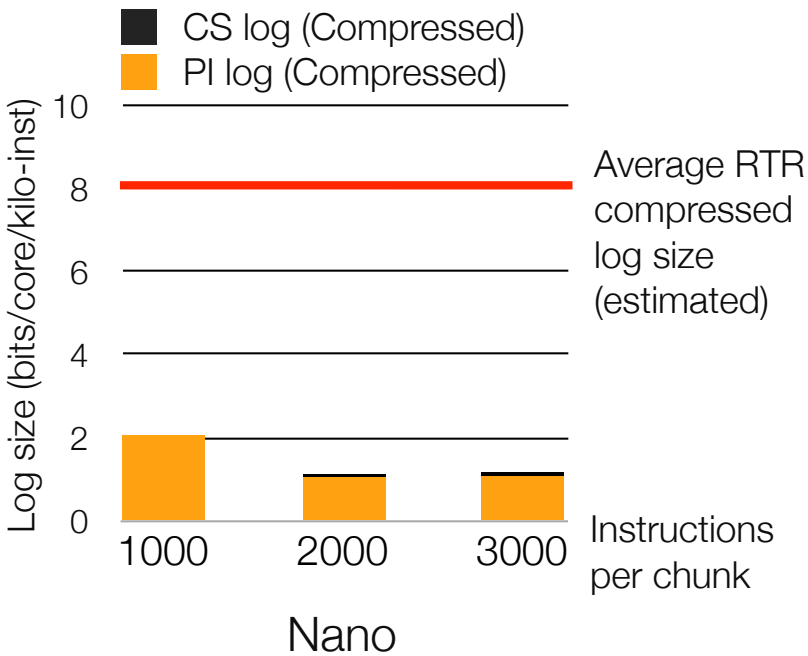
- Background on replay schemes
- DeLorean: Chunk-Based replay scheme
- Advanced DeLorean Execution
- DeLorean Replay
- Evaluation

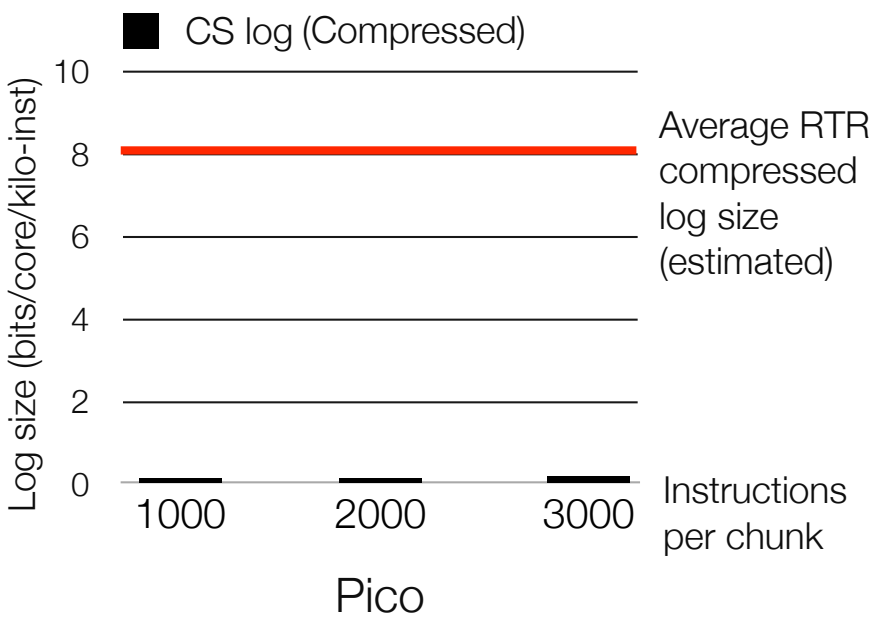
DeLorean

Evaluation Setup

- Chunk-based simulator supports both execution and replay
- 8-processor CMP
  - 1000, 2000, 3000 instructions per chunk
  - PI Log: 4-bit entries
  - CS Log: 32-bit entries
- Applications: SPLASH, SPECjbb 2000 and SPECweb 2005

