Implications of shared data synchronization techniques on multi-core energy efficiency

Ashok Gautham[ɣ], Kunal Korgaonkar[ɣϕ], Patanjali Slpsk[ɣ], Shankar Balachandran[ɣ], Kamakoti Veezhinathan[ɣ]

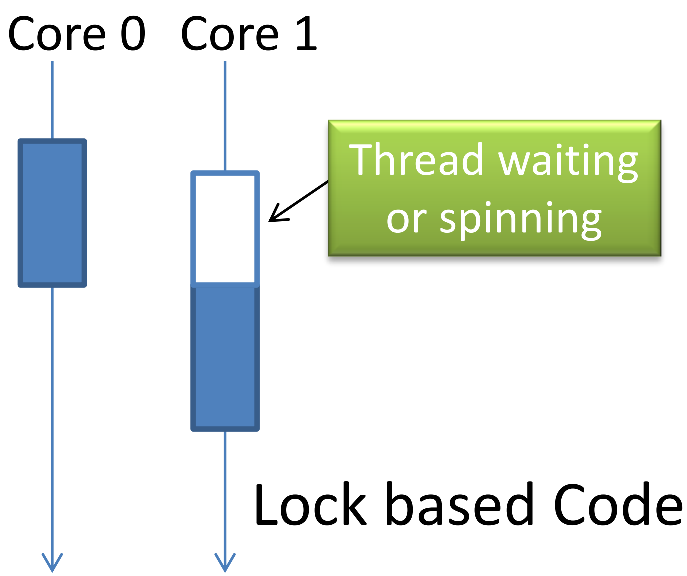
[ɣ] IIT Madras, Chennai, India

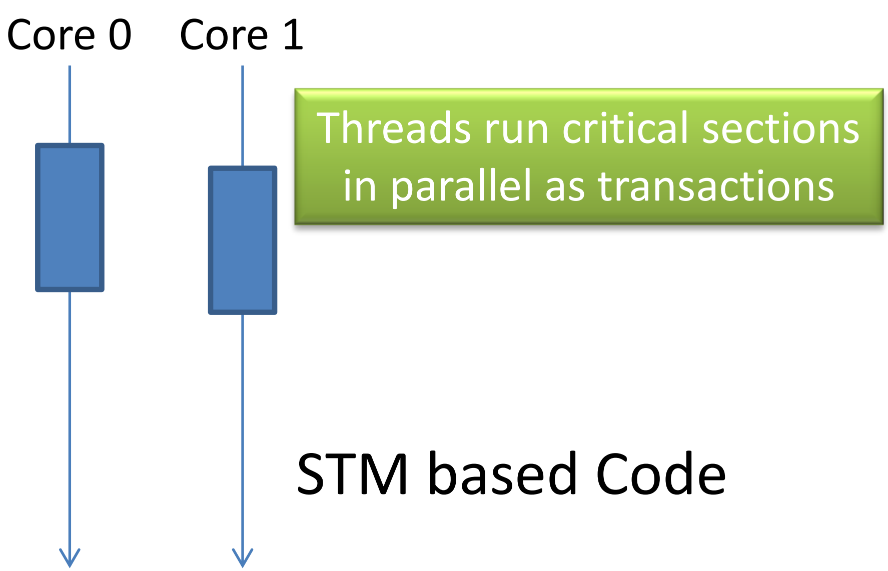
[ϕ] IBM Research, Bangalore, India

HOTPOWER 2012

What We Have Done

- We evaluated
  - Mutex, Spinlock and Software Transactional memory (STM) on a commodity multi-core
  - Sequential implementation also considered
- Synchronization techniques differ based on how they execute Critical Sections (CSs)





Glimpse of Results

- Metrics used - Performance Per Watt (PPW), Energy Delay Product (EDP), and not just Performance

<u>Results:</u>

- The extent of savings by using STM (for one of the benchmarks) over best performing lock-based code
  - 224% speedup
  - 653% reduction in EDP
  - 213% improvement in PPW

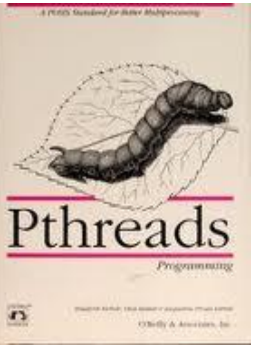
Behavior and Trends

	STM	Mutex	Spinlock	Sequential
Cores Utilization	All cores kept busy at all times	Cores are underutilized; Presence of idle periods	Cores kept busy, but spinning (wasteful cycles)	Only one core used
Core Sleep State Usage (energy saving states)	None	**Power saving sleep states are used, but not the deepest sleep state**	None	**Other cores in deepest sleep state at all times**
Duration	Short	Long	Long	Long
Power	High	Low	High	Low
Energy	Low	Still high	High	Low

Experimental setup

Linux – OS
(with core sleep state manager/governor)

Intel Sandy Bridge
Commodity Processor
With per-core sleep state support.



