Managed by Fermi Research Alliance, LLC for the U.S. Department of Energy Office of Science

Using the CMS Threaded Application in a Production Environment

Dr Christopher Jones

for CMS Offline and Computing

CHEP 2015

14 April 2015

Goals for Multi-Threading in CMS

Application

- Reduce memory needed per CPU core

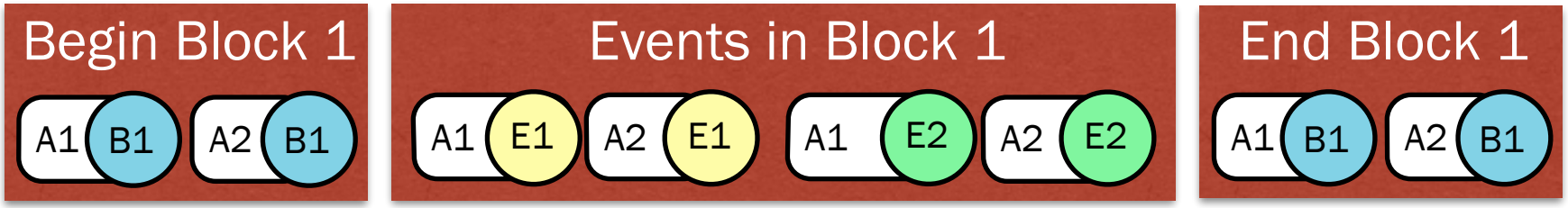
Workflow System

- Reduce number of requests to database
- Reduce number of open files
- Reduce time it takes to process one *block*
  - *block* is 23 seconds of data taking
  - all events in a *block* must be processed by only one job
- Reduce number of jobs to be tracked in a workflow

See Monday's talk "*Evolution of CMS workload management towards multicore job support*" for details on workflow changes

