Parallelization of the FICO Xpress-Optimizer

Michael Perregaard and James Farmer

Timo Berthold
Fair Isaac Germany GmbH

FICO Xpress-Optimizer

- Objective
  - Linear objective
  - Convex quadratic objective
- Constraint types
  - Linear constraints
  - Convex quadratic constraints
  - Second order cones
  - Indicator constraints
  - Special ordered sets (SOS1 and SOS2)
- Column types
  - Continuous
  - Integeger
  - Semi-continuous
  - Semi-integer
  - Partial-integer

Barrier Optimizer

Simplex Optimizer

MIP Optimizer

Feature of Xpress solvers

Predictability and reproducibility

- All solver components deterministic by default.
- Platform independence:
  - win64, linux64 and mac64 will produce identical solves
- Minimal dependence on threads and architecture.

Dual Simplex:

- Dependence only on single versus multi-threaded.
- Force multi-threading [FORCEPARALLELDUAL=1] to remove all threading dependencies.

Barrier:

- Independent of cache size
- Mostly independent of vectorization support [CPUPLATFORM]
  - None, SSE2, AVX identical. Only AVX2 differs.
- Dependent on BARTHREADS and BARCORES.

Parallel technology: Shared memory

My laptop

- One 4-core socket
- 16 GB of RAM

My server

- Two 10-core sockets
- 256 GB of RAM

My smartphone

- One 4-core socket
- 2 GB of RAM

(My) HRLN III compute node

- Two 12-core sockets
- 64 GB of RAM

Parallel technology: Distributed memory

Supercomputers (Top 500)

#	Site	Manufact.	Processor	Cores	Total cores
1	Wuxi, China	Sunway	ShenWei 26010	256	10,649,600
2	Guangzhou, China	NUDT	Intel Xeon E5-2692v2	12	3,120,000
3	Livermore, USA	Cray	Power BQC	16	1,572,864
96	Berlin, Germany	Cray	Intel Xeon E5-2695v2	12	44,928
500	Nizhni Novgorod, Russia	Niagara	Intel Xeon E5-2660v2	10	10,800

June 2016

Parallel technology: Distributed memory

Supercomputers (Top 500)

#	Site	Manufact.	Processor		tal cores
1	Wuxi, China	Sunway	ShenWei 26	256	10,649,600
2	Guangzhou, China	NUDT	In	12	3,120,000
3	Livermore, USA	Cray	BQC	16	1,572,864
96	Berlin, Germany		Intel Xeon E5-2695v2	12	44,928
500	Nizhni	Niagara	Intel Xeon E5-2660v2	10	10,800

Following presentation by Yuji Shinano

June 2016

Shared memory parallelization is needed in any case!

Parallelism in Xpress 7.9

Parallelization of branch-and-bound algorithm

Parallel threads

Individual node-dives from active nodes of the branch-and-bound tree.

Work pool

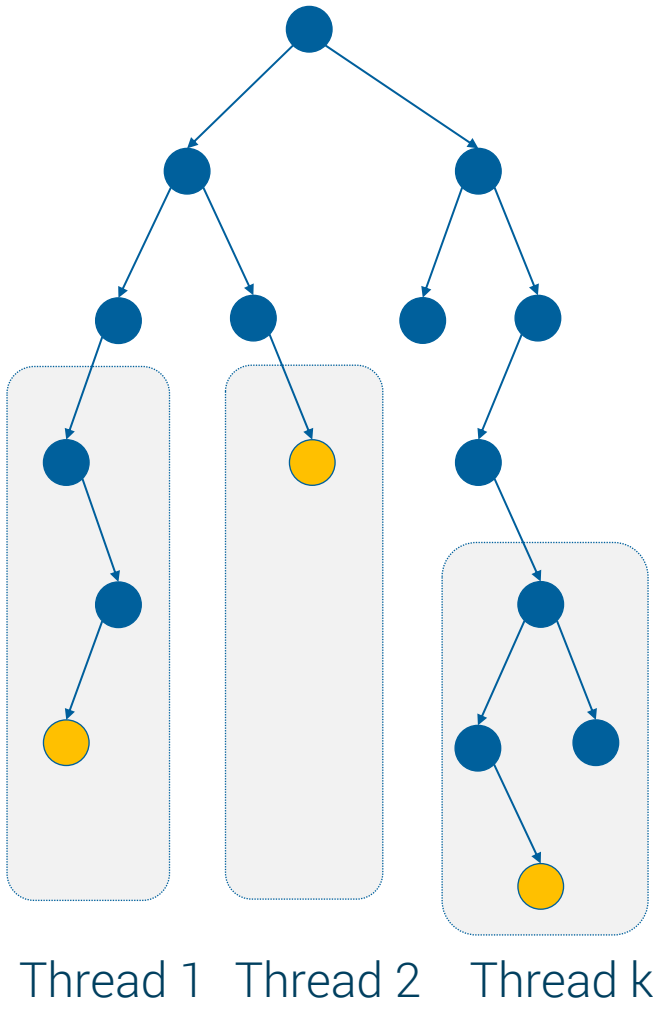
Central pool of all active nodes.

Synchronization

Exchange collected search data (nodes, solutions, statistics, …).

Deterministic Timer

Effort estimates that roughly track elapsed time.

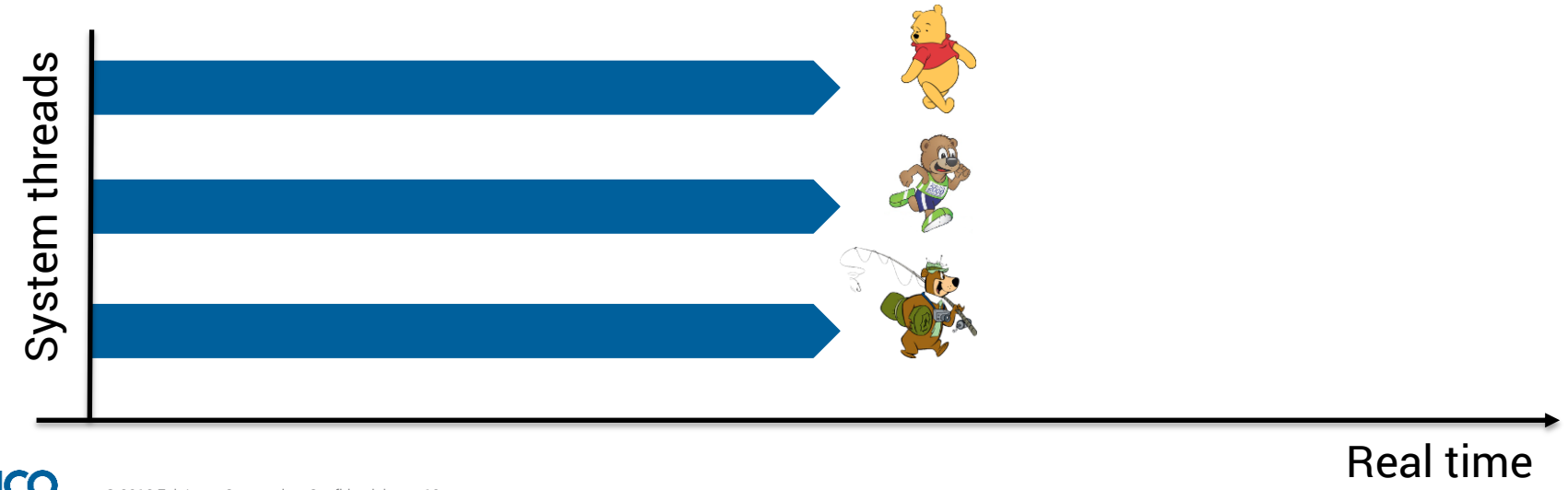


Parallelization bottlenecks

- Access to shared data
- Up-to-dateness of shared data
- Accuracy of effort counts

