TÉCNICO
LISBOA

Green-CM: Energy efficient contention management for Transactional Memory

Shady Alaa

Paolo Romano – INESC-ID/IST

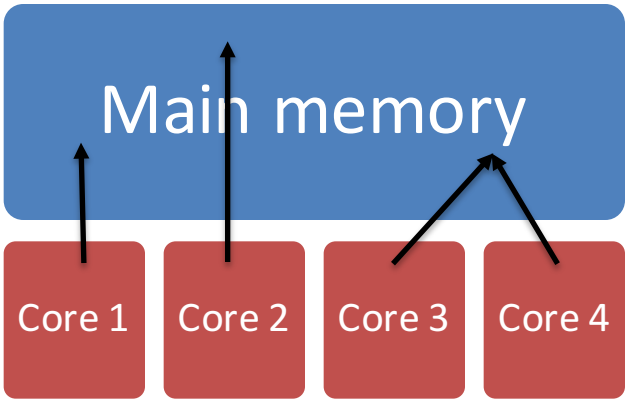
Mats Brorsson - KTH

Agenda

- Introduction
- Related work
- Architecture
- Green-CM
- Evaluation
- Conclusion

Introduction

- Multicores are everywhere
  - Complex programming
    - Locks
    - Deadlocks
  - Transactional memory
    - Atomics blocks
    - Transparent from programmer




```
atomic{
    if(bal>amount)
        withdraw(amount);
}
```

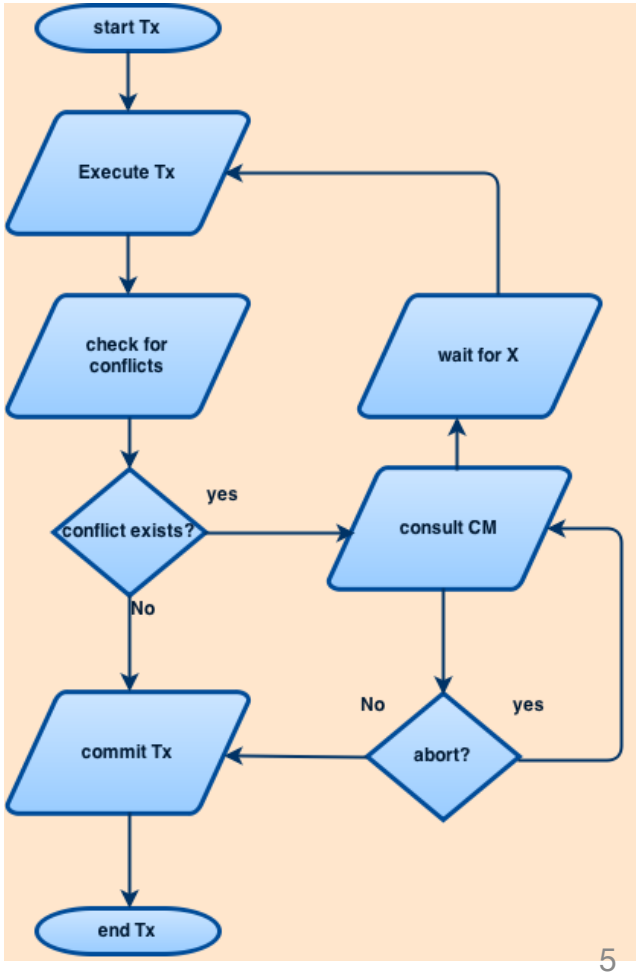
Introduction

- Energy efficiency
  - First order design choice
  - Battery based devices
  - Data centers

- Goal
  - Energy efficient transactional memory in terms of both energy and performance