Fermilab

Original Single-Threaded Application Design



Events are grouped in *blocks*

Block: B1

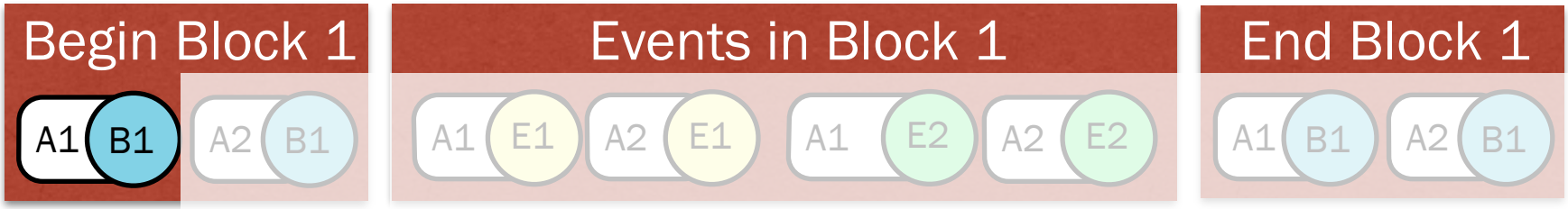
Events: E1 E2

Algorithms are used to process blocks and events

Algorithms: A1 A2

Framework runs algorithms in a specified order

Original Single-Threaded Application Design



Events are grouped in *blocks*

Block: B1

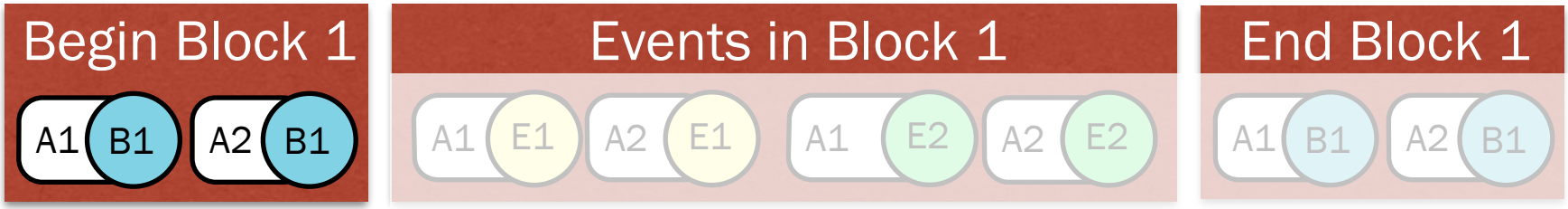
Events: E1 E2

Algorithms are used to process blocks and events

Algorithms: A1 A2

Framework runs algorithms in a specified order

Original Single-Threaded Application Design



Events are grouped in *blocks*

Block: B1

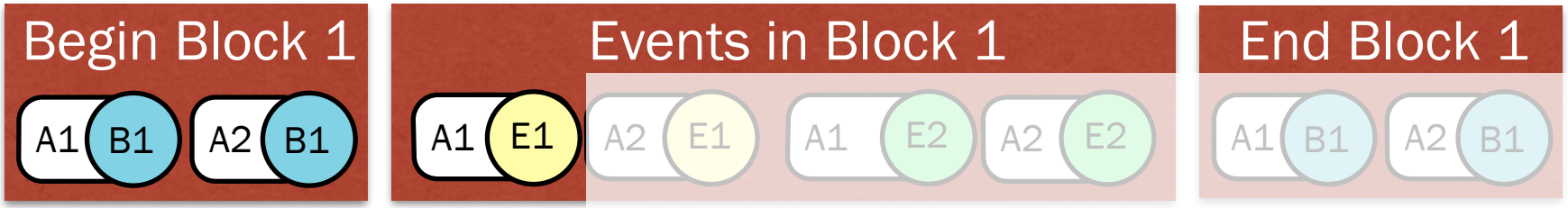
Events: E1 E2

Algorithms are used to process blocks and events

Algorithms: A1 A2

Framework runs algorithms in a specified order

Original Single-Threaded Application Design



Events are grouped in *blocks*

Block: B1

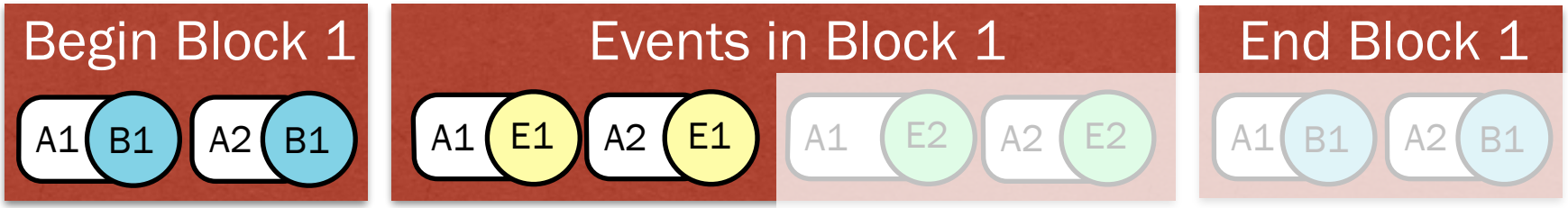
Events: E1 E2

Algorithms are used to process blocks and events

Algorithms: A1 A2

Framework runs algorithms in a specified order

Original Single-Threaded Application Design



Events are grouped in *blocks*

Block: B1

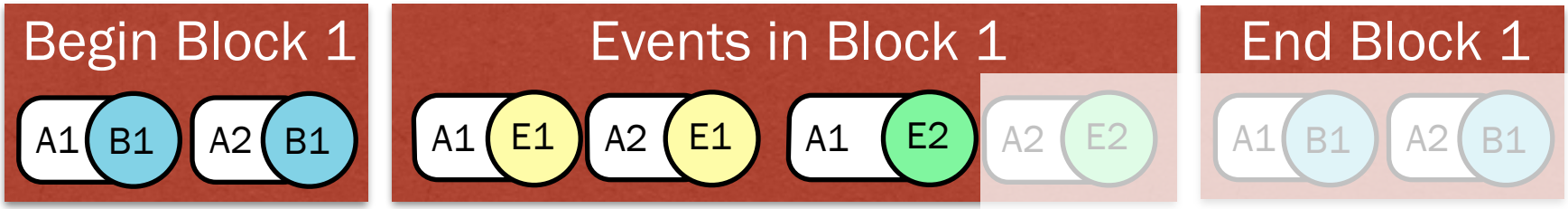
Events: E1 E2

Algorithms are used to process blocks and events

Algorithms: A1 A2

Framework runs algorithms in a specified order

Original Single-Threaded Application Design



