

## Research Summary

Multiprocessors are ubiquitous, but programming them continues to be challenging.

**Our Goal:** Simplify multiprocessor programming without compromising performance

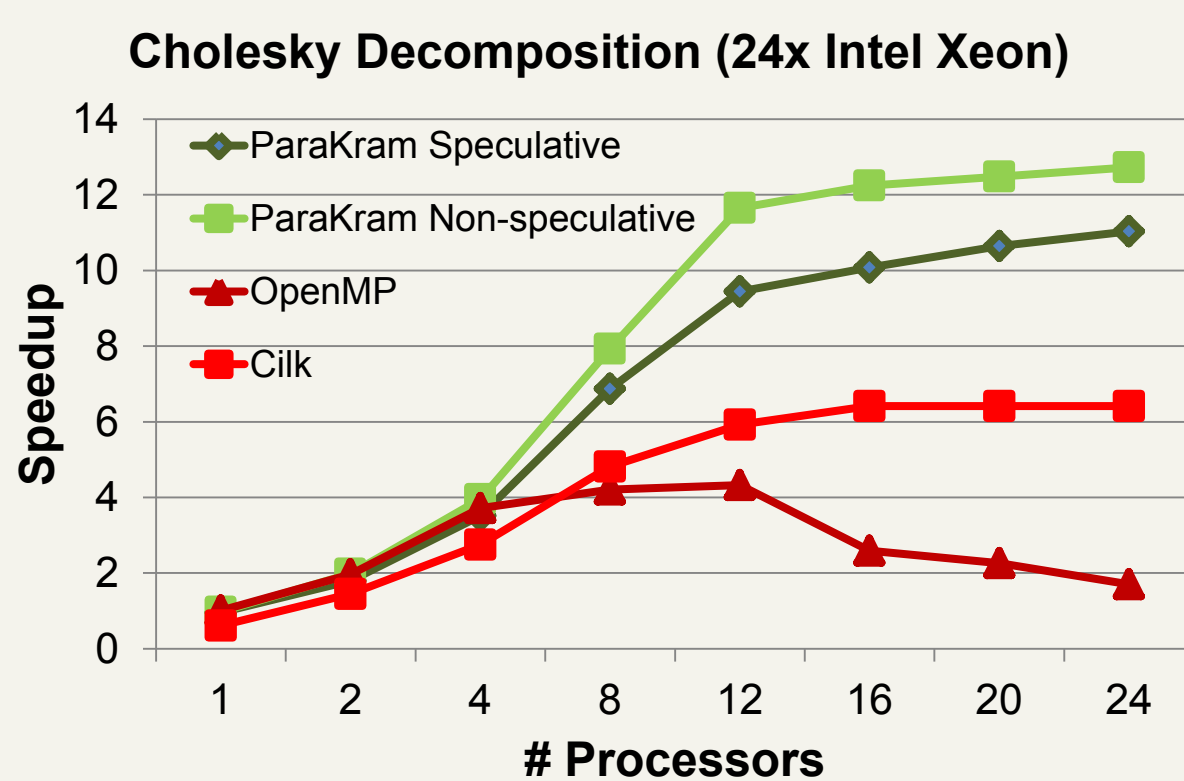
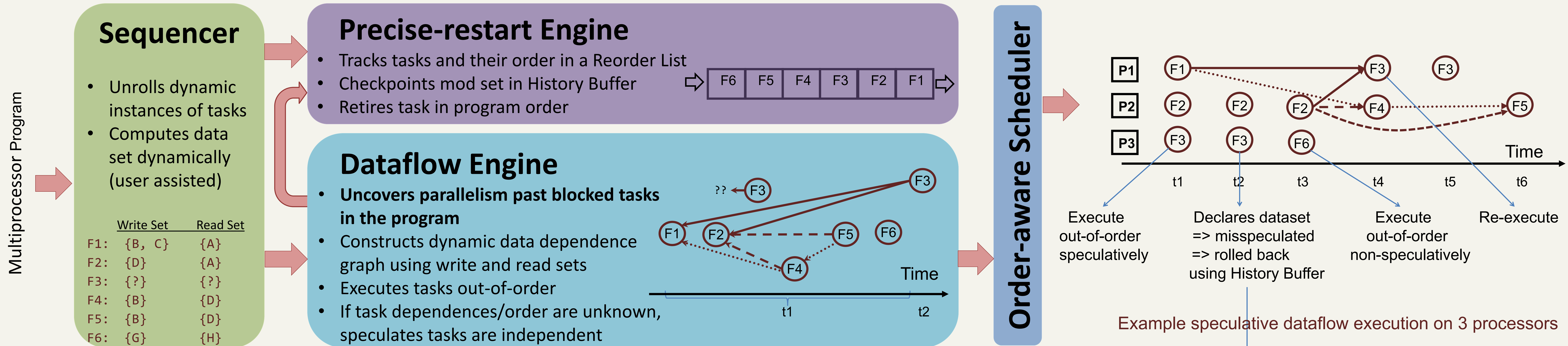
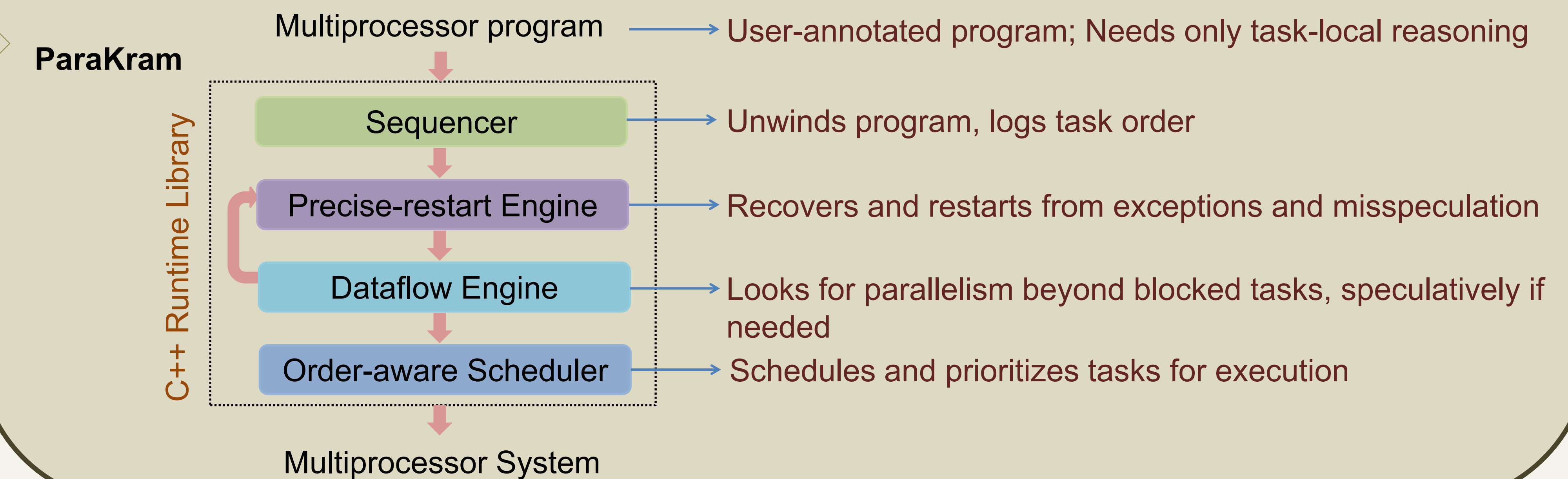
Conventional Wisdom	Our Approach
Order in programs obstructs parallelism	Order can help to expose parallelism!
Use non-deterministic programs, or make dataflow in programs explicit	Use <b>ordered</b> programs; maintain precise <b>program-order</b> execution semantics
Programmer should expose parallelism	Use <b>run-time dataflow</b> and <b>speculative</b> techniques to expose parallelism