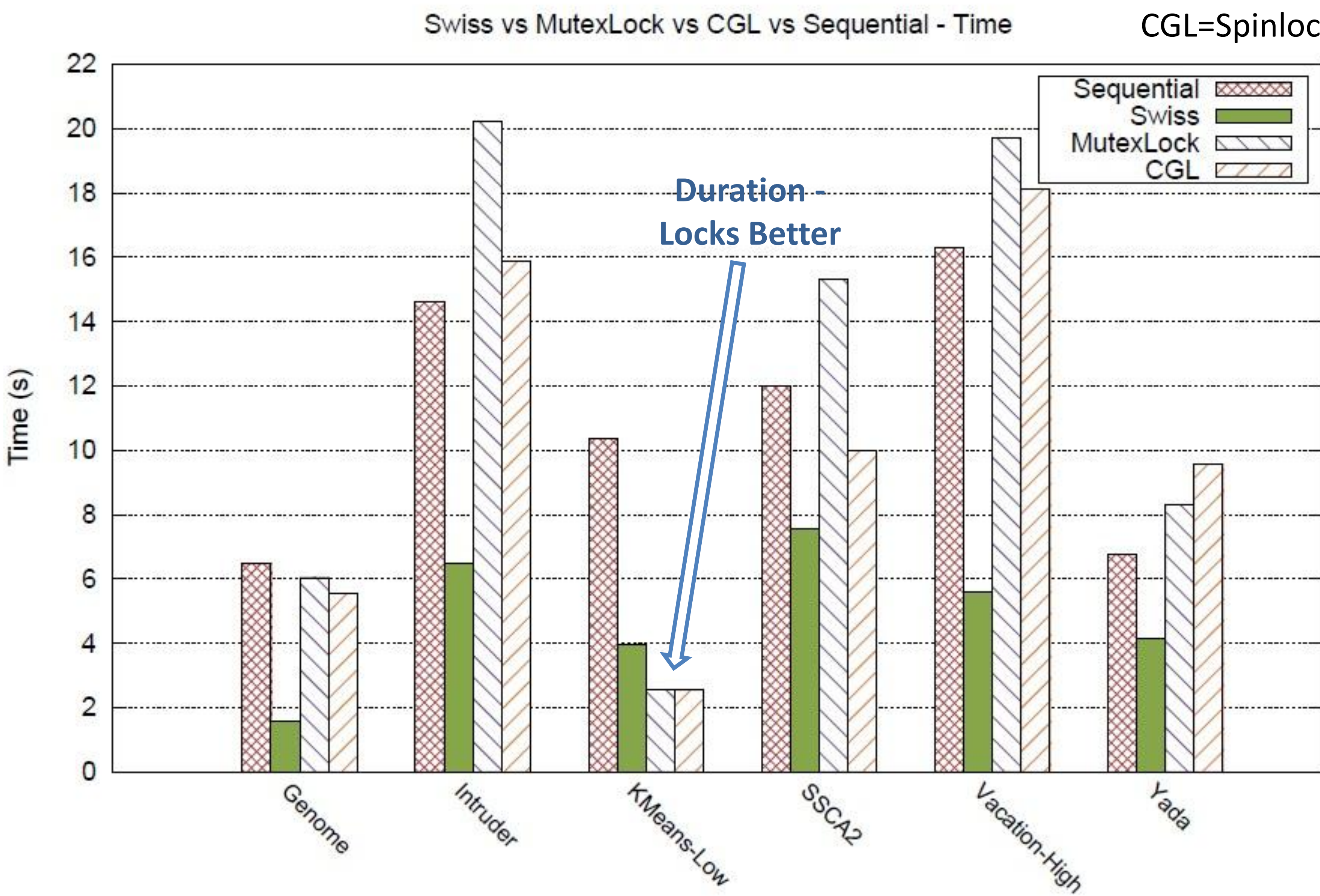
Standard Pthread
Library for Mutex

State of the art STM – Swiss TM