Events are grouped in *blocks*

Block: B1

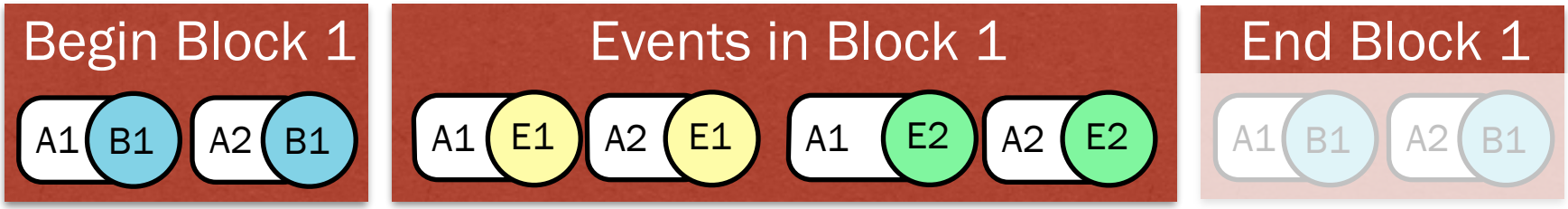
Events: E1 E2

Algorithms are used to process blocks and events

Algorithms: A1 A2

Framework runs algorithms in a specified order

Original Single-Threaded Application Design



Events are grouped in *blocks*

Block: B1

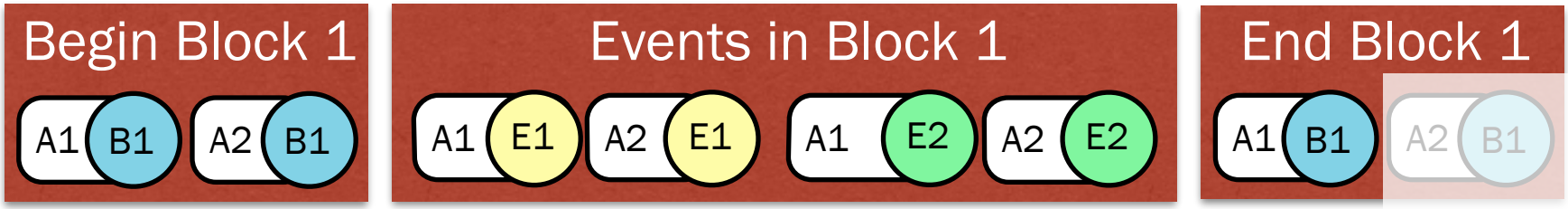
Events: E1 E2

Algorithms are used to process blocks and events

Algorithms: A1 A2

Framework runs algorithms in a specified order

Original Single-Threaded Application Design



Events are grouped in *blocks*

Block: B1

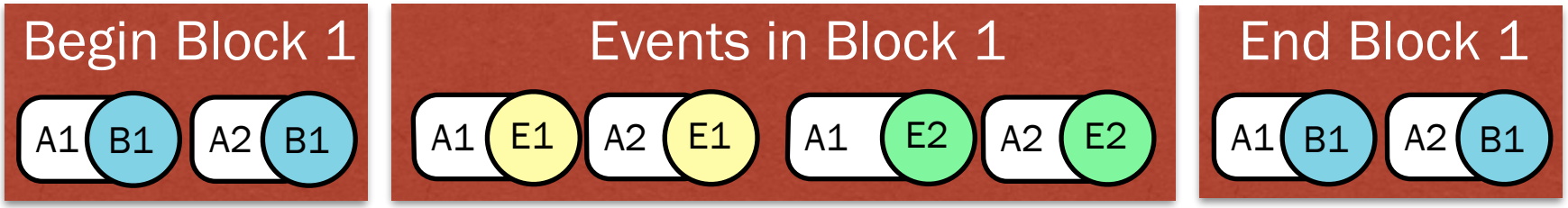
Events: E1 E2

Algorithms are used to process blocks and events

Algorithms: A1 A2

Framework runs algorithms in a specified order

Original Single-Threaded Application Design



Events are grouped in *blocks*

Block: B1

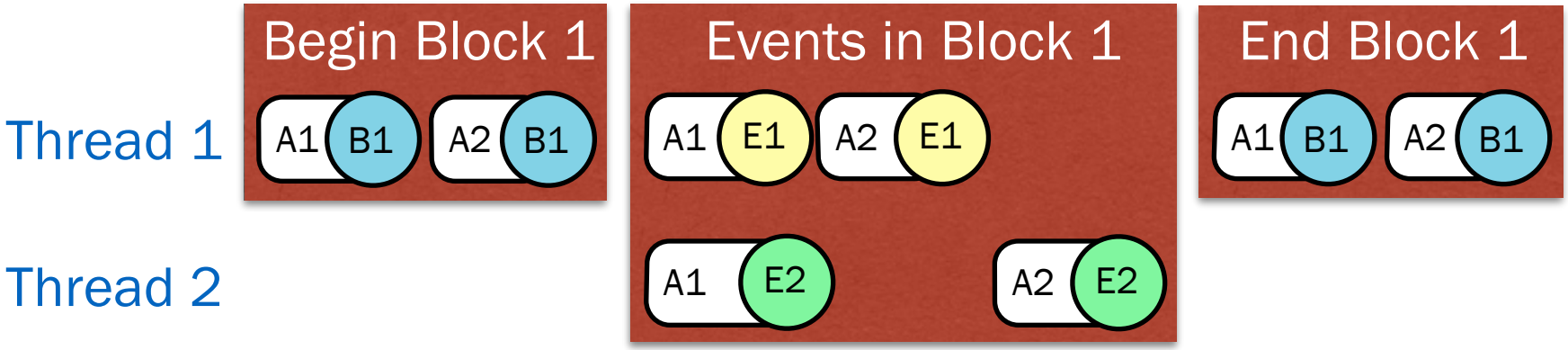
Events: E1 E2

Algorithms are used to process blocks and events

Algorithms: A1 A2

Framework runs algorithms in a specified order

Multi-Threaded Application Design



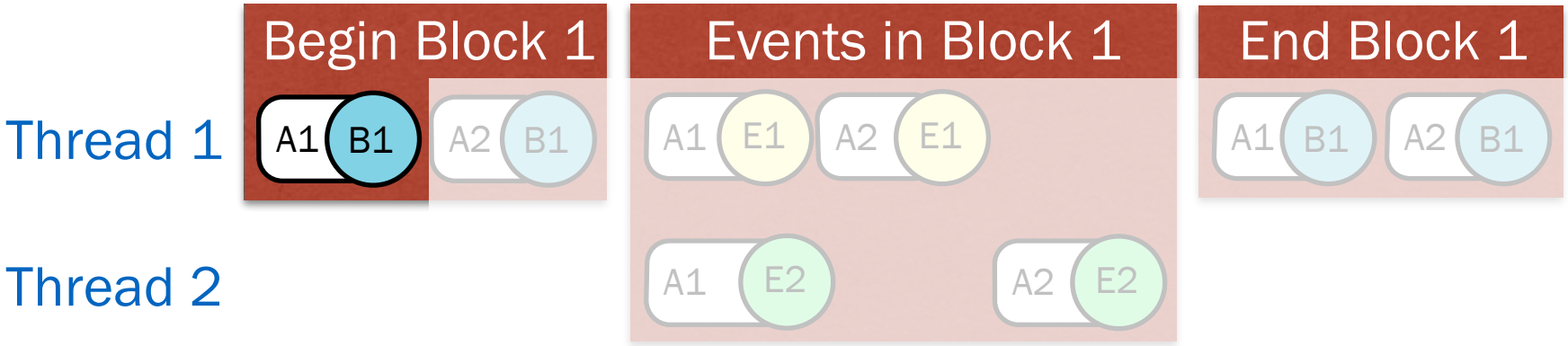
Evolved from original single-threaded application

- Updated algorithms can run concurrently (A1)
- Only one legacy algorithm is allowed to run at a time (A2)

Processes events in *block* concurrently

- Synchronizes on *block* boundaries (B1)