Introduction

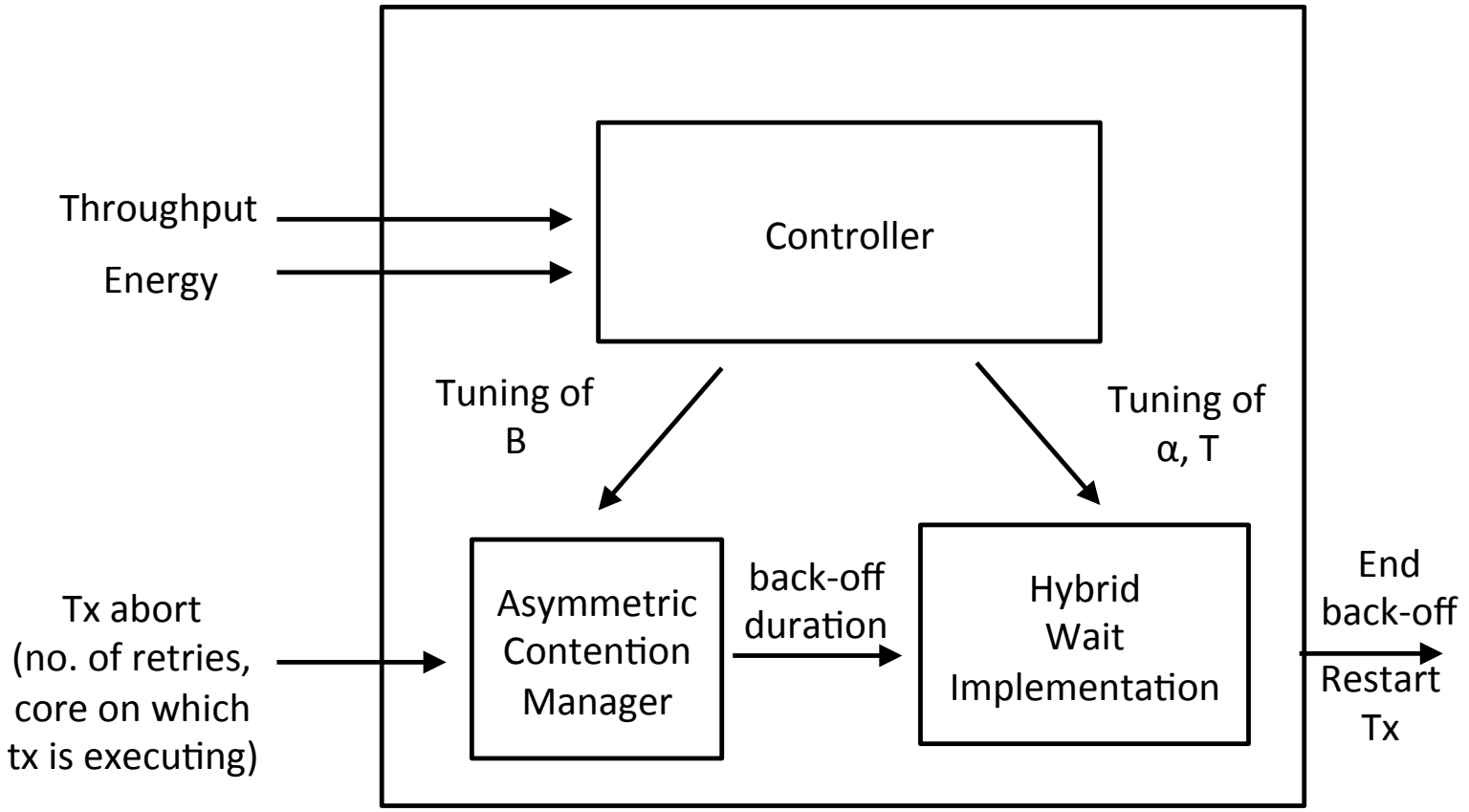
- Contention Manager
  - minimize contention
  - which transaction to abort
  - when to restart an aborted transaction
- Energy efficiency:
  - wait implementation
  - DVFS