Experimentally proven to be best

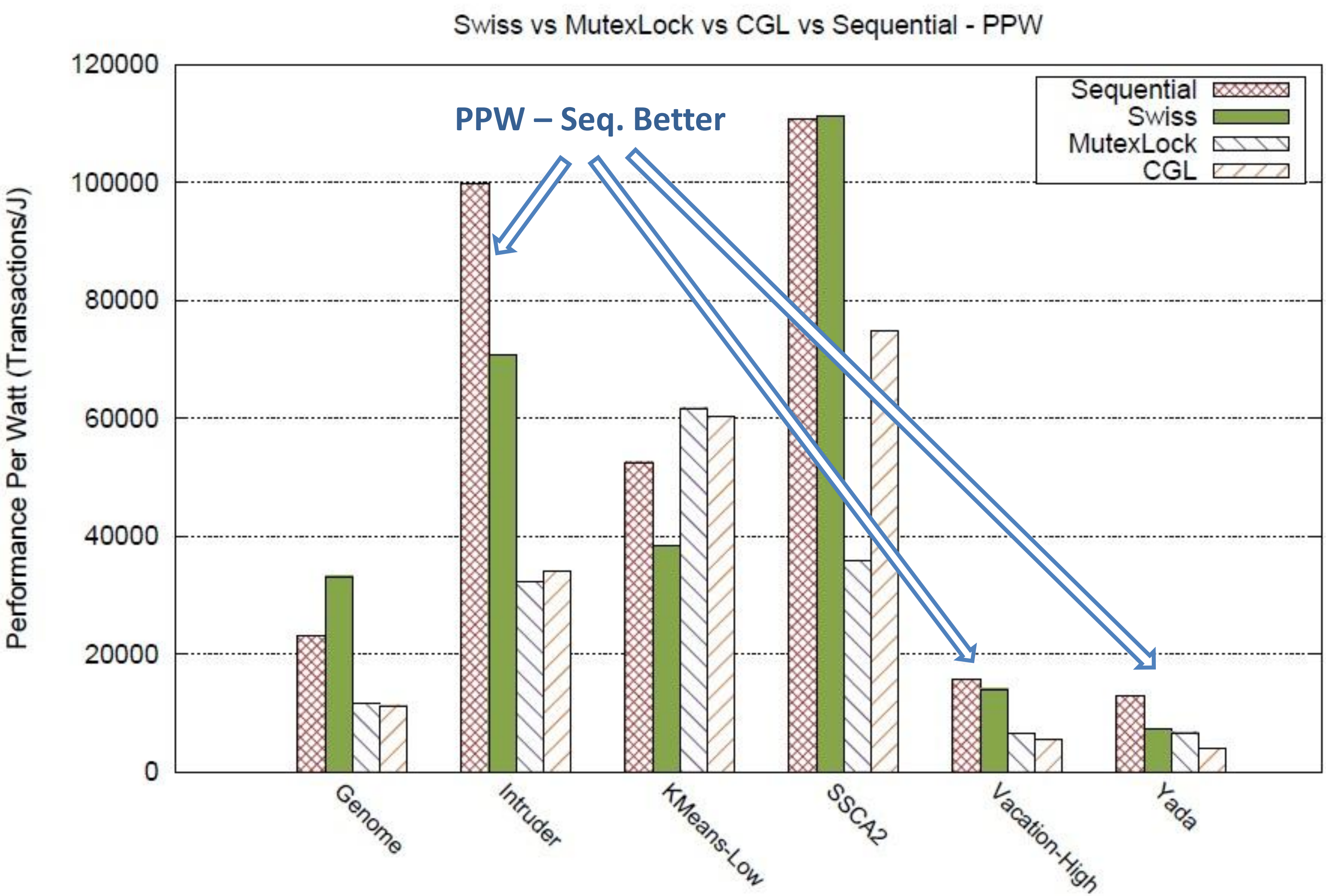
STAMP TM Benchmark
Range of ‘time inside CS’ – 2% to 90%