Benefits:

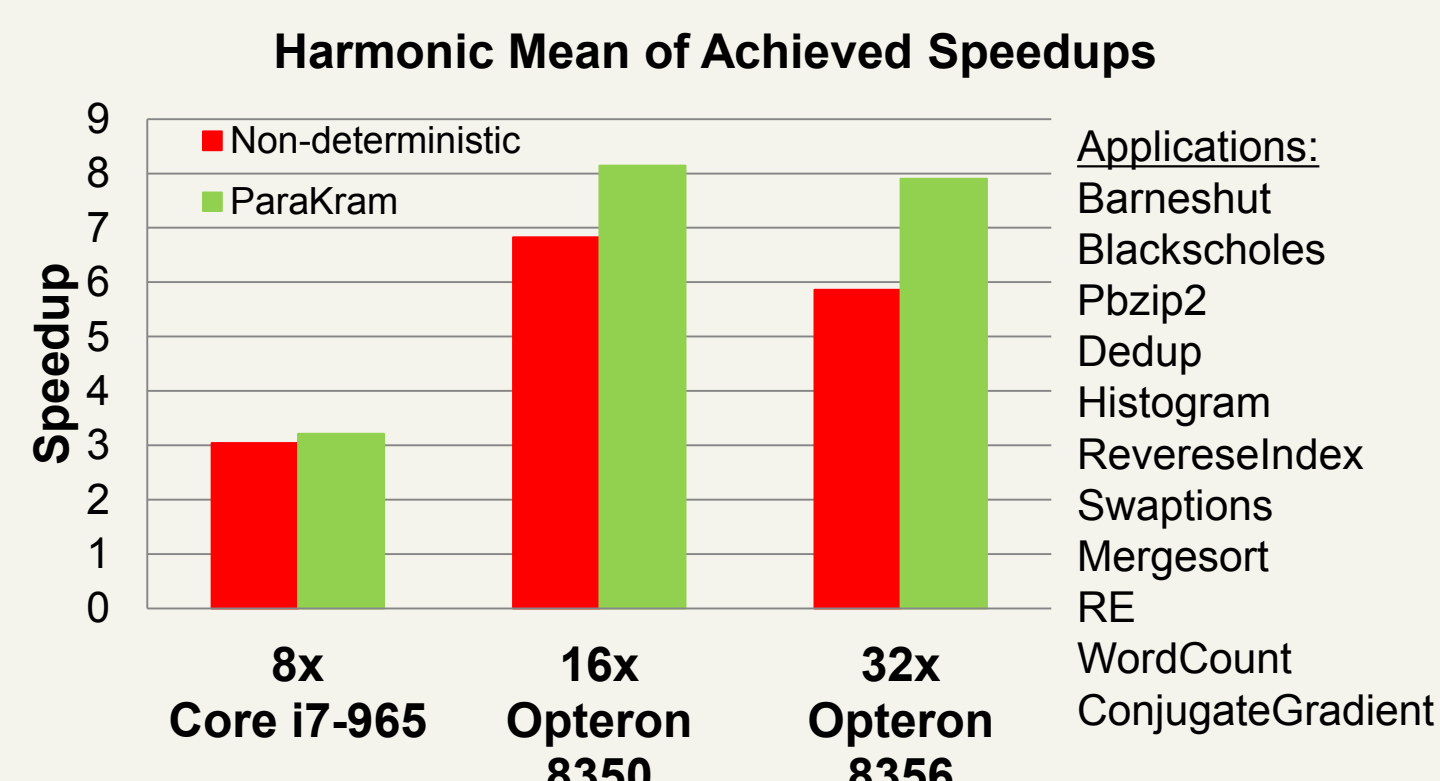
- Simplified programming; Simplified system design; Better reliability
- Performance at par or better (5% to 288%) than conventional methods

## Proposed System: ParaKram

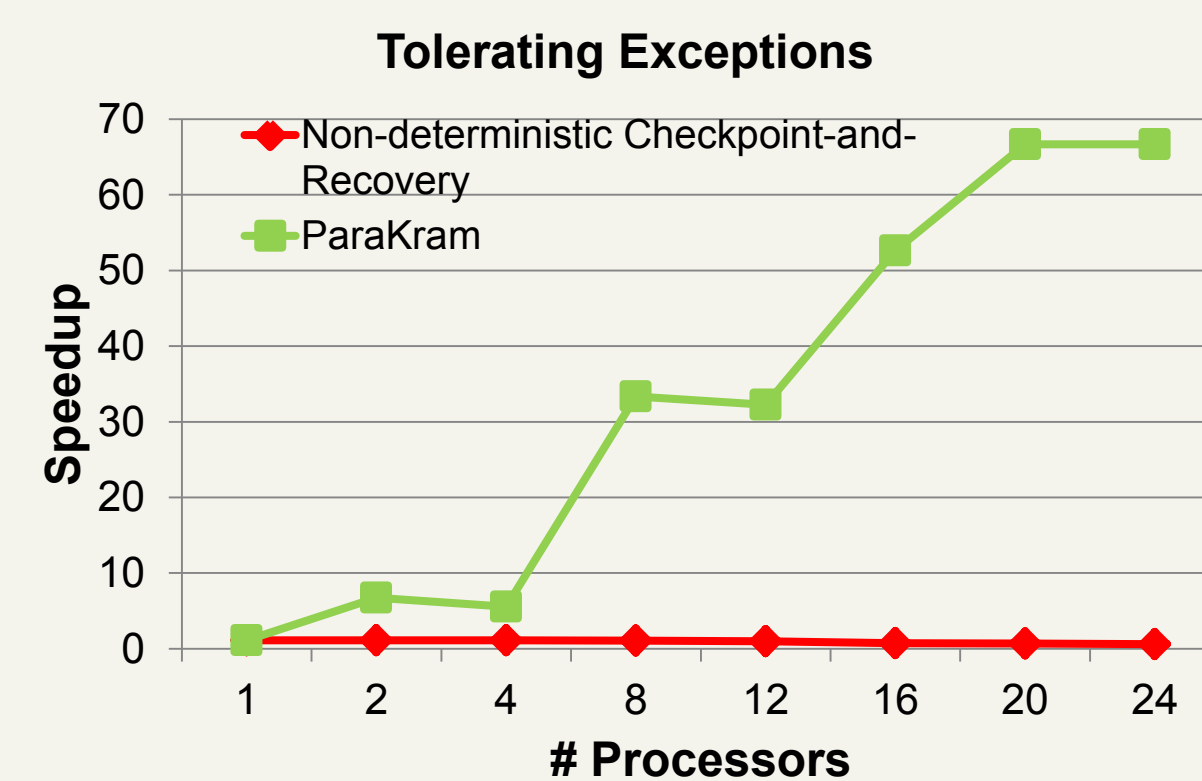
- A **Runtime system** to manage the multiprocessor program's execution
- Performs **out-of-order superscalar processor-like execution** on multicores
  - Uses data dependences to perform **dataflow** execution
  - Supports program-wide **precise exceptions** (overall order)
  - **Speculates** when dependences are unknown, but keeps ordered semantics