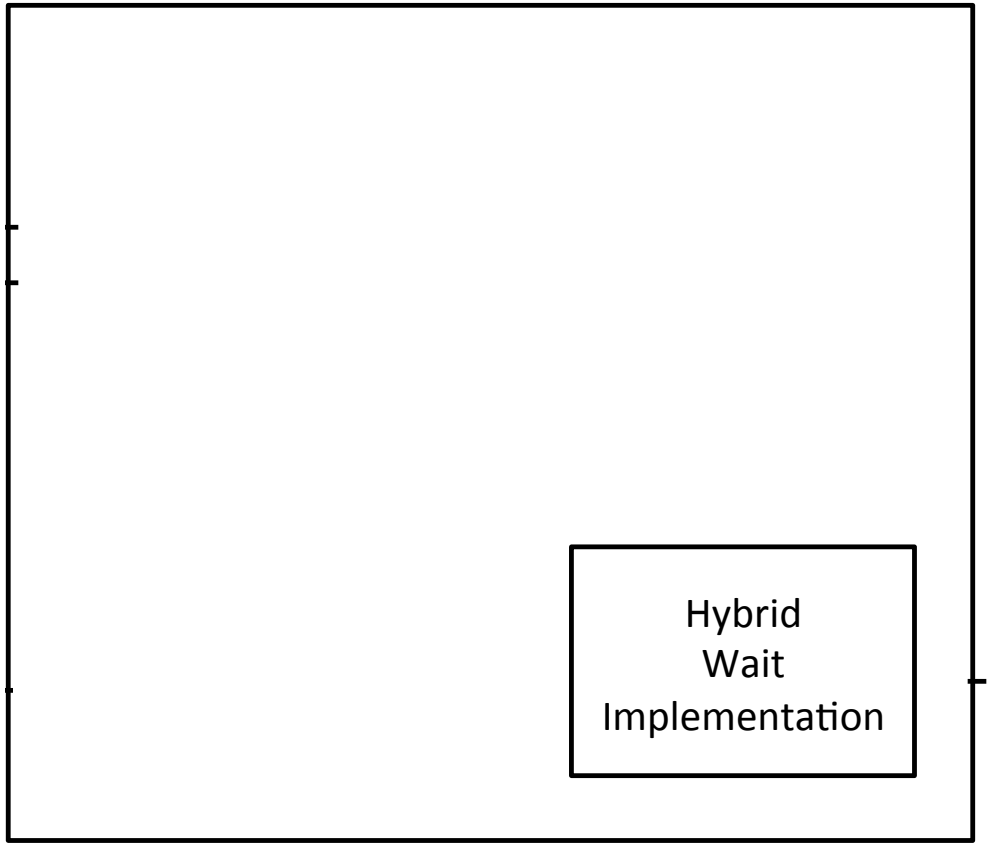
Related work

- Few work in literature
  - Mainly HTM
    - Clock gating processors upon abort
  - Lowering frequency upon abort
    - Using simulator
- Studies
  - HTM consume lower energy
    - Does not fit all workloads
  - Need for adaptability
- Using DVFS in TM
  - Fastlane
    - Designed for low number of threads

Architecture



Architecture



Implementing waits

- Building block for contention managers
- Drastic effect on energy consumption
- Can be implemented in two ways:
  - Busy waiting
  - sleeping

Implementing waits

- Busy waiting
  - Fine granularity
  - Similar to real actual work

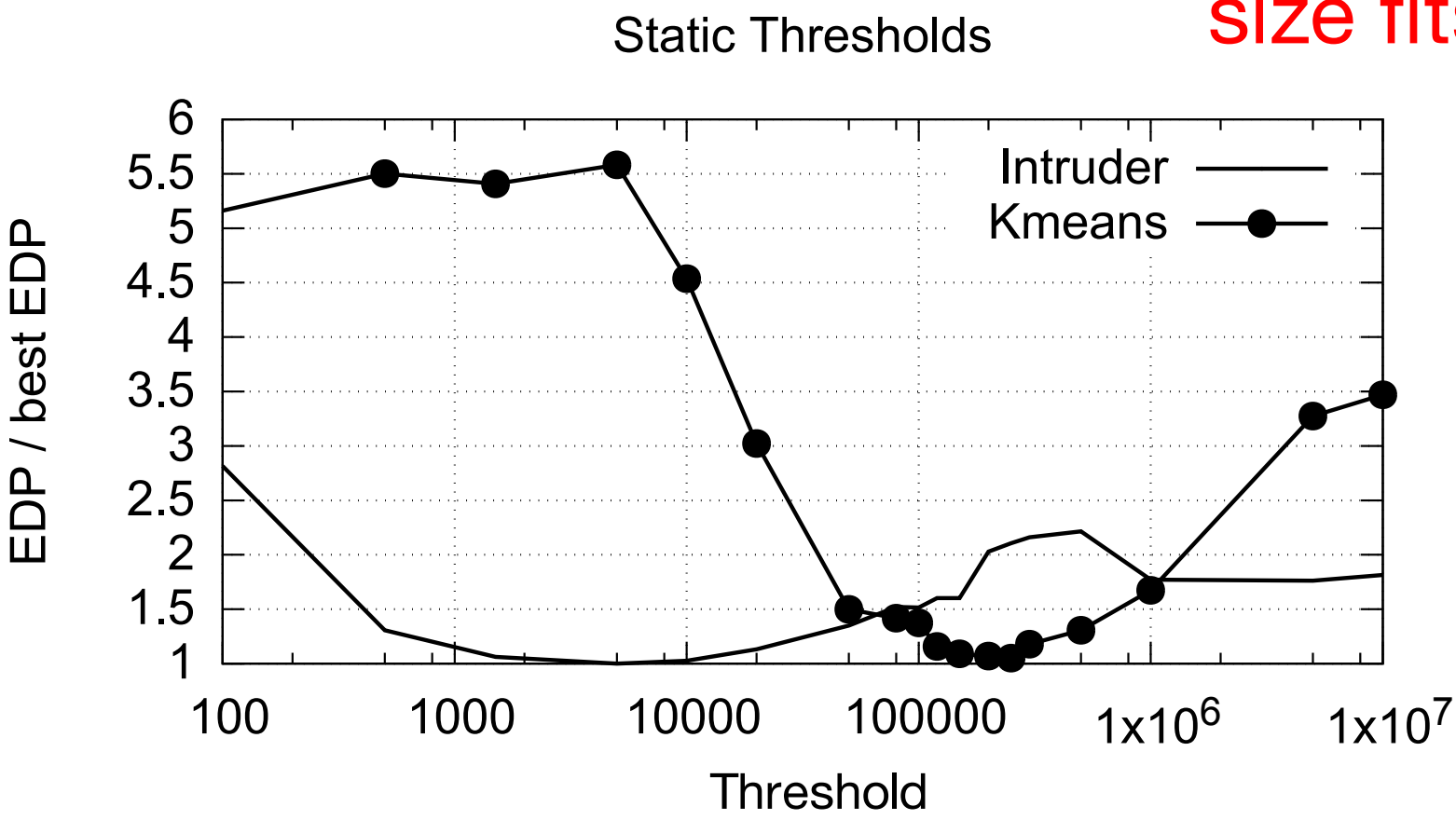
- Sleeping
  - Coarse granularity
  - Low energy consumption
  - expensive