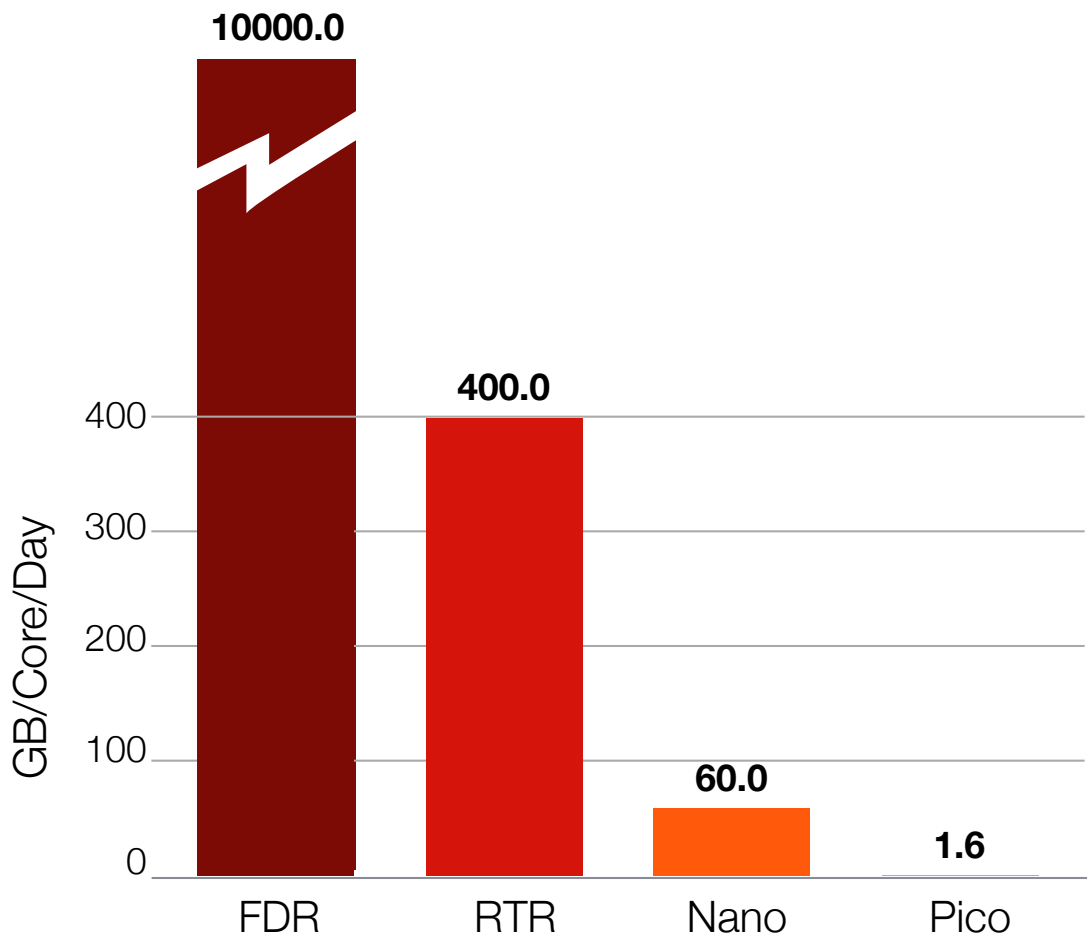
Log Space Requirements





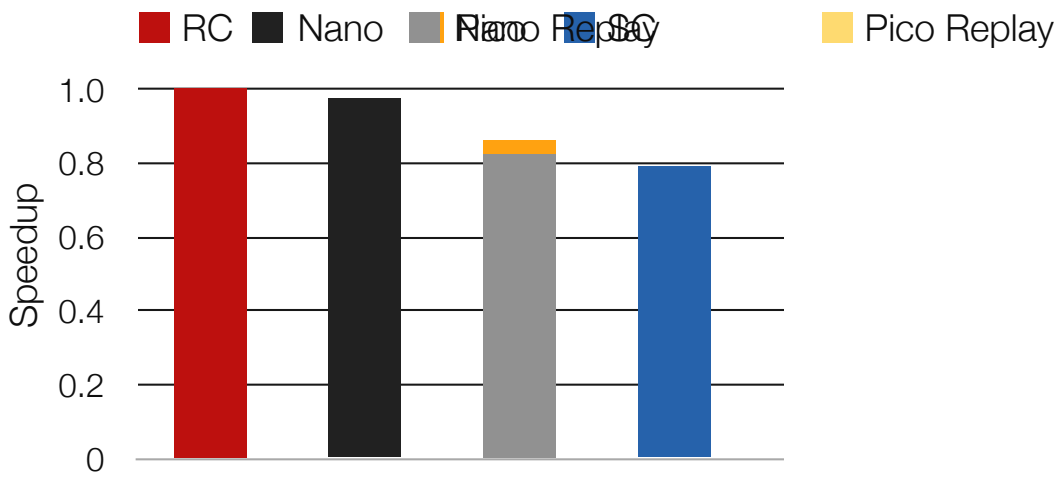
- Nano's log is 16% of RTR's (2000-instruction configuration)
  - With one additional optimization, Nano's log is 7.5% of RTR's
- Picos's log is 0.6% of RTR's (1000-instruction configuration)