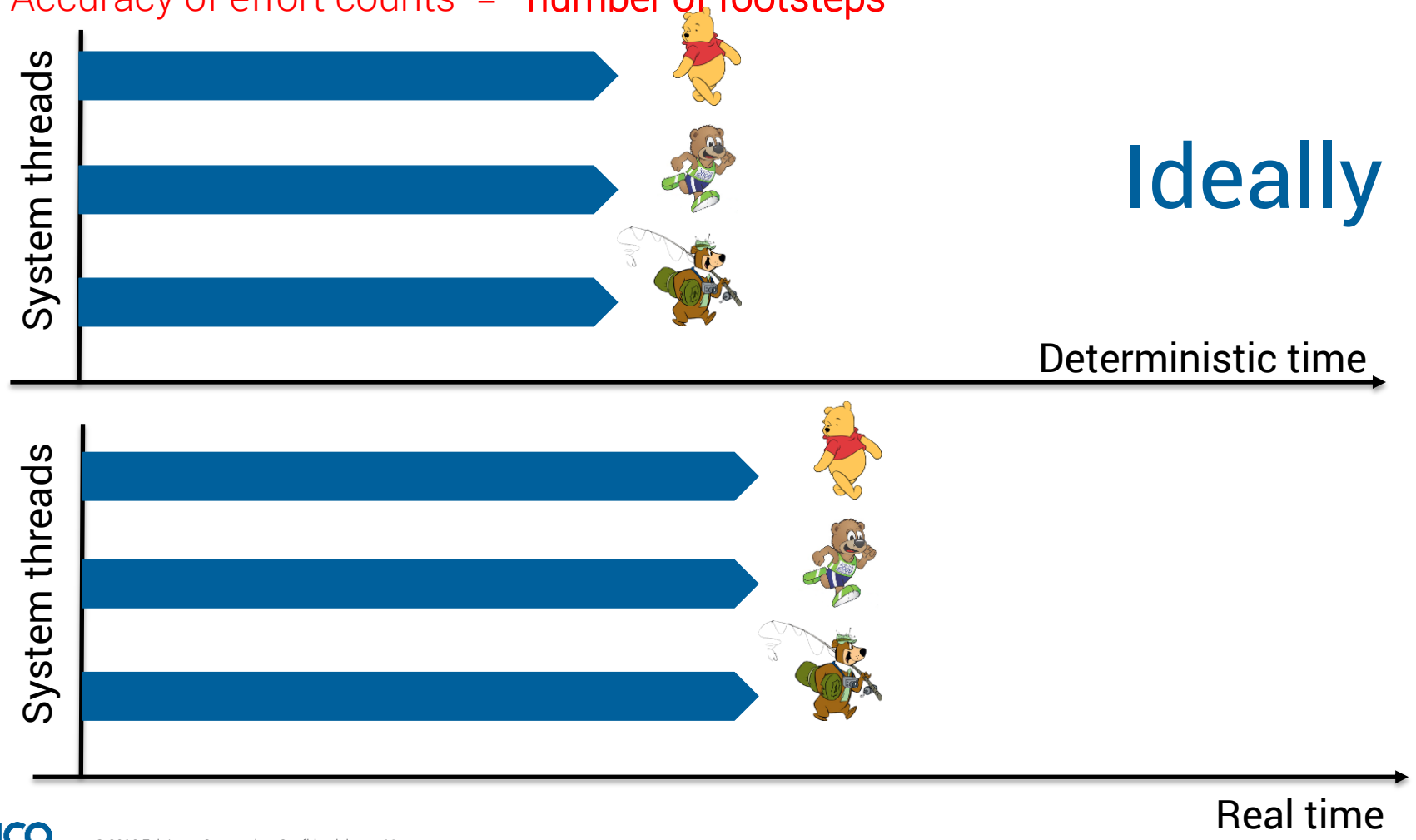
Parallelization bottlenecks

- Access to shared data
- Up-to-dateness of shared data
- Accuracy of effort counts



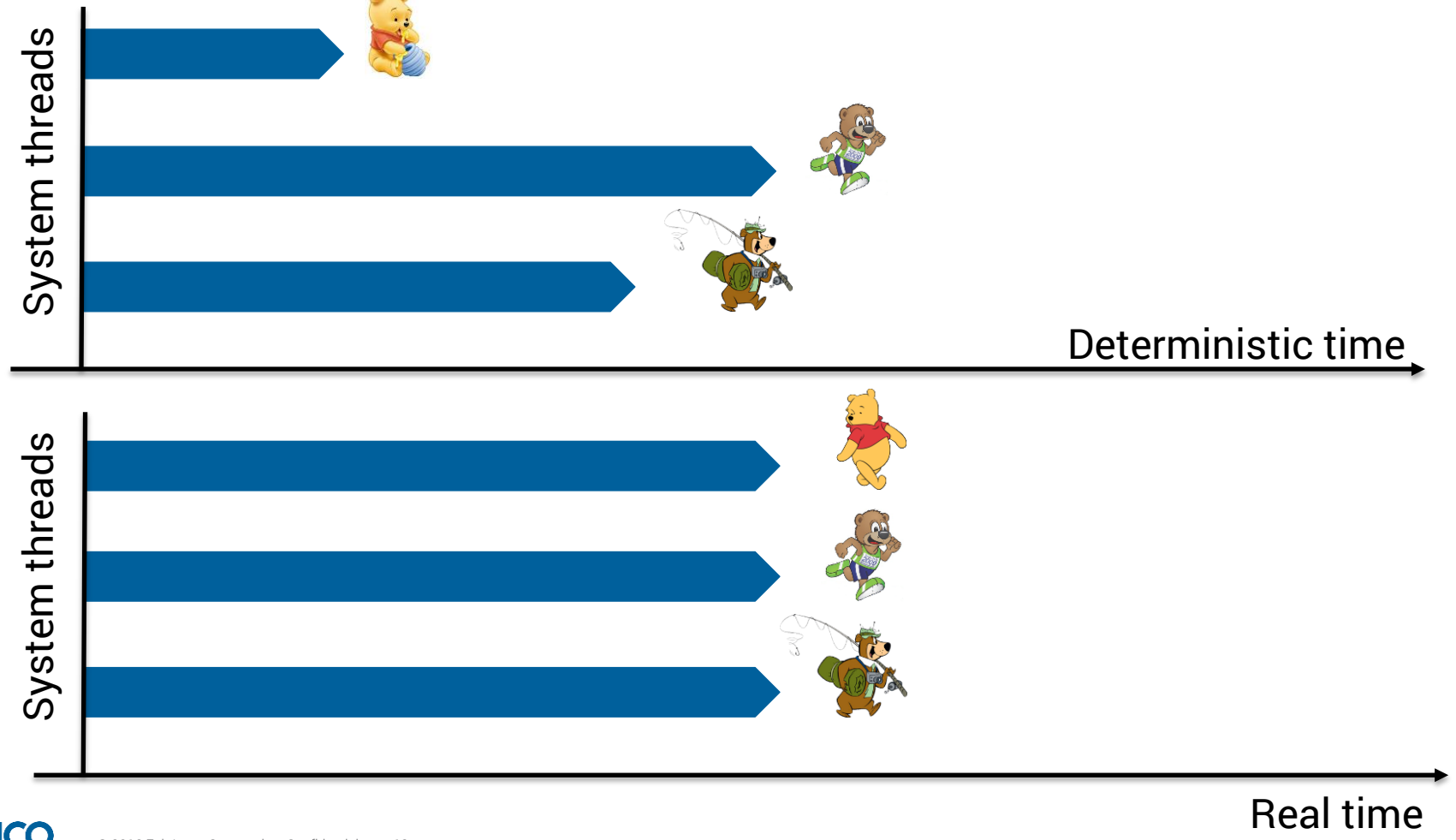
Parallelization bottlenecks

- Access to shared data
- Up-to-dateness of shared data
- Accuracy of effort counts = number of footsteps