Implementing waits

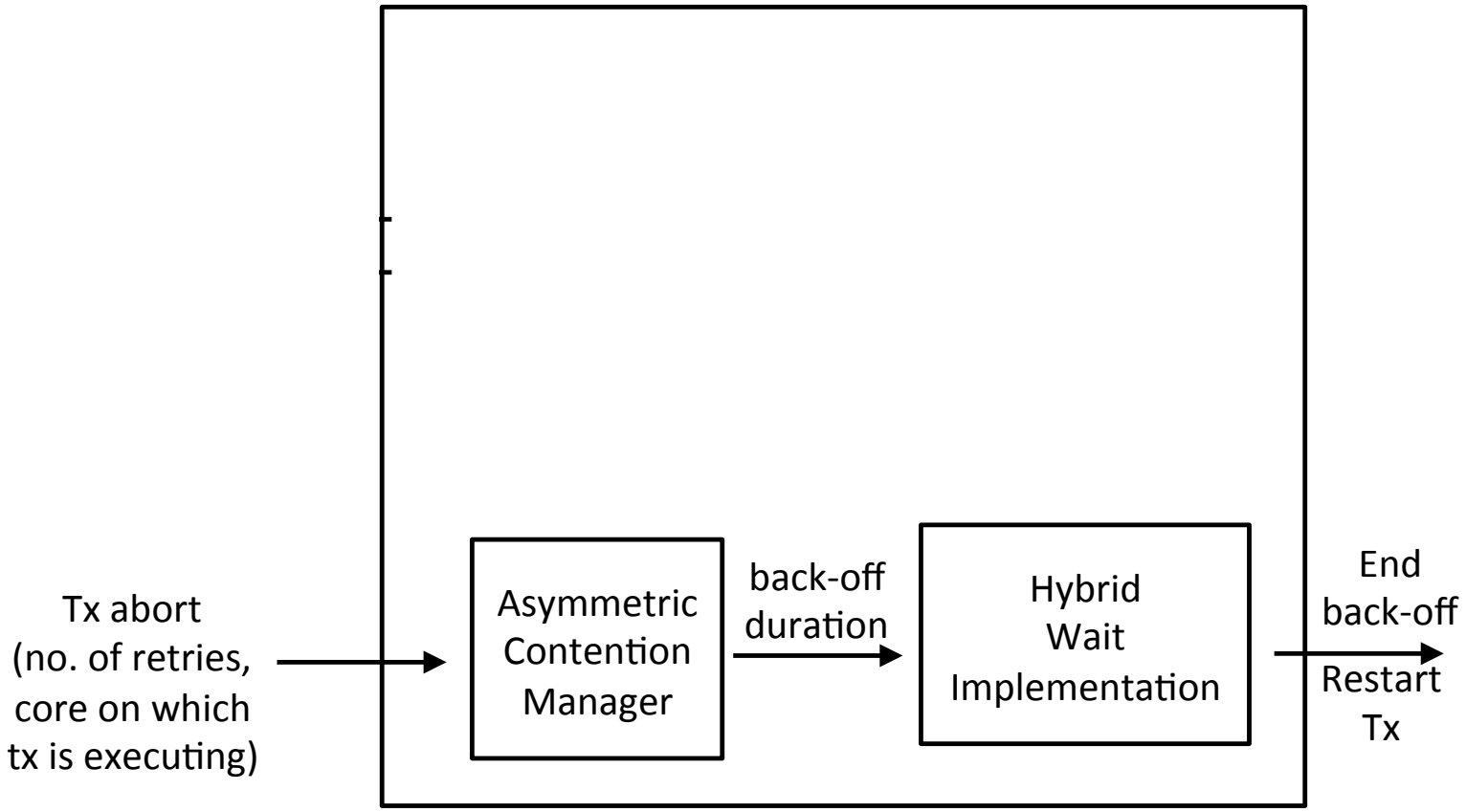
- Hybrid approach
  - Either busy wait or sleep
    - Adaptive fashion
  - How to determine the threshold
    - Cost of sleep