Duration = Performance

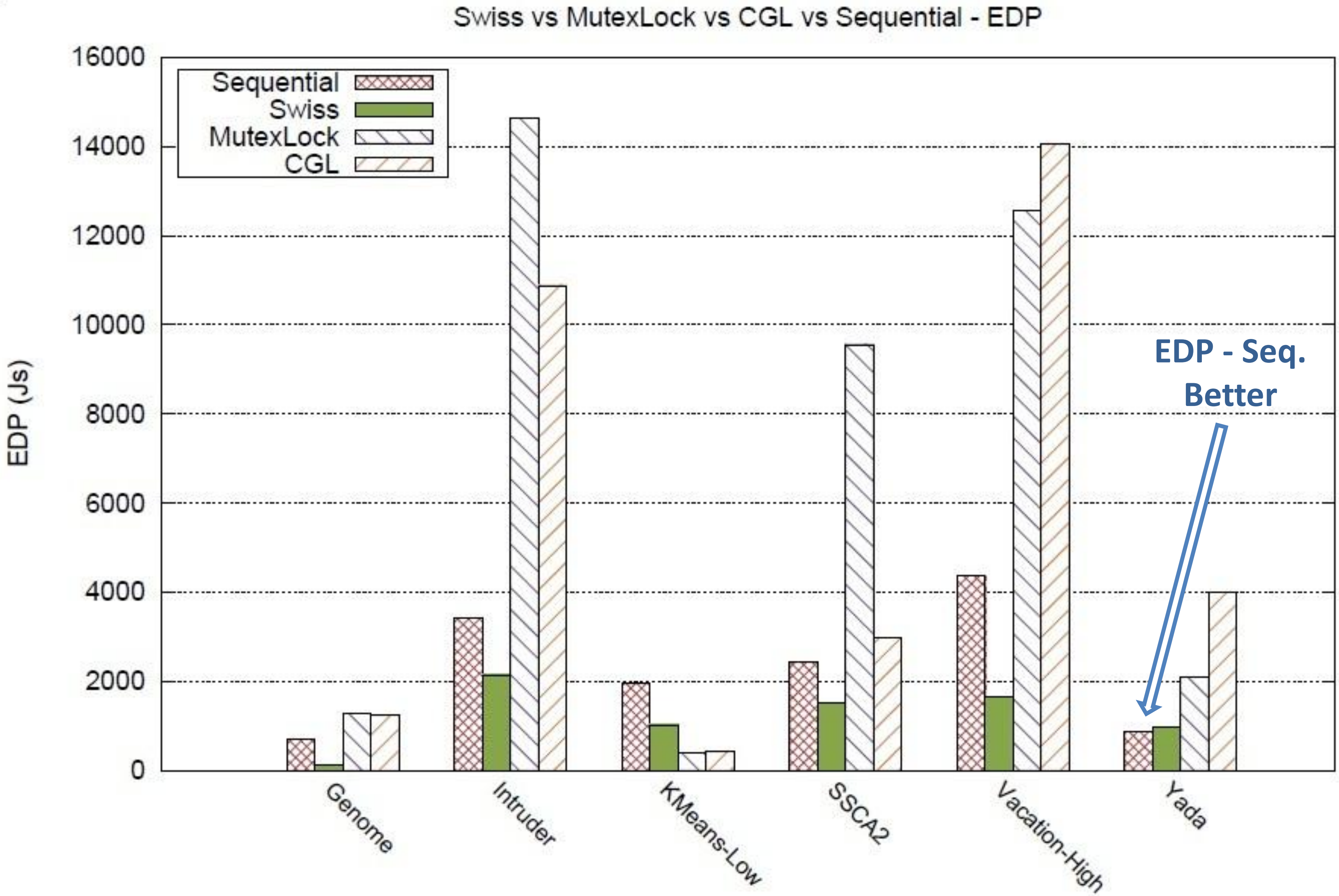
CGL=Spinlock



PPW = Energy


Swiss vs MutexLock vs CGL vs Sequential - PPW
Performance Per Watt (Transactions/J)
120000
100000
80000
60000
40000
20000
0
Sequential
Swiss
MutexLock
CGL
PPW – Seq. Better
Genome
Intruder
KMeans-Low
SSCA2
Vacation-High
Yada

EDP (= Performance and Energy)