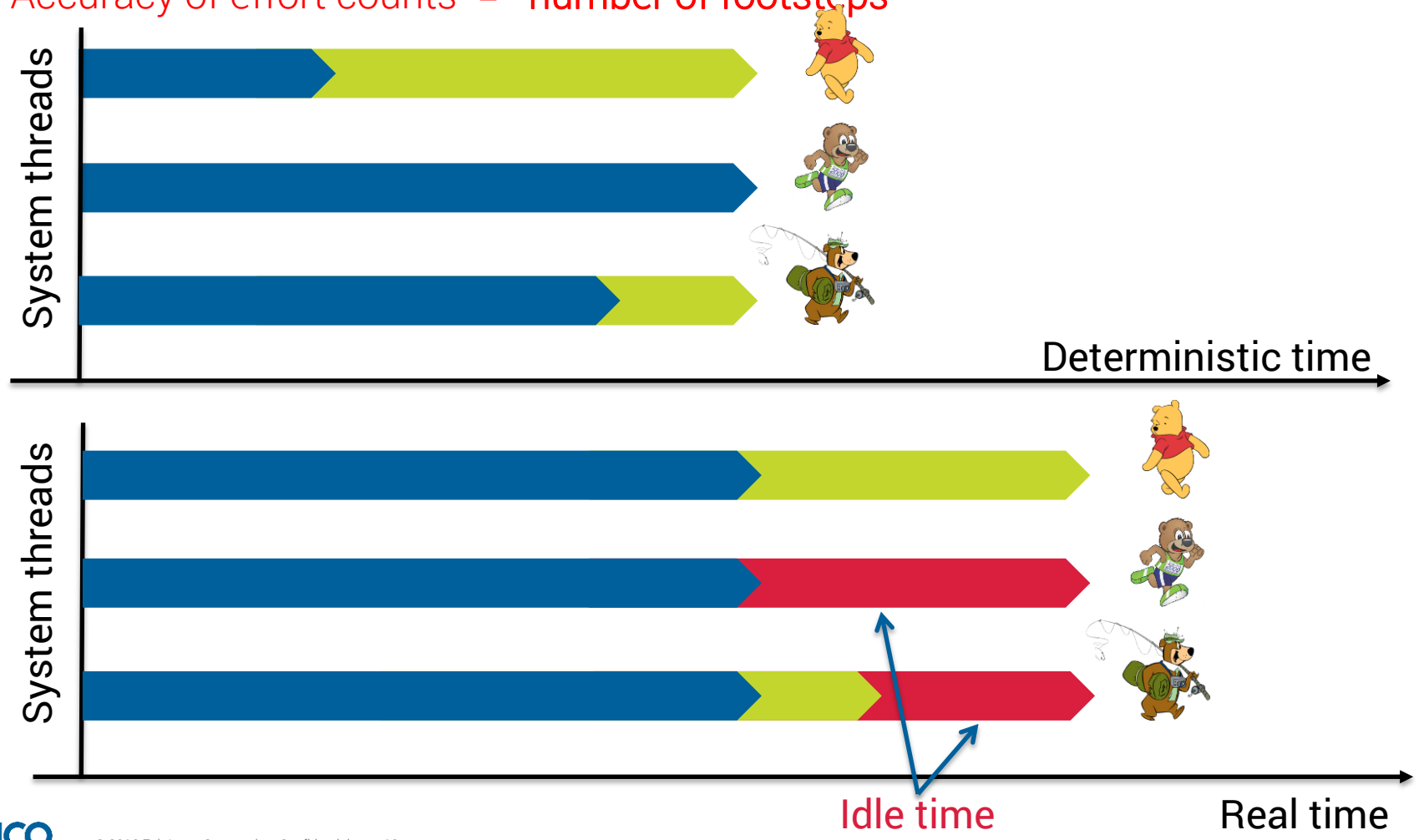
Parallelization bottlenecks

- Access to shared data
- Up-to-dateness of shared data
- Accuracy of effort counts = number of footsteps



Parallelization bottlenecks

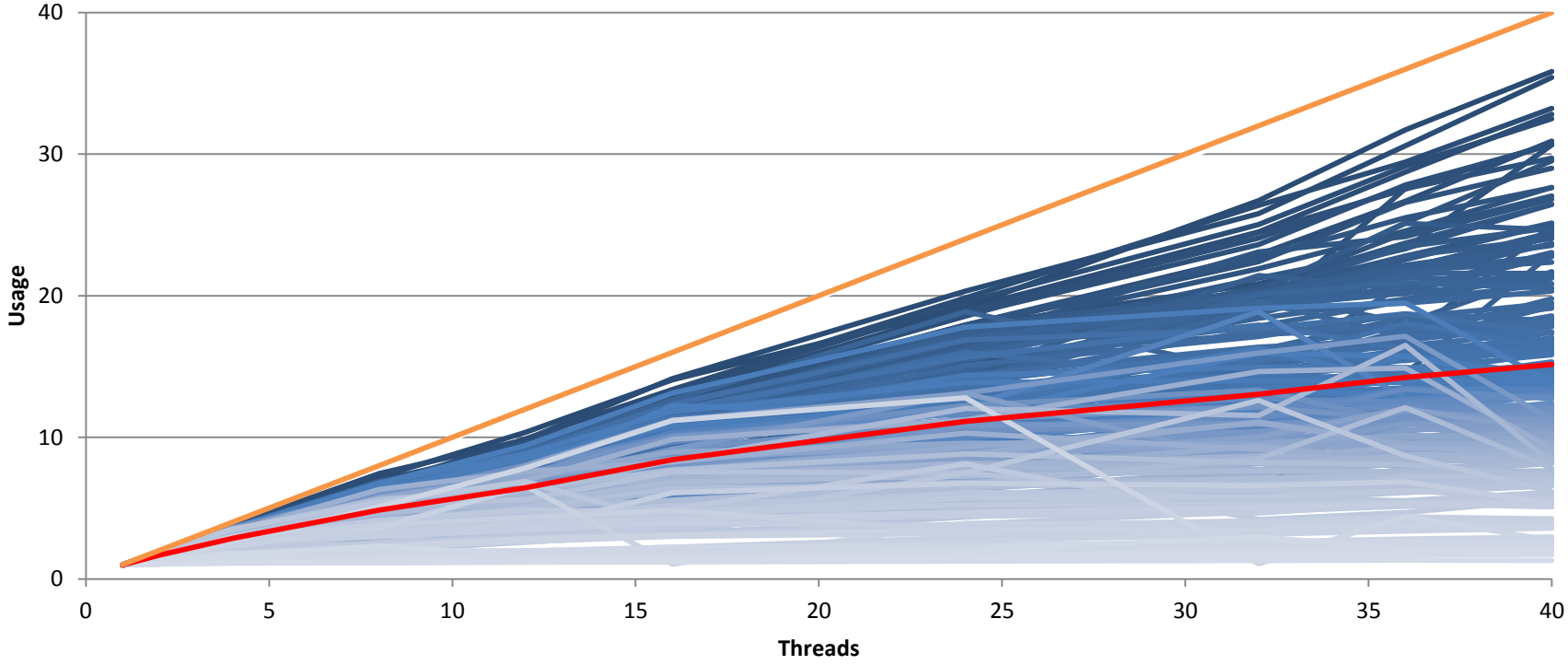
- Access to shared data
- Up-to-dateness of shared data
- Accuracy of effort counts = number of footsteps