Implementing waits

No one size fits all

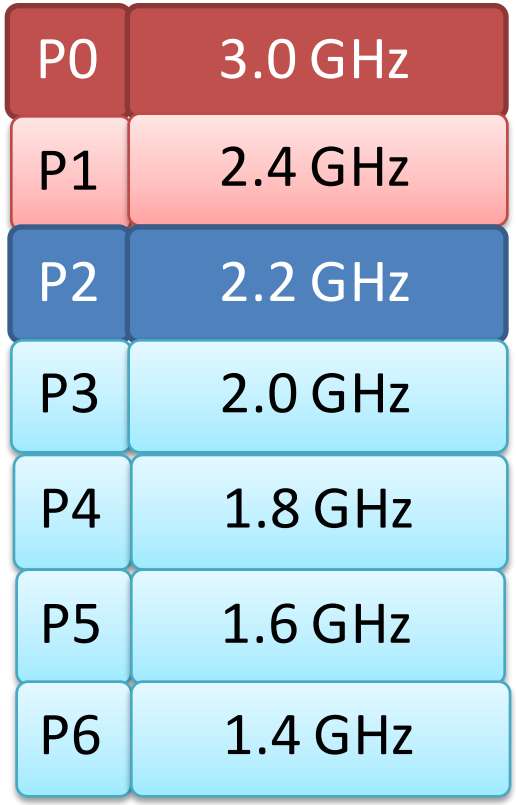


Architecture



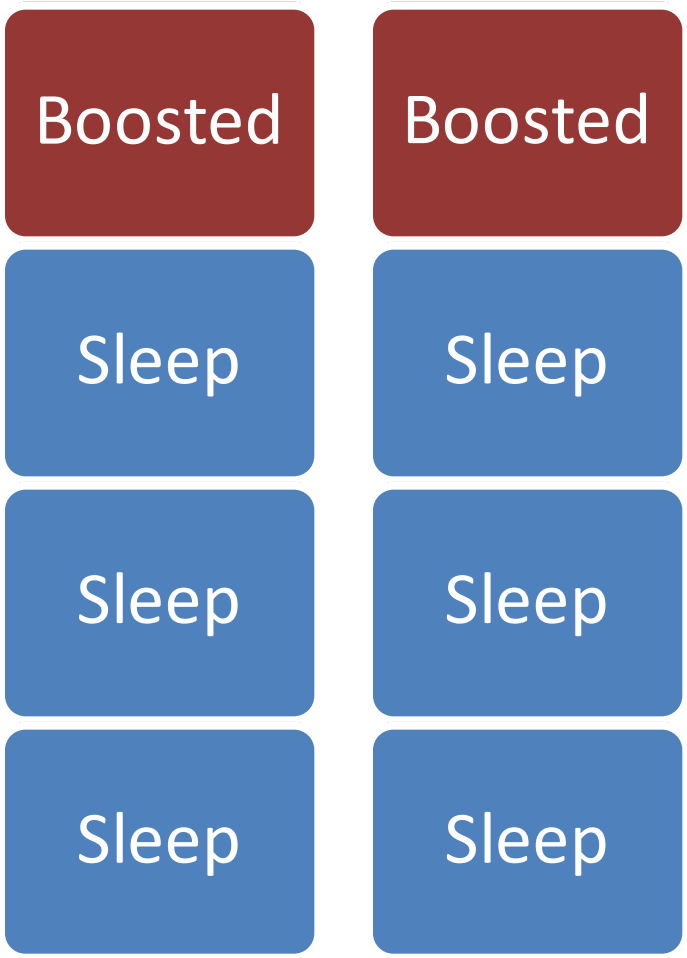
Asymmetric CM

- DVFS
  - Variable operating frequency
- Exploiting DVFS
  - Boosting active threads
  - Reducing freq. of backing off threads
- Enabling DVFS
  - Manual control is expensive
  - How to favor automatic boosting



Asymmetric CM

- Linear backoff cores:
  - Shorter backoff periods
  - Mainly busy waiting backoffs
- Exp. Backoff cores:
  - Longer backoff periods
  - Mainly sleep waiting
- Favor boosting
  - When enough cores are in sleep states