Multi-Threaded Application Design



Evolved from original single-threaded application

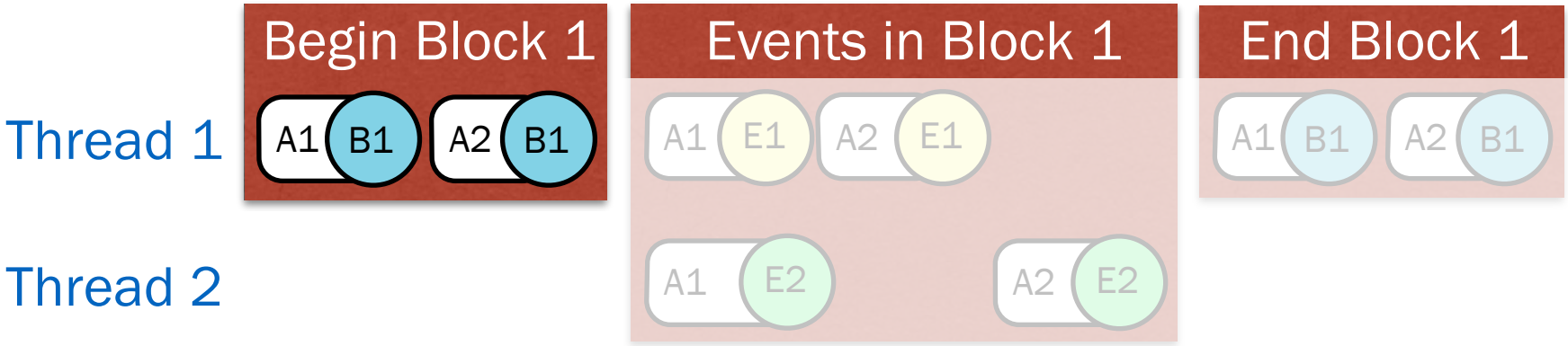
- Updated algorithms can run concurrently (A1)
- Only one legacy algorithm is allowed to run at a time (A2)

Processes events in *block* concurrently

- Synchronizes on *block* boundaries (B1)

Fermilab

Multi-Threaded Application Design



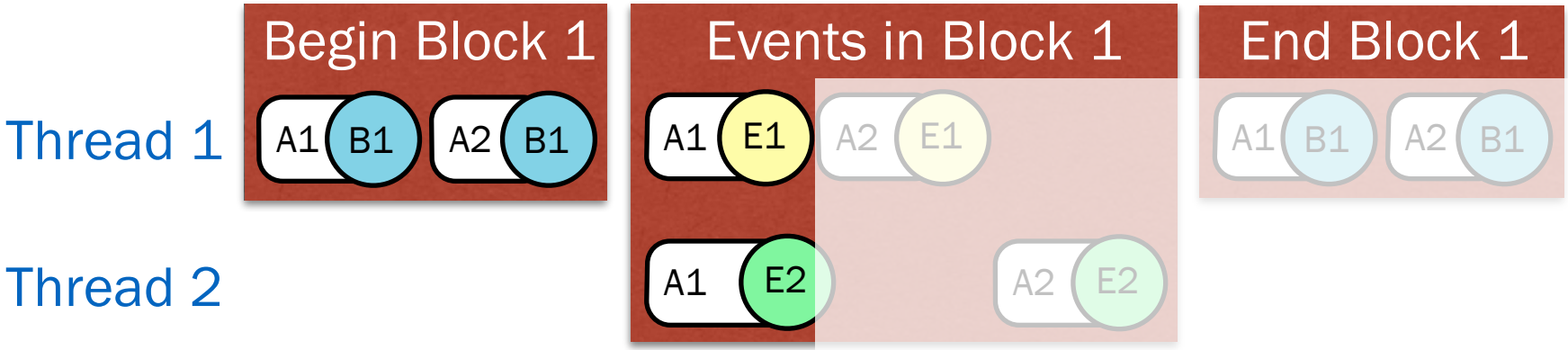
Evolved from original single-threaded application

- Updated algorithms can run concurrently (A1)
- Only one legacy algorithm is allowed to run at a time (A2)

Processes events in *block* concurrently

- Synchronizes on *block* boundaries (B1)

Multi-Threaded Application Design



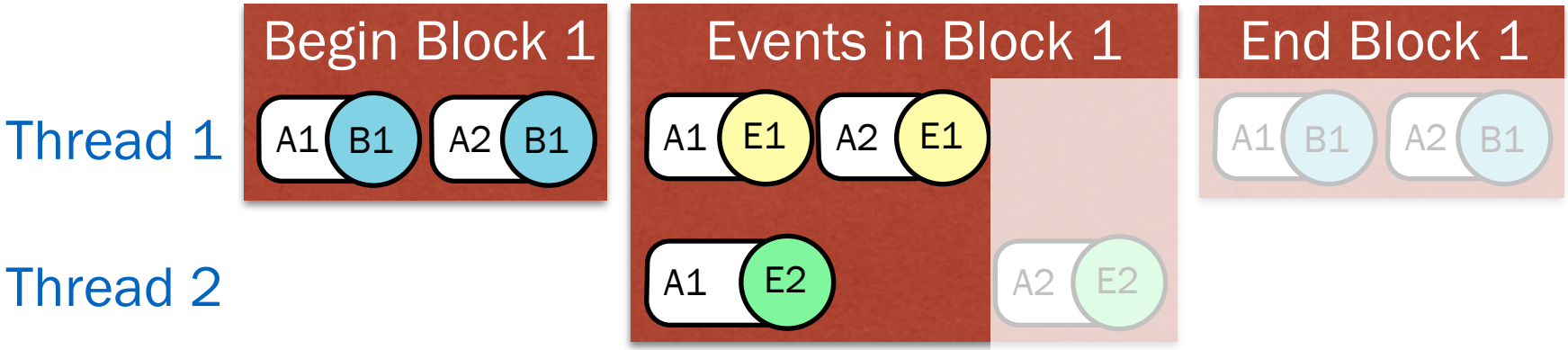
Evolved from original single-threaded application

- Updated algorithms can run concurrently (A1)
- Only one legacy algorithm is allowed to run at a time (A2)

Processes events in *block* concurrently

- Synchronizes on *block* boundaries (B1)