Parallelism in Xpress 7.9

Workload MIPLIB 2010 Easy/Solvable (205 Instances)

Usage = Total Thread Time / Elapsed Time

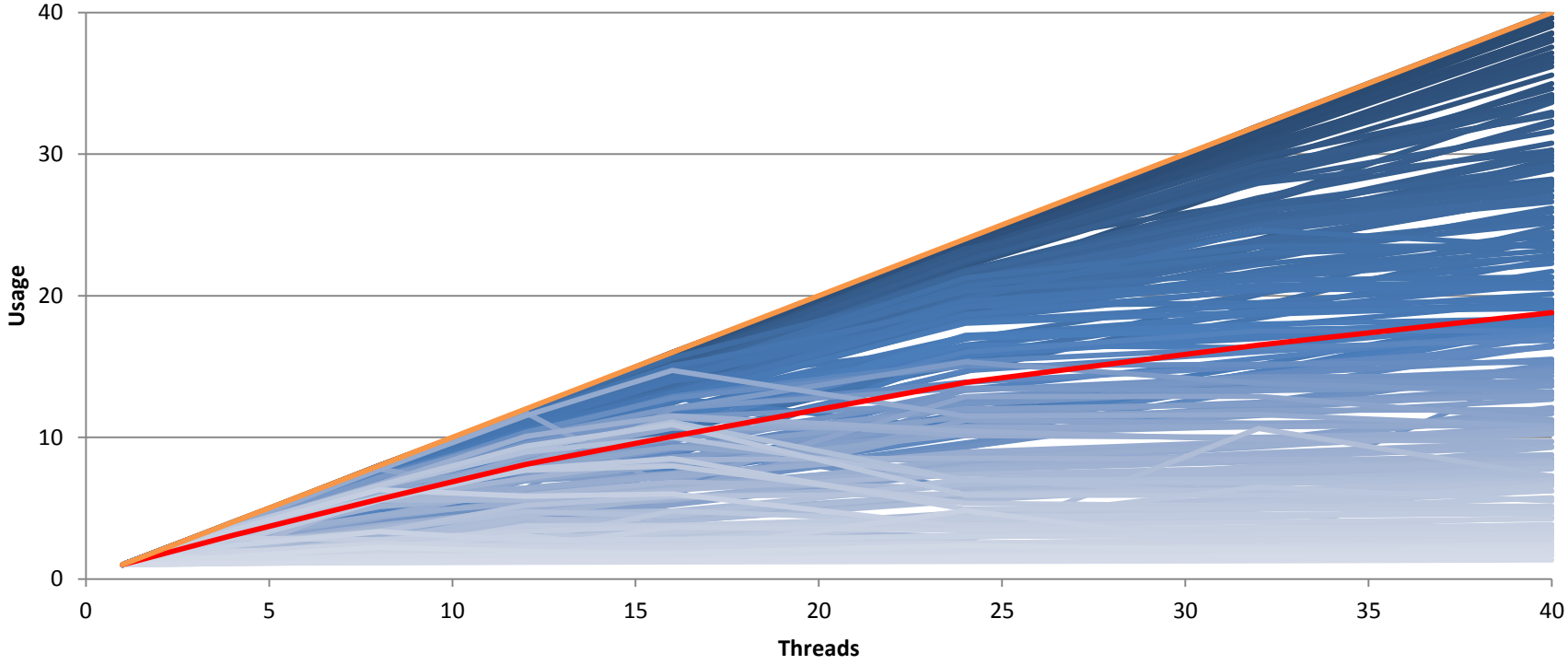


Threads	1	2	4	8	12	16	24	32	36	40
Usage	0.99	1.67	2.87	4.85	6.45	8.42	11.12	13.04	14.23	15.13

Parallelism in Xpress 8.0

Workload MIPLIB 2010 Easy/Solvable (205 Instances)

Usage = Total Thread Time / Elapsed Time

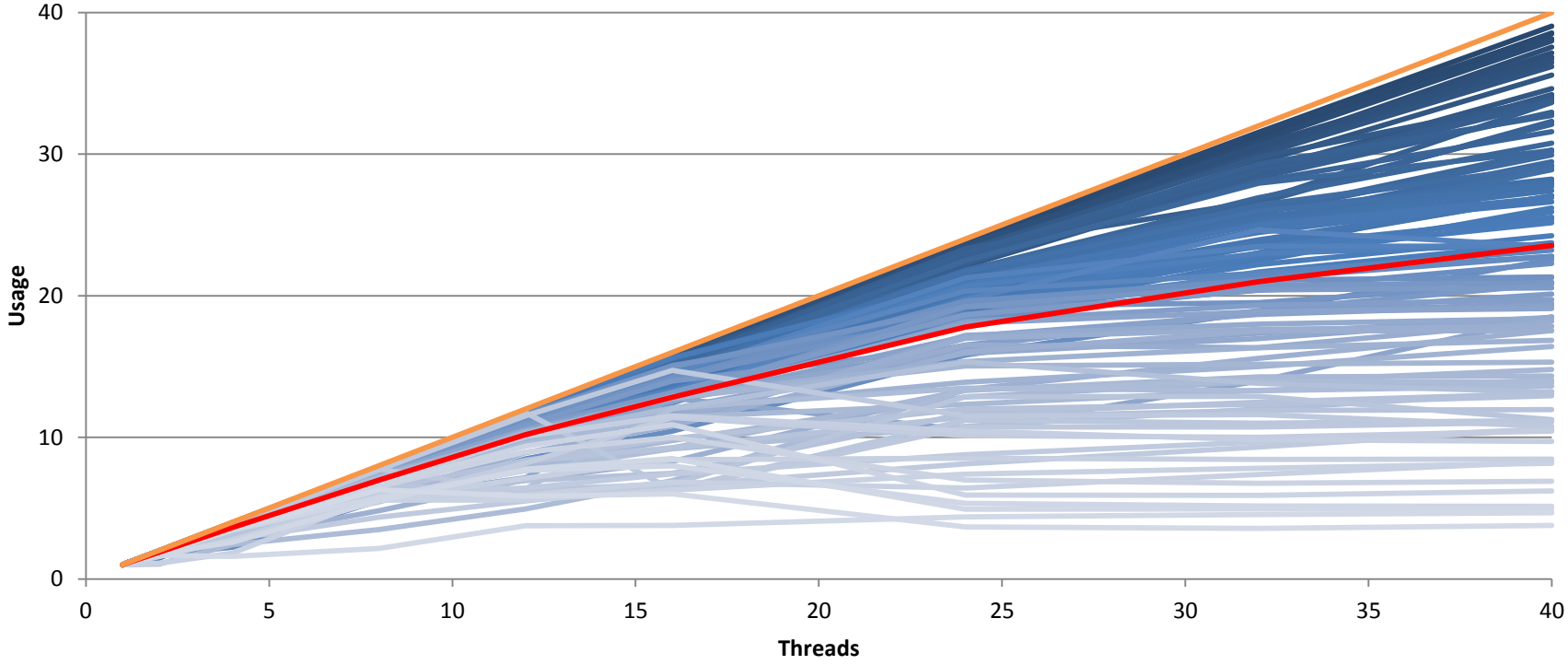


Threads	1	2	4	8	12	16	24	32	36	40
7.9	0.99	1.67	2.87	4.85	6.45	8.42	11.12	13.04	14.23	15.13
8.0	0.99	1.70	3.07	5.52	8.02	10.07	13.86	16.51	--	18.81