If We Were To Record a Day of Execution



- Execution and replay of long periods of time is feasible

Performance



Initial and Replay Execution

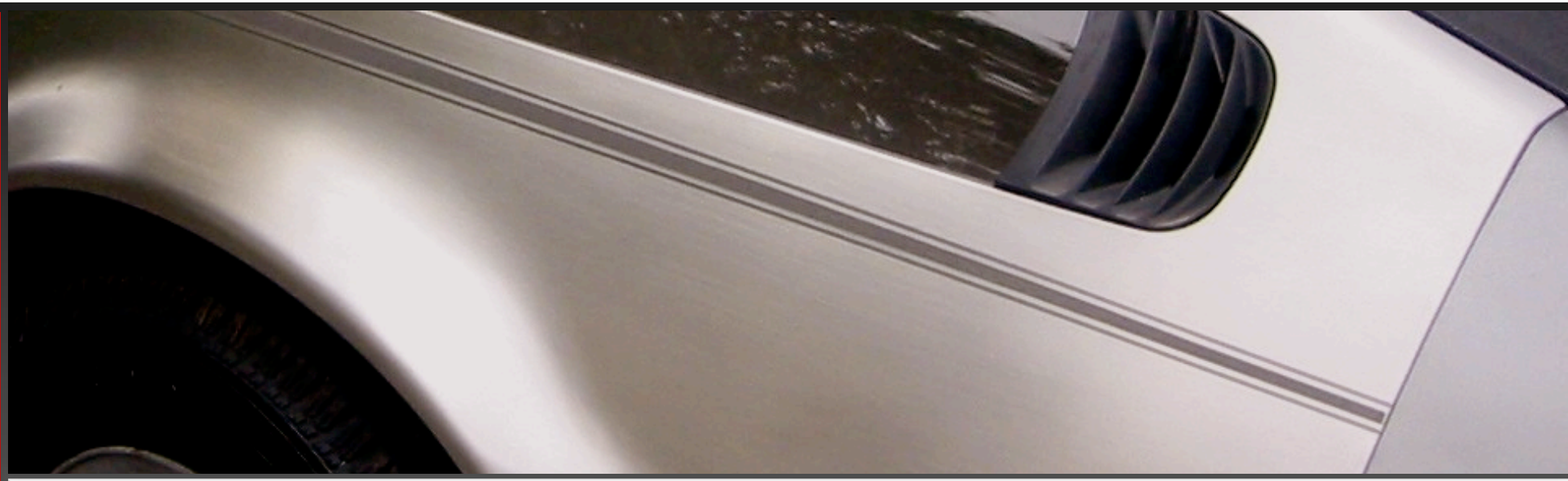
- Initial execution: Nano almost as fast as RC, Pico faster than SC
- Replay: comparable to initial execution speed

Conclusions

- DeLorean is a new approach to HW replay of parallel applications
- Advantages over traditional HW replay systems:
  - Executes at high speed (like most aggressive consistency models)
  - Replays at comparable speed
  - Summarizes execution into a very small log (0.6% of current schemes)
- Great help for programmers that need to debug parallel codes

Future Work

- Adapt DeLorean to conventional multiprocessors without chunk-based execution
- Take conventional replay schemes, use very aggressive SC implementations, and compare to DeLorean
- Combine the best aspects of the different recording approaches (DeLorean, RTR and Strata)



DeLorean:

Recording and Deterministically Replaying Shared Memory Multiprocessor Execution Efficiently

Pablo Montesinos, Luis Ceze* and Josep Torrellas

Department of Computer Science
University of Illinois at Urbana-Champaign

http://iacoma.cs.uiuc.edu

*Department of Computer Science and Engineering
University of Washington