Multi-Threaded Application Design



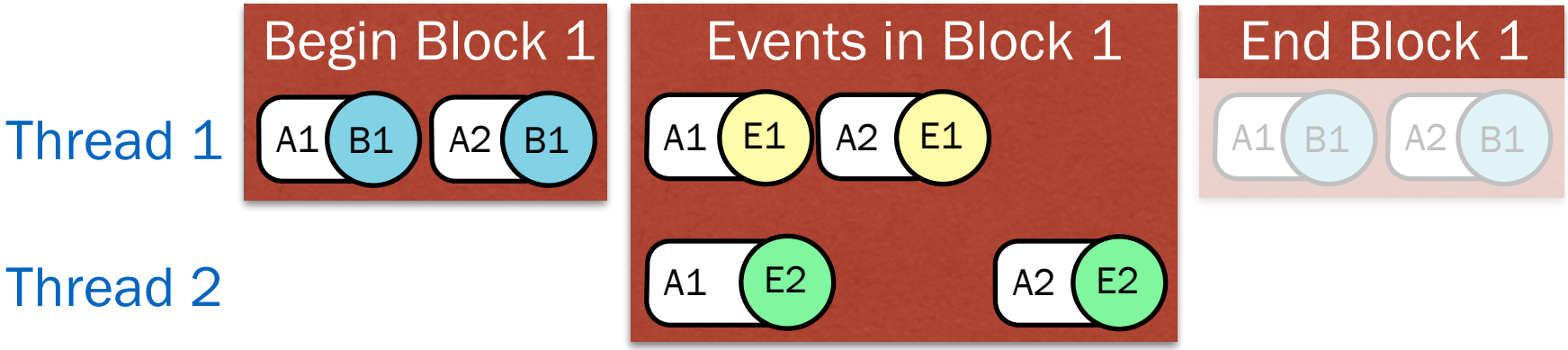
Evolved from original single-threaded application

- Updated algorithms can run concurrently (A1)
- Only one legacy algorithm is allowed to run at a time (A2)

Processes events in *block* concurrently

- Synchronizes on *block* boundaries (B1)

Multi-Threaded Application Design



Evolved from original single-threaded application

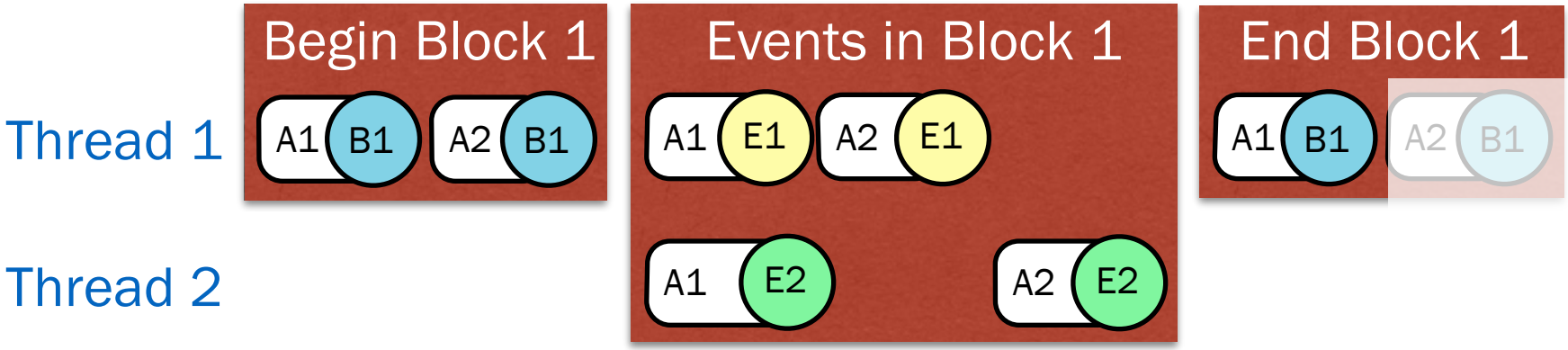
- Updated algorithms can run concurrently (A1)
- Only one legacy algorithm is allowed to run at a time (A2)

Processes events in *block* concurrently

- Synchronizes on *block* boundaries (B1)

Fermilab

Multi-Threaded Application Design



Evolved from original single-threaded application

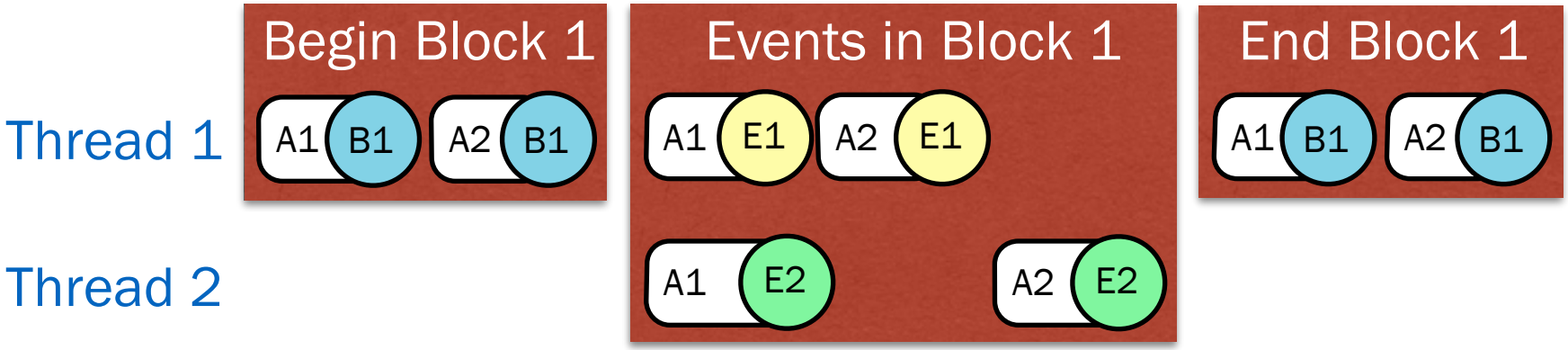
- Updated algorithms can run concurrently (A1)
- Only one legacy algorithm is allowed to run at a time (A2)

Processes events in *block* concurrently

- Synchronizes on *block* boundaries (B1)