Parallelism in Xpress 8.0

Workload MIPLIB 2010 Easy/Solvable at least 10000 nodes (117 instances)

Usage = Total Thread Time / Elapsed Time



Threads	1	2	4	8	12	16	24	32	36	40
7.9	0.99	1.76	3.15	5.50	7.24	9.50	12.54	14.50	15.65	16.51
8.0	0.99	1.82	3.63	6.99	10.10	12.82	17.78	20.98	--	23.55

Parallelism in Xpress 8.0

Main goal: Decrease dependence on effort counters

1. Always have work available, produce more tasks than (system) threads
2. To decrease waiting time, all tasks access only partial information

Uses a general purpose Job Scheduler:

- Specifically designed for deterministic, parallel execution of jobs.
- Intended for any multi-threaded application with dynamic jobs and data exchanges.
- Automatic scheduling and execution of jobs.
- Handles communication and ensures data delivery is deterministic.
- Ensures events and read barriers are deterministic.
- Frees application developer from worrying about determinism.
- Scalable.

Parallel MIP built on top of the Job Scheduler

- Jobs = parallel heuristics or node solves or ..

Parallelism in Xpress 8.0 (cntd)

Main goal: Decrease dependence on effort counters

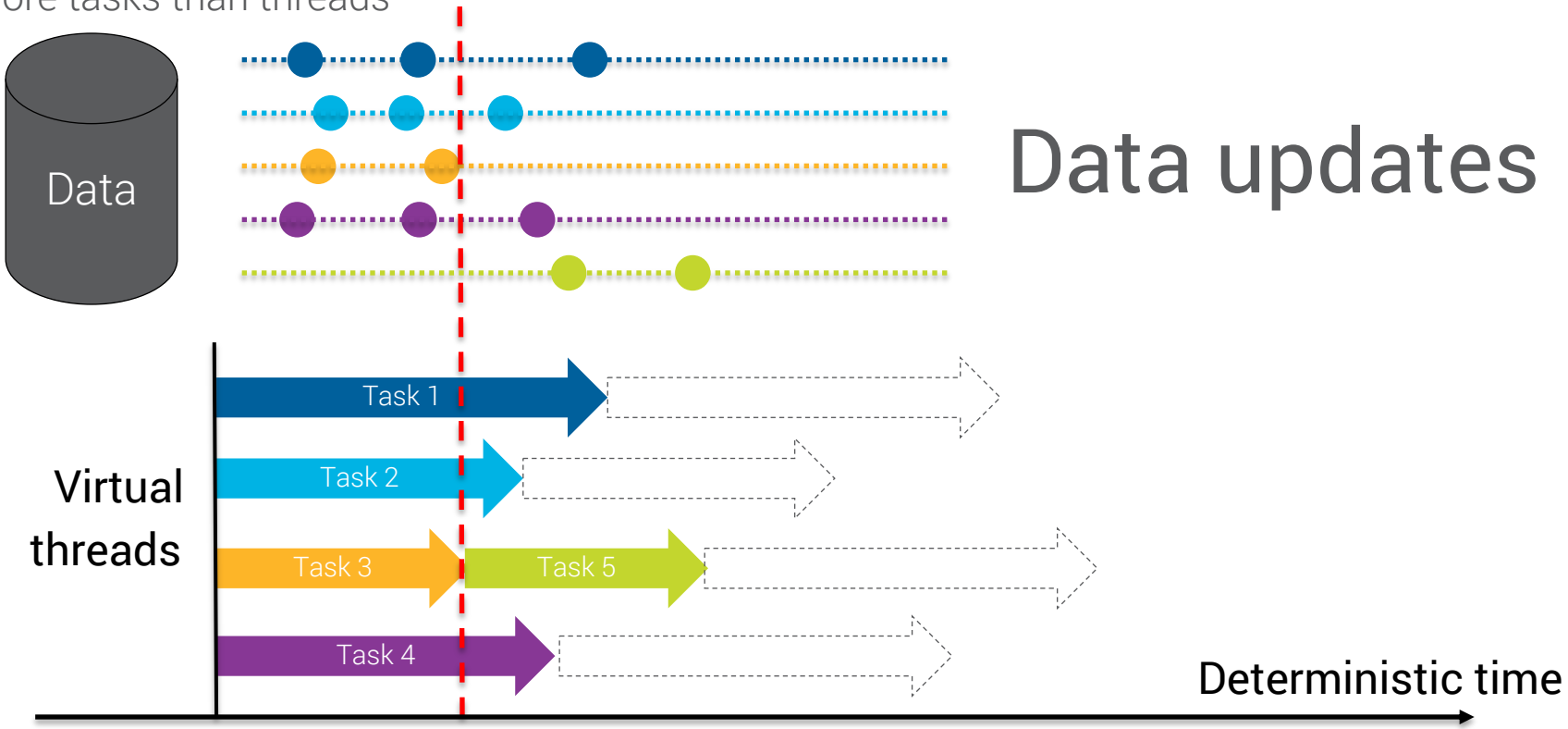
1. Always have work available, produce more tasks than (system) threads
2. To decrease waiting time, all tasks access only partial information

- Job Scheduler can be told to run in opportunistic mode.
  - Relaxes deterministic ordering of events.
  - No special handling required in the application.
- Sequential run = single-threaded opportunistic.
- Rough proportionality of effort counter with elapsed time sufficient. ±50% easily good enough.

Scheduler ensures scalability and proper CPU utilization.
Application only needs to provide sufficiently many jobs.