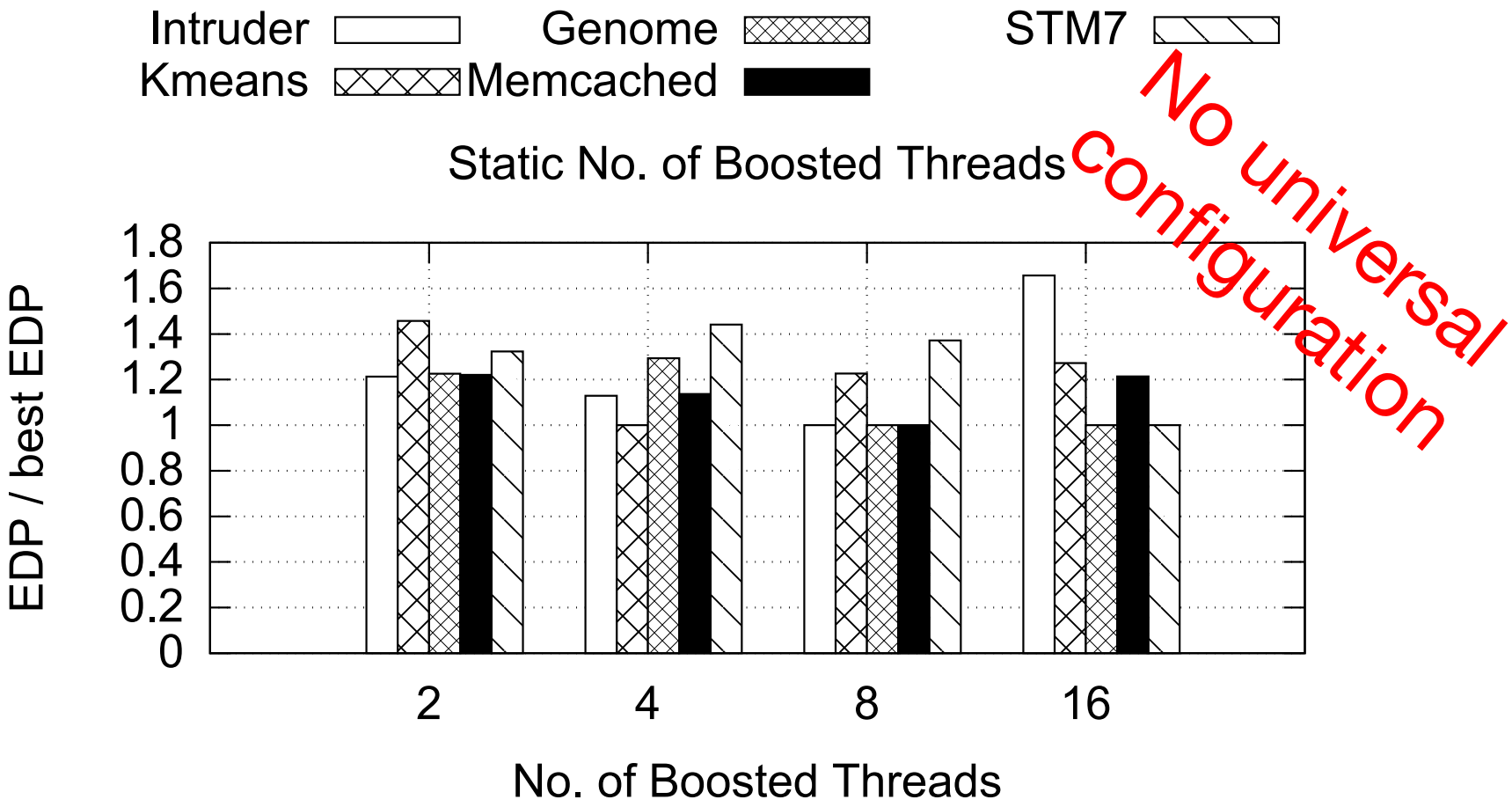
8 core processor

Asymmetric CM

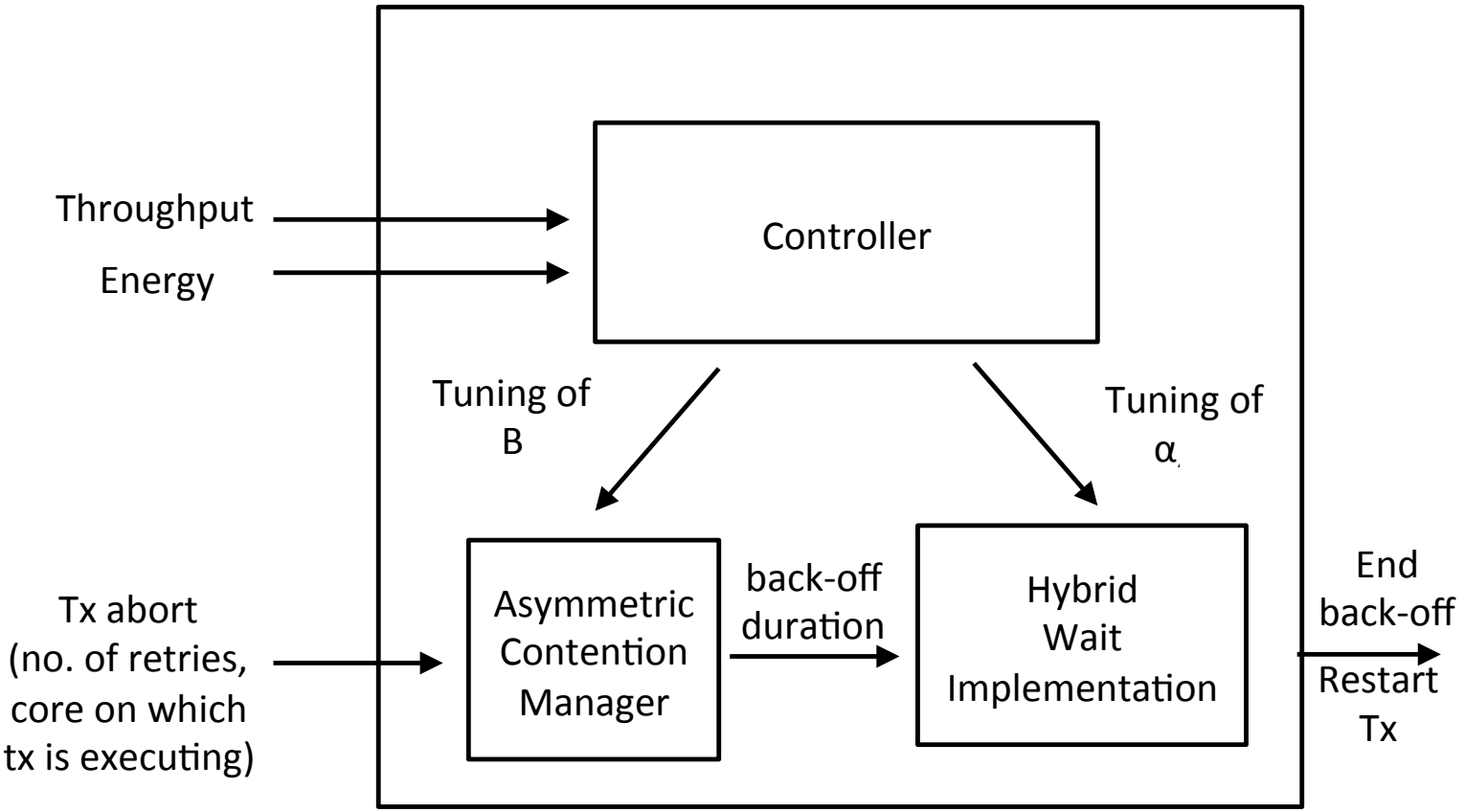
- Increased contention?
  - Cores not backing off exponentially