Fermilab

Multi-Threaded Application Design



Evolved from original single-threaded application

- Updated algorithms can run concurrently (A1)
- Only one legacy algorithm is allowed to run at a time (A2)

Processes events in *block* concurrently

- Synchronizes on *block* boundaries (B1)

Application Changes

Have been steadily converting algorithms to be thread-friendly

- 1100 algorithms were converted
- 2800 algorithms have yet to be converted

Took 6.3 person-years to do changes

- Converting to threaded-framework: 3 person-years
- Converting algorithms: 3 person-years
- Converting external libraries: .3 person-years

Many thanks to the ROOT team for incorporating our changes

Fermilab

2015 Data Processing Plan

Run reconstruction step with multiple threads

- Reduces time to process one *block* for real data

Run all other steps single-threaded

Following results are only for the reconstruction step

Fermilab

Application Performance: Measurement Technique

Machine

AMD 64-core Opteron 6376 with 126GB RAM

Data Type

$t\bar{t}$ Monte Carlo with LHC Run 2 conditions

Low Pileup: 50ns bunch spacing with average 4 collisions/crossing

High Pileup: 25ns bunch spacing with average 40 collisions/crossing

Application Configuration

Reconstruction

Reconstruction plus Monitoring

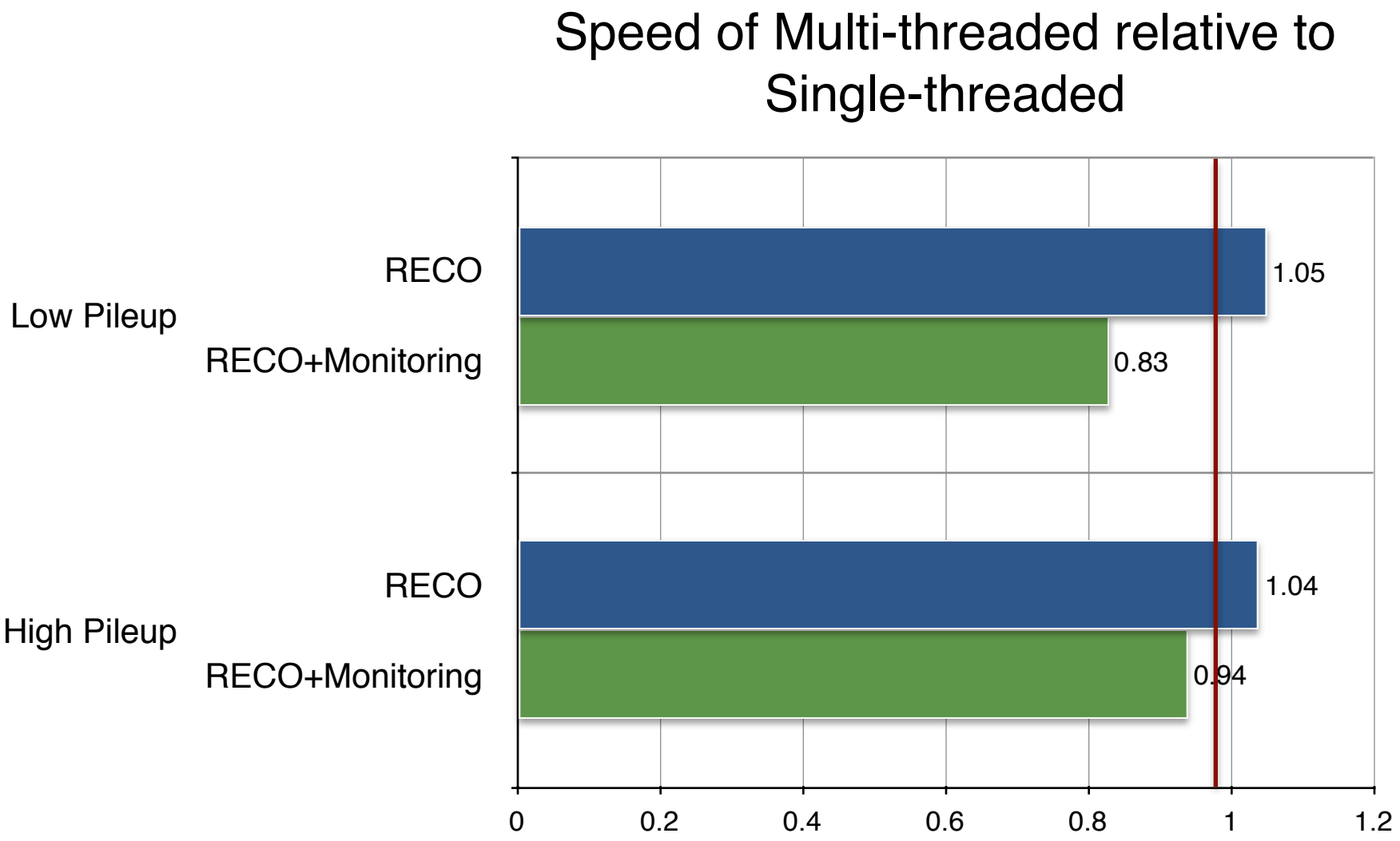
Jobs

Single-core: 64 jobs run simultaneously

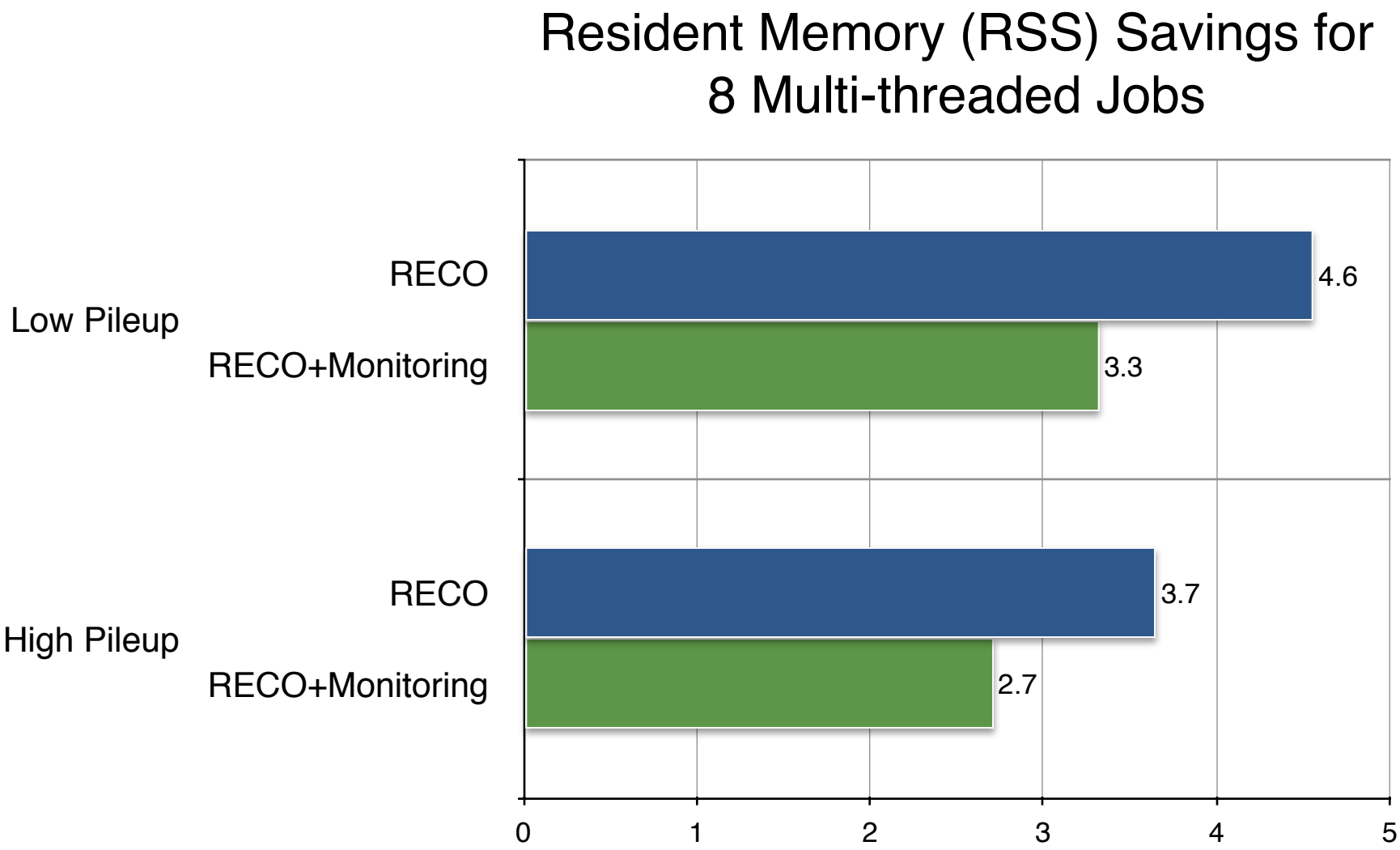
Multi-core: 8 simultaneous jobs each with 8 threads

Fermilab

Application Performance: Processing Time



Application Performance: Memory



Scale Test of Reconstruction

Reprocessed a sample of LHC Run 1 data

- Reconstruction and Monitoring
- Closer to low pileup than high pileup
- Each job processed multiple *blocks*

Jobs ran at all CMS Tier 1 sites

Used an older CMS software release

- Slightly less parallel efficient than new code
- Does not have all thread-safety fixes

Used only 4 threads per job

Scale Test Results: Failure Rates

72 of 45000 jobs failed because of the application

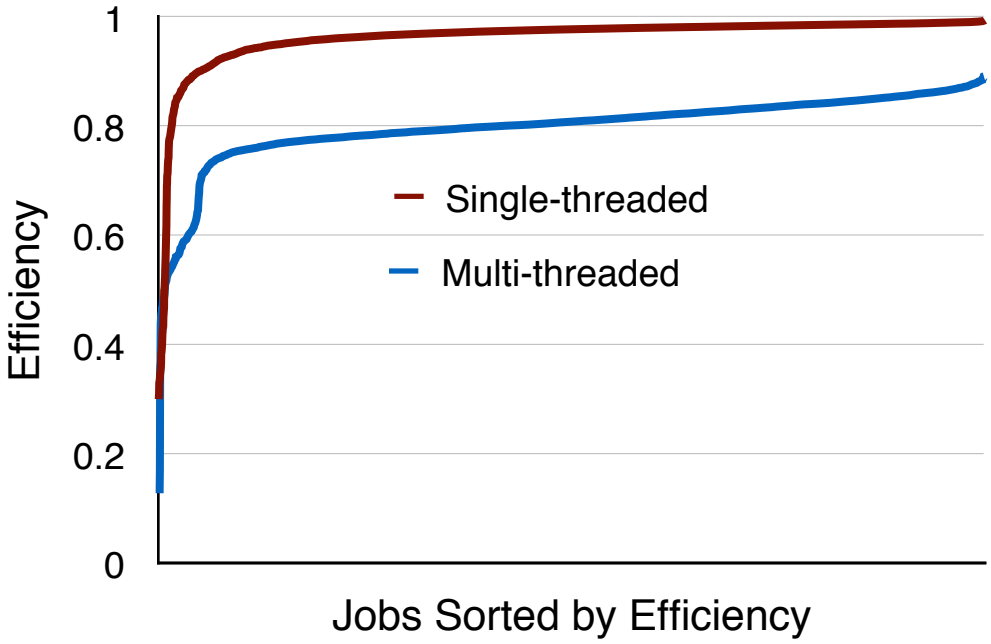
- >99.8% success rate for a job
- Not all failures were caused by threading problems