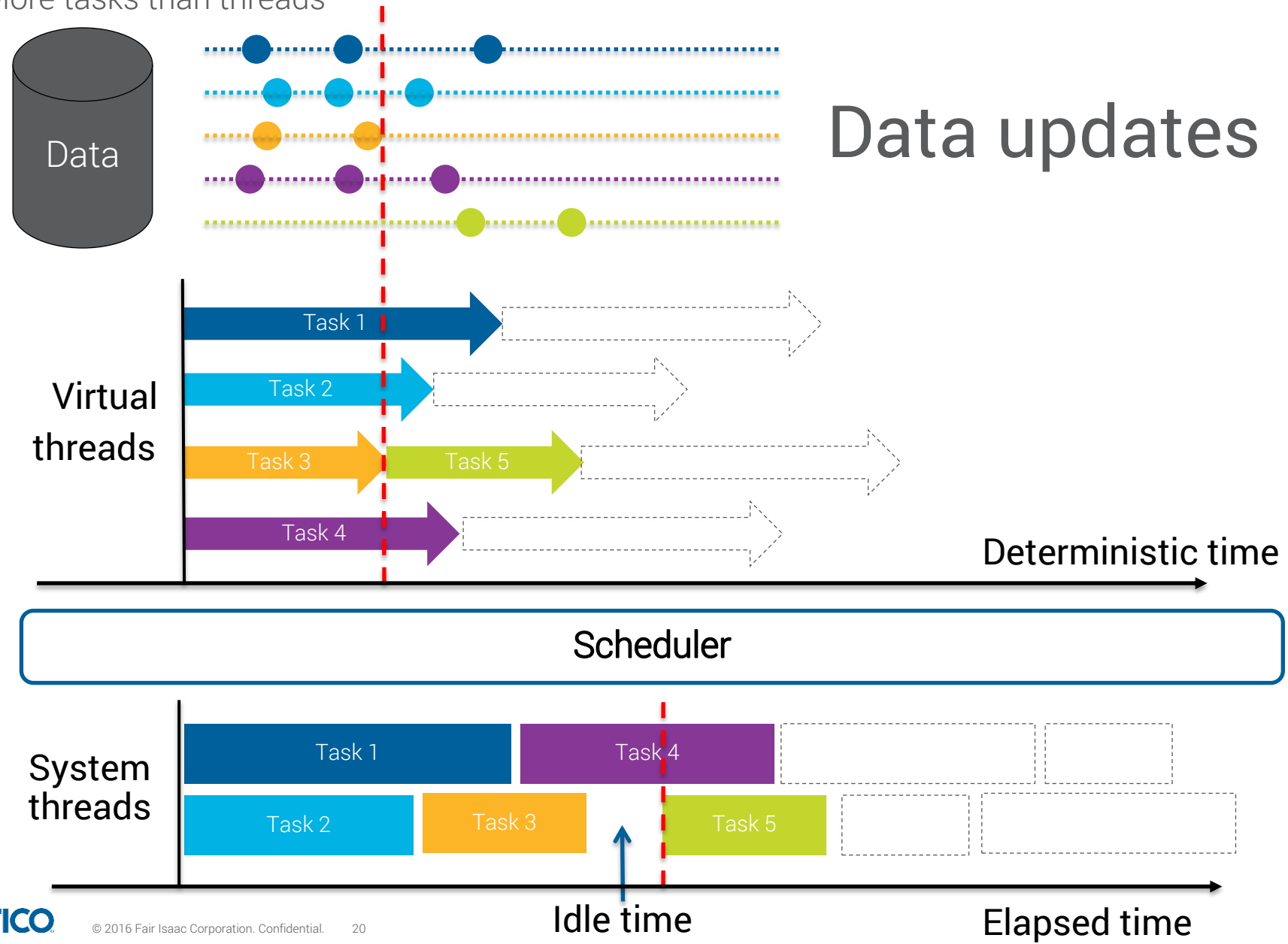
Parallelism visualization

More tasks than threads



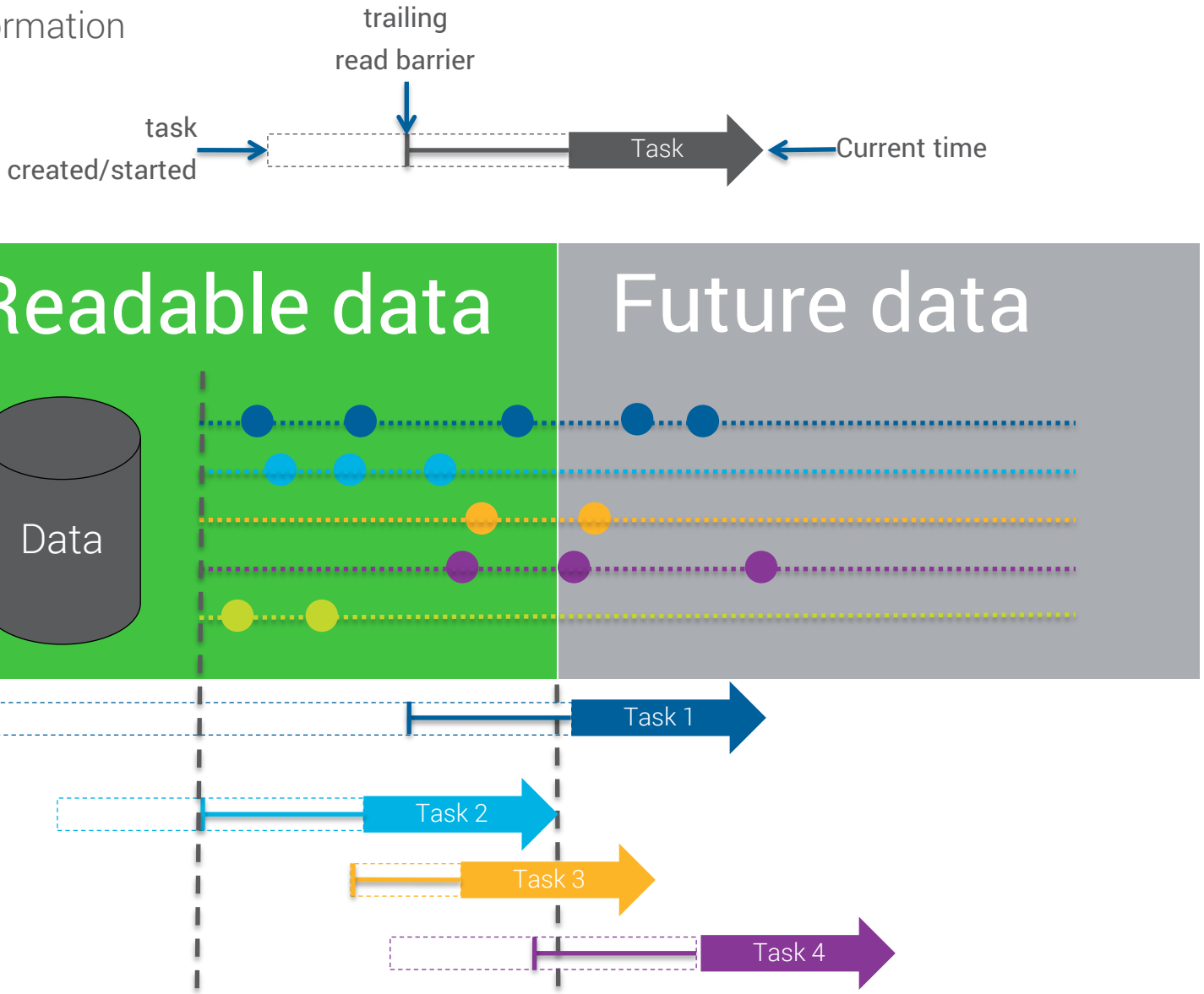
Parallelism visualization

More tasks than threads



Parallelism visualization

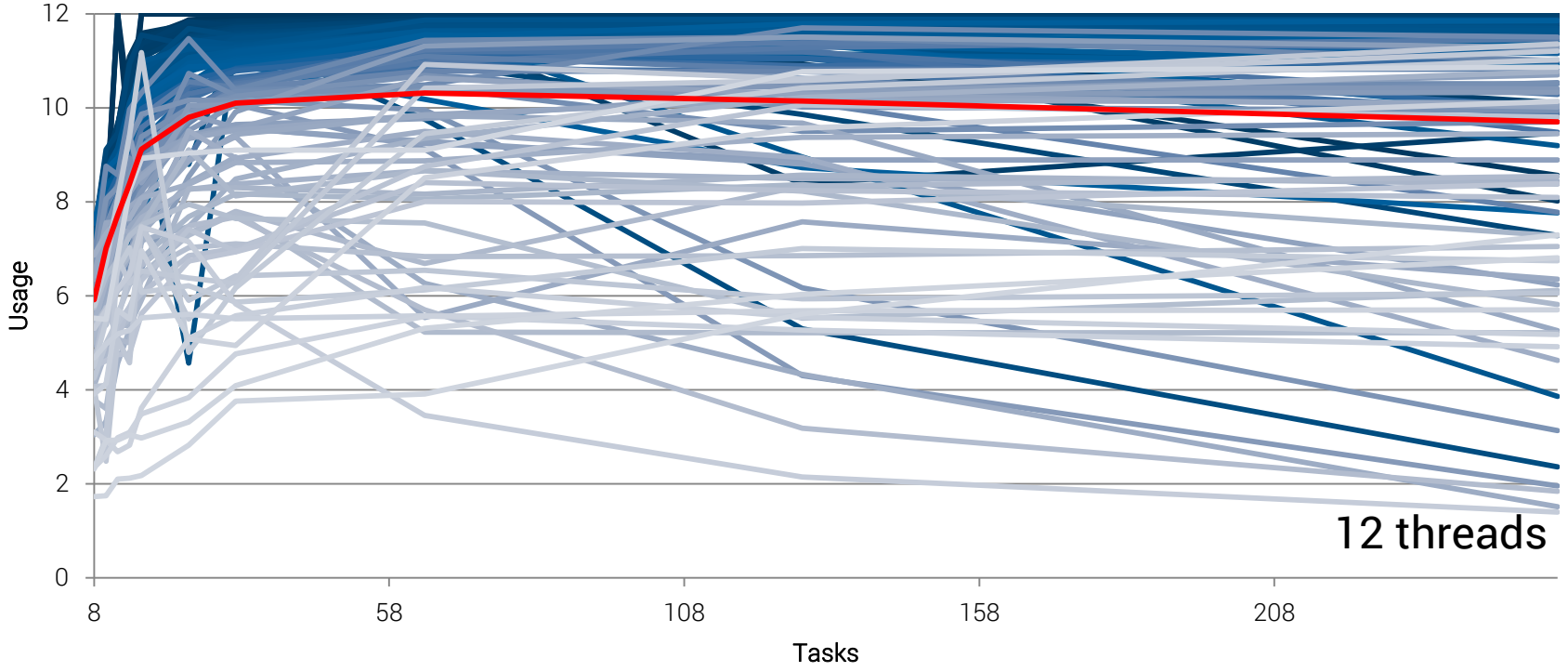
Partial information



Parallelism in Xpress 8.0

Workload MIPLIB 2010 Easy/Solvable at least 10000 nodes (123 instances)