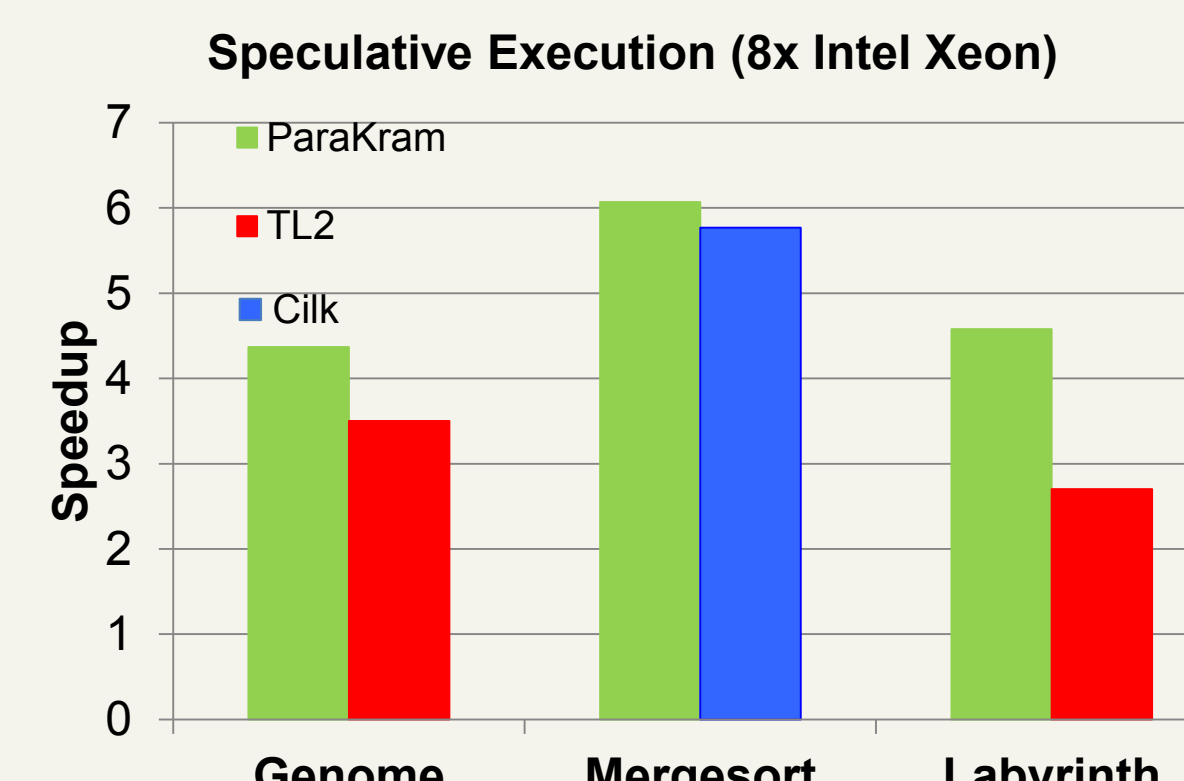
ParaKram speedup is 288% higher than non-deterministic OpenMP, 75% over Cilk



ParaKram speedup (harmonic mean) is 20% higher than non-deterministic Pthreads (excludes Cholesky)



ParaKram scales with system size; Non-deterministic method does not scale



ParaKram speedup is up to 77% higher than non-deterministic Cilk and TL2 STM

Example speculative dataflow execution on 3 processors

Epoch	Reorder List Entries	Completed	Retired
t1	[F6, F5, F4, F3, F2, F1]		
t2	[F6, F5, F4, F3, F2]	F1	F1
t3	[F6, F5, F4, F3, F2]	F1	
t4	[F6, F5, F4, F3]	F1, F6, F2	F2

Precisely restarting misspeculated task (F3 from above)