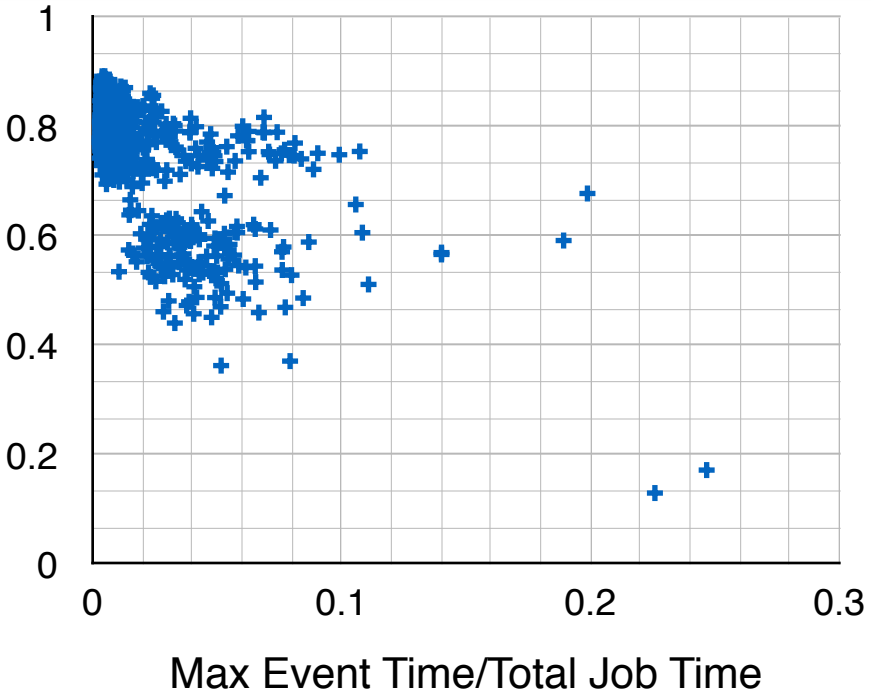
Workflow system will retry a job 4 times

- Threading related problems tend to appear randomly per job
- Chance of a job not succeeding after 4 retries: 1 in 160×10^9

Fermilab

Scale Test Results: CPU Efficiency





Achieved an average CPU efficiency of 80%

- Single-threaded average CPU efficiency 97%
- *Newest software much more thread efficient*

First event latency causes largest inefficiency

- e.g. database reading, first remote file read

Conclusion

CMS has successfully transitioned to using multiple threads

Job success rate is sufficient for Run 2 data processing

Newest software will give sufficient CPU efficiency at 8 threads

Ongoing work to increase parallelization

- Multiple algorithms in parallel within one event
- Parallel processing of *blocks*

Fermilab

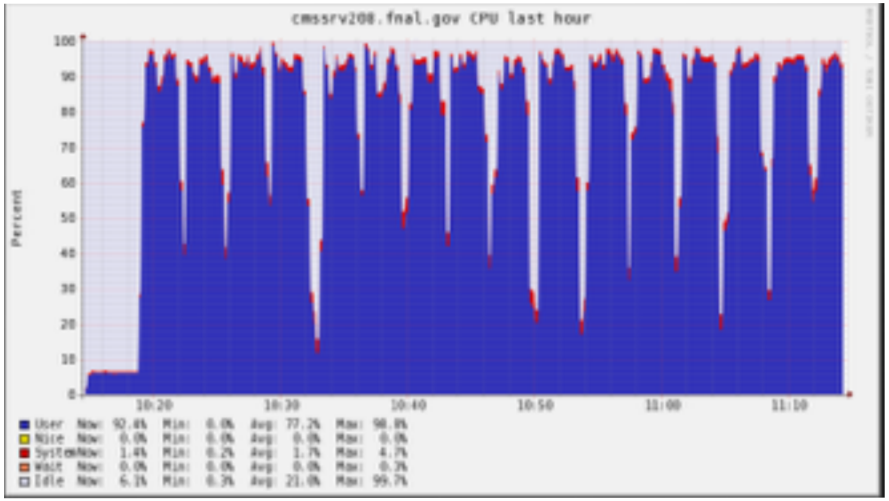
Backup Slides

Fermilab

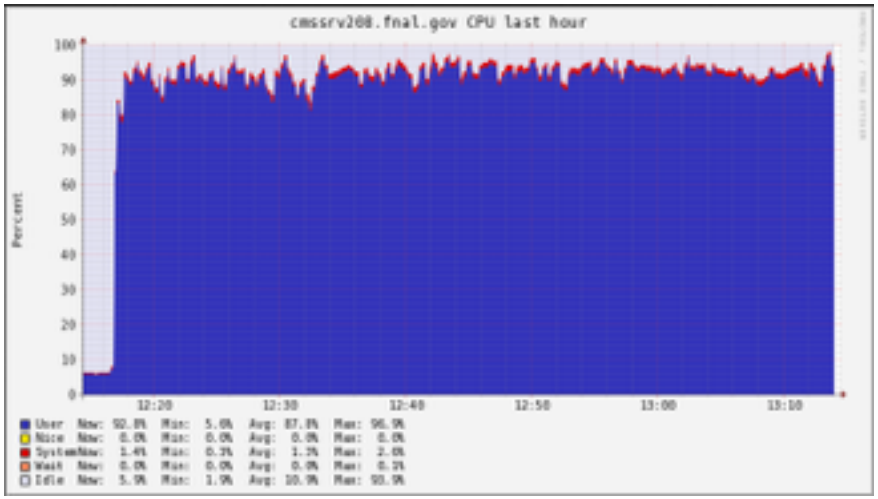
Luminosity Block Synchronization Effect

CPU Utilization

100 Events / Block



All Events in one Block



Fewer events in a block means more serialization

Application measurements done with 1 block

Future work will mitigate some of this effect