Usage = Total Thread Time / Elapsed Time

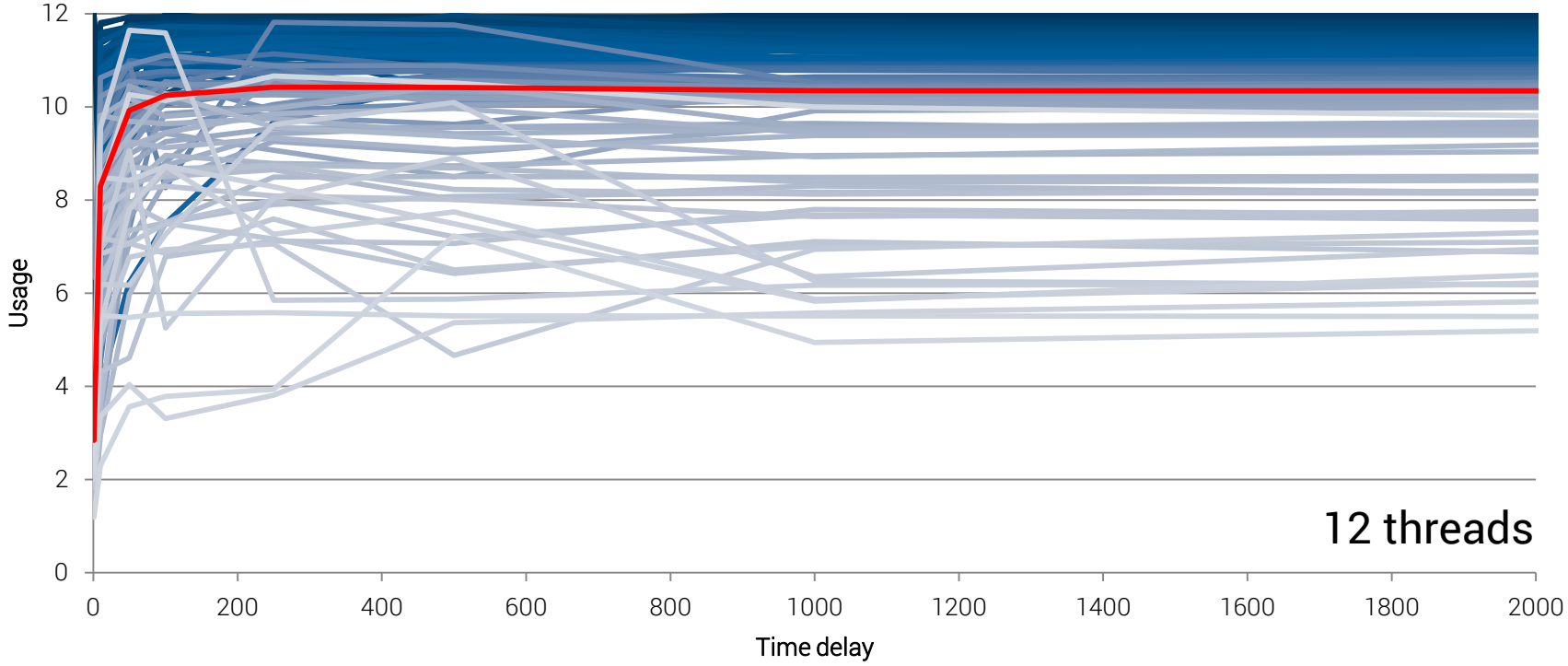


Tasks	8	10	12	14	16	24	32	64	128	256
Usage	5.92	7.00	7.71	8.38	9.12	9.80	10.10	10.31	10.14	9.70
Speed	1.00	1.16	1.27	1.16	1.33	1.41	1.39	1.41	1.28	1.09

Parallelism in Xpress 8.0

Workload MIPLIB 2010 Easy/Solvable at least 10000 nodes (117 instances)