Trading off Performance for Reduction in Avg. and Peak Power

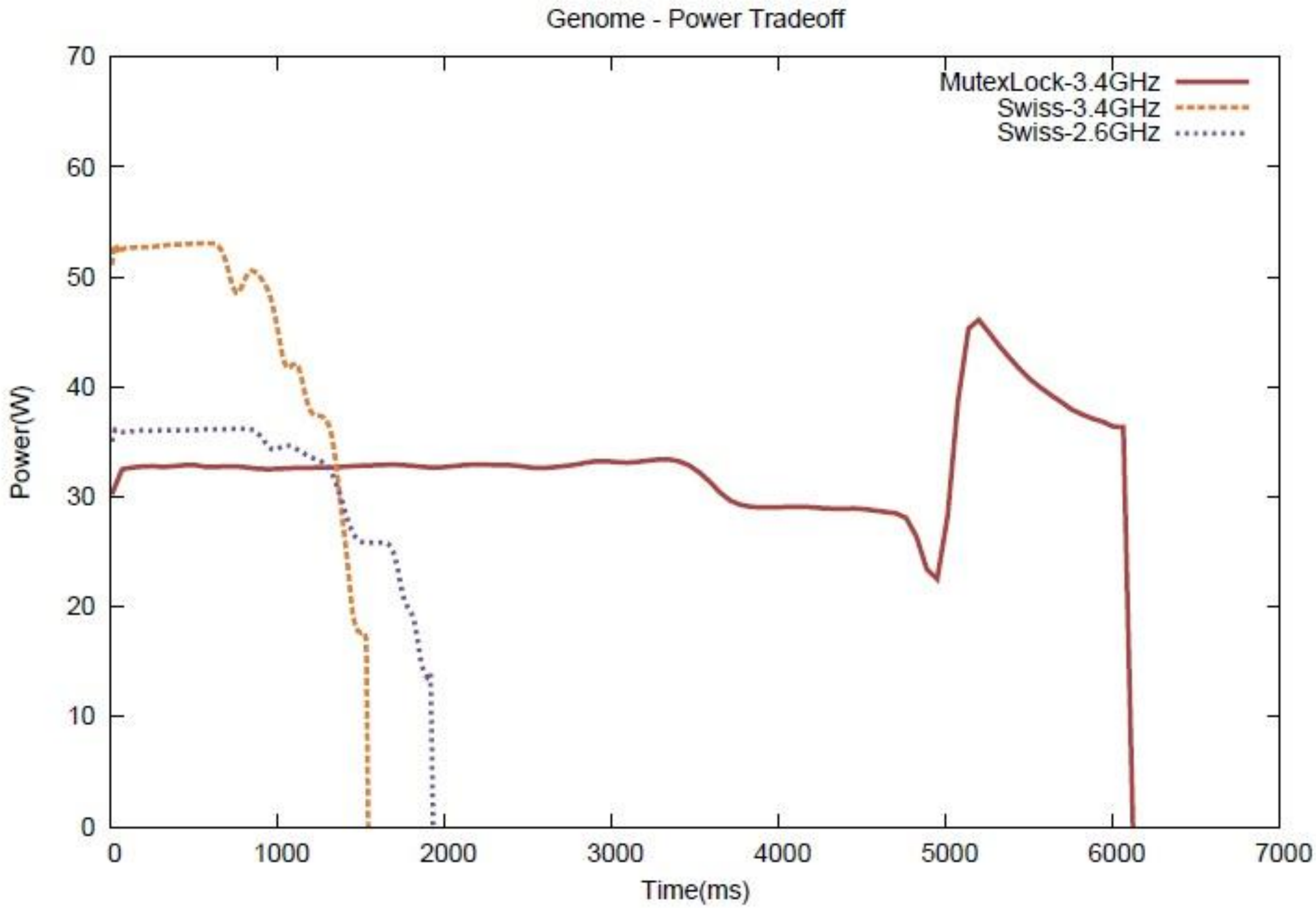


Figure 3: STM Performance-Energy trade-off

Future Work

- We plan to cover a wider set of benchmarks
  - One of the questions we want to answer - Does there exist applications for which both lock and STM scale, but STM scale better?
- We also want to check how programs using mutex be made more energy efficient
  - by exploiting core 'sleep states' in a more fine grained and controlled manner
- For a program using STM, conflict rate still is a key workload parameter dictating the resultant performance and energy
  - A thorough quantitative analysis of such workload parameters is needed

Questions