- Control number of cores to be boosted

Asymmetric CM



Architecture

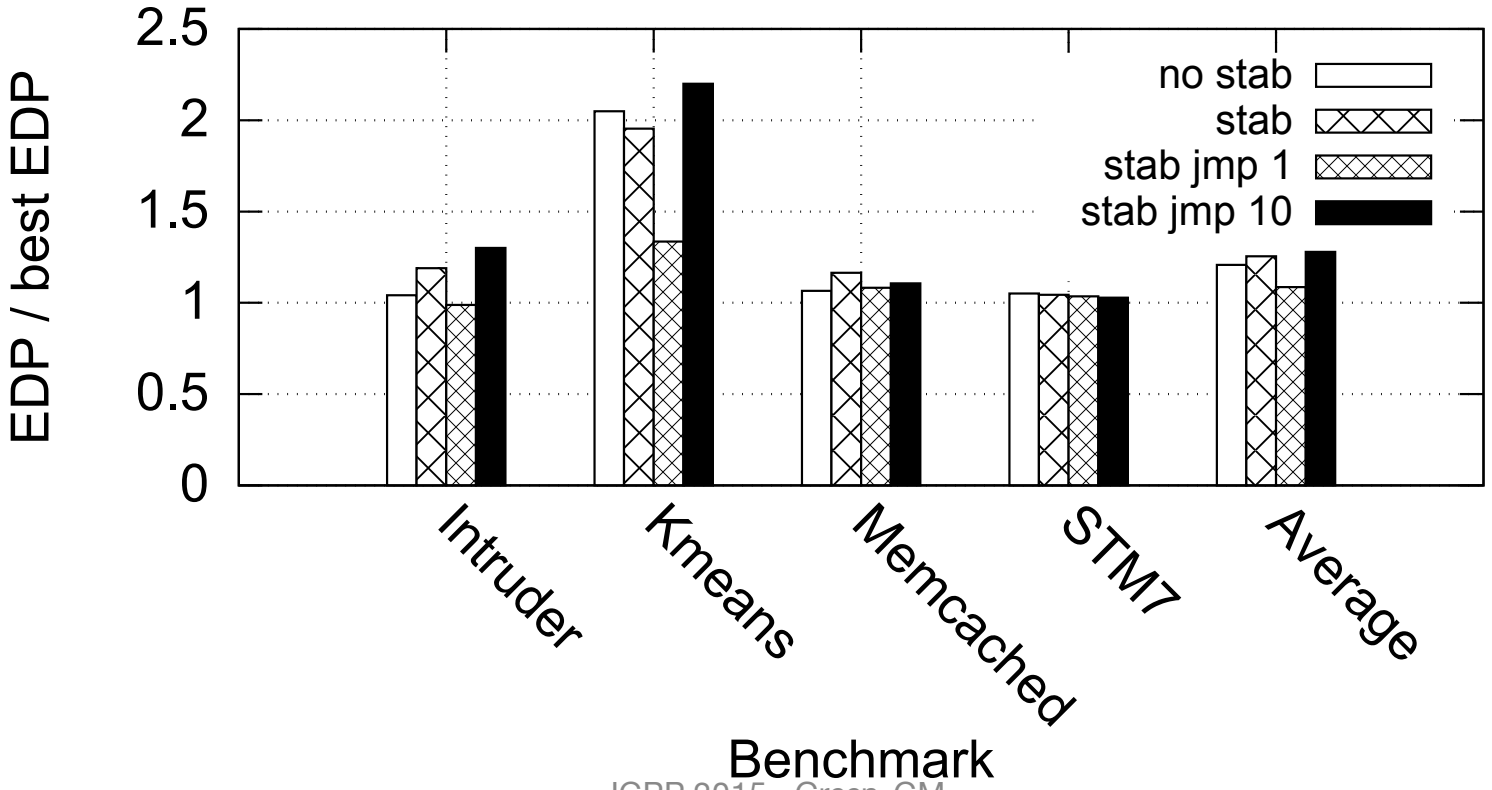


Controller

- Online, lightweight
- Hill climbing
- Challenges:
  - Collection of energy
  - Multi dimensional
- Different exploration strategies
  - Stabilization
  - Random jumps

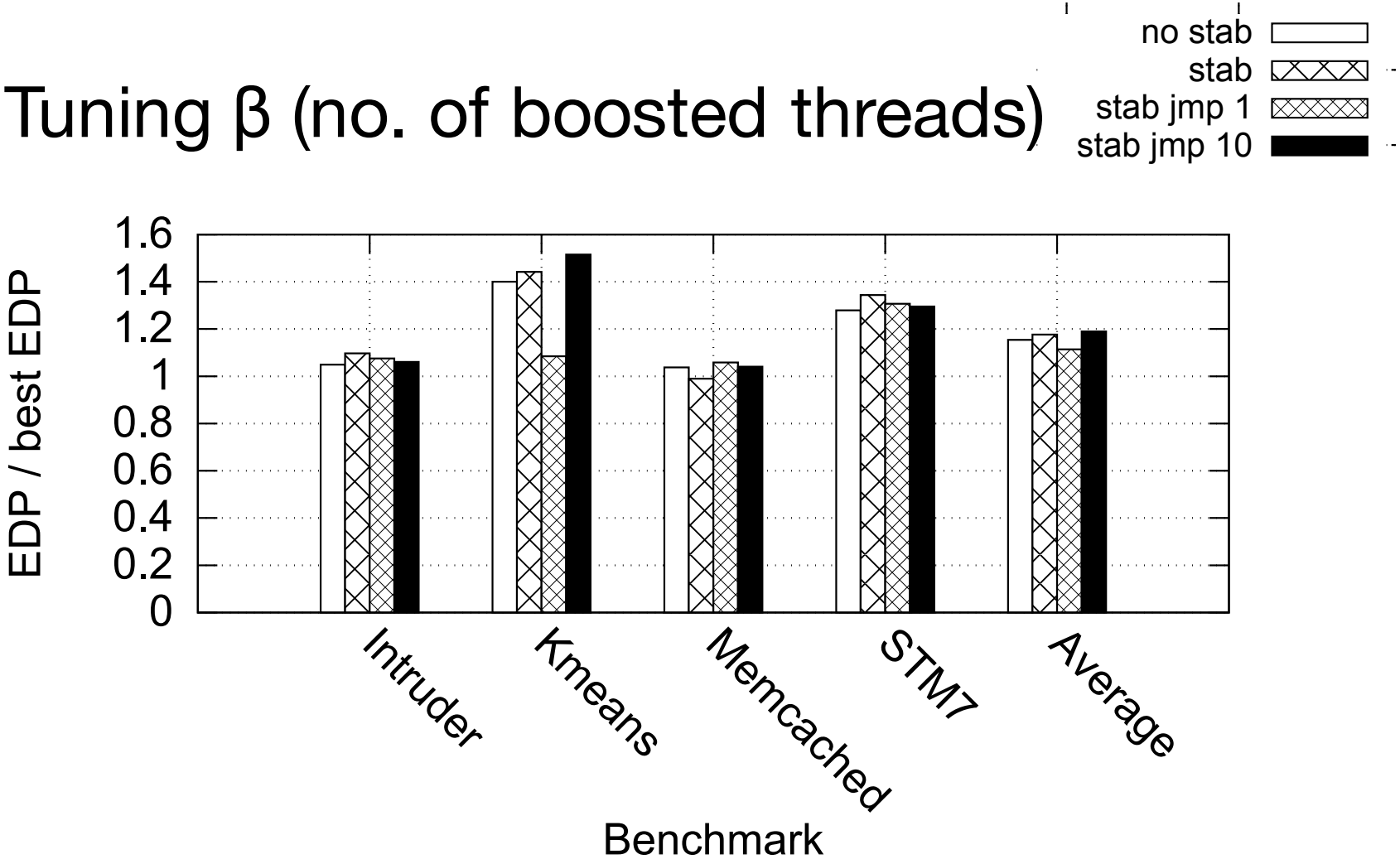
Controller

- Tuning α (threshold for hybrid)