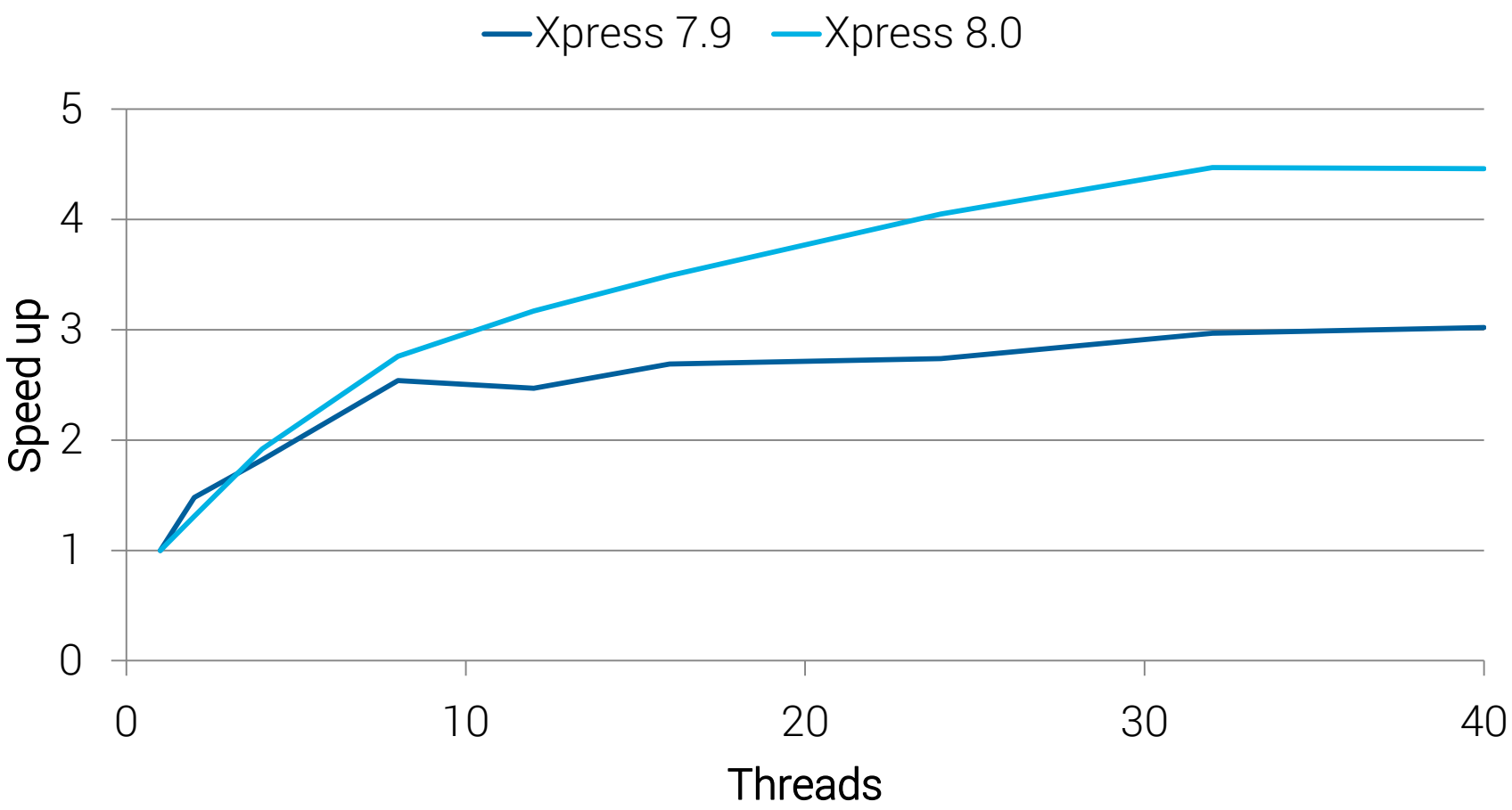
Usage = Total Thread Time / Elapsed Time



Delay	1	10	50	100	250	500	1000	5000	10000	Inf
Usage	2.84	8.30	9.92	10.24	10.42	10.41	10.34	10.32	10.34	10.36
Speed	1.00	1.60	1.89	2.33	2.04	2.11	2.31	2.36	2.30	2.29

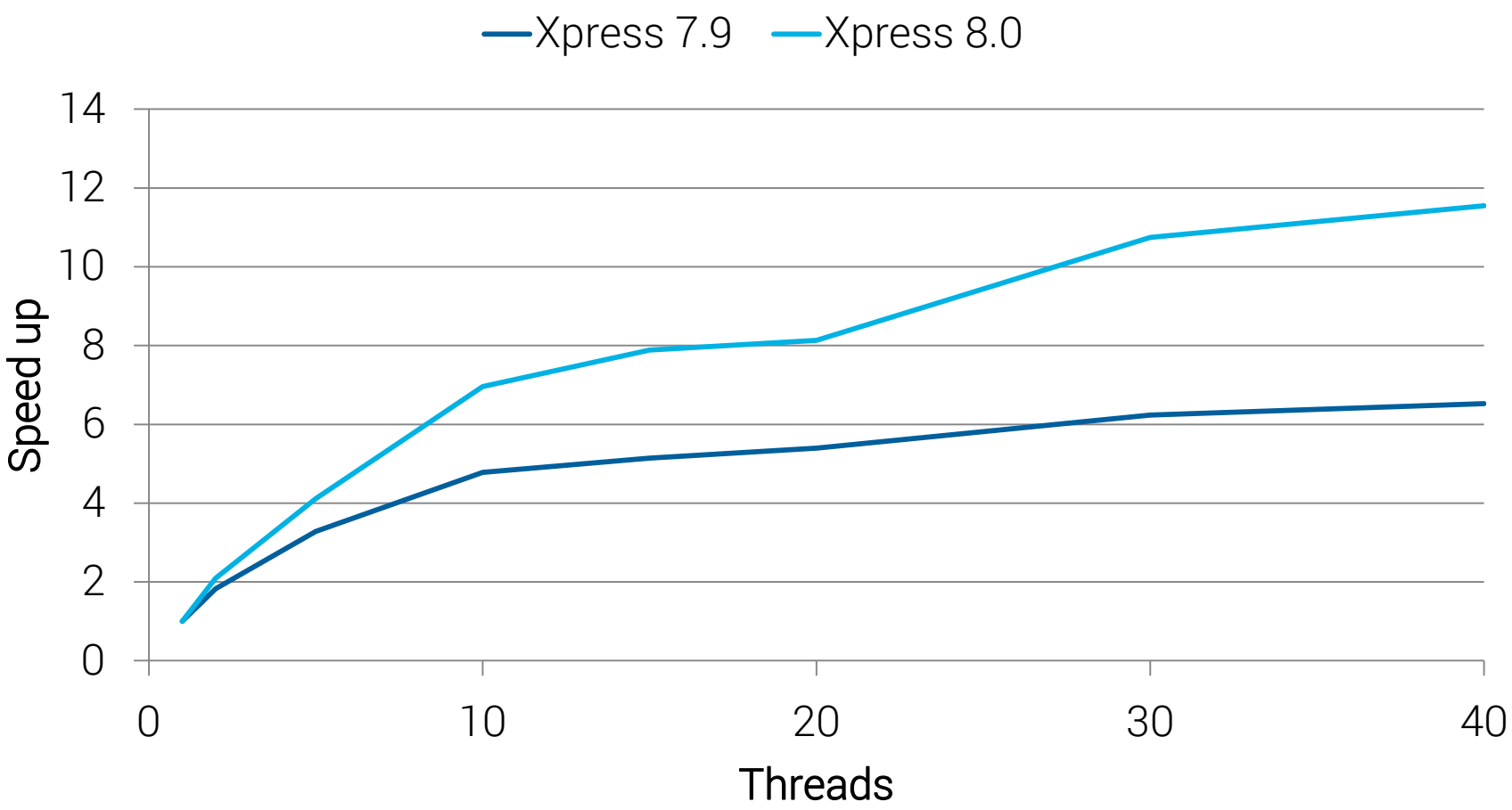
Parallel speed up Xpress 7.9 versus 8.0

MIPLIB 2010 Easy/Solvable at least 10000 nodes



Parallel speed up Xpress 7.9 versus 8.0

Internal parallel test set (152 instances)



Conclusion

New Parallel Job Scheduler

- Weak dependence on effort counters
- Use a trailing read barrier for each task → partial information
- Create more tasks than threads

Solution path is

- Platform independent
- System threads independent (depends on the number of tasks)

Reproducibility and Reliability

Any multi-threaded run can be reproduced as a single-threaded run!

FICO Decisions

Thank You!

Timo Berthold
Fair Isaac Germany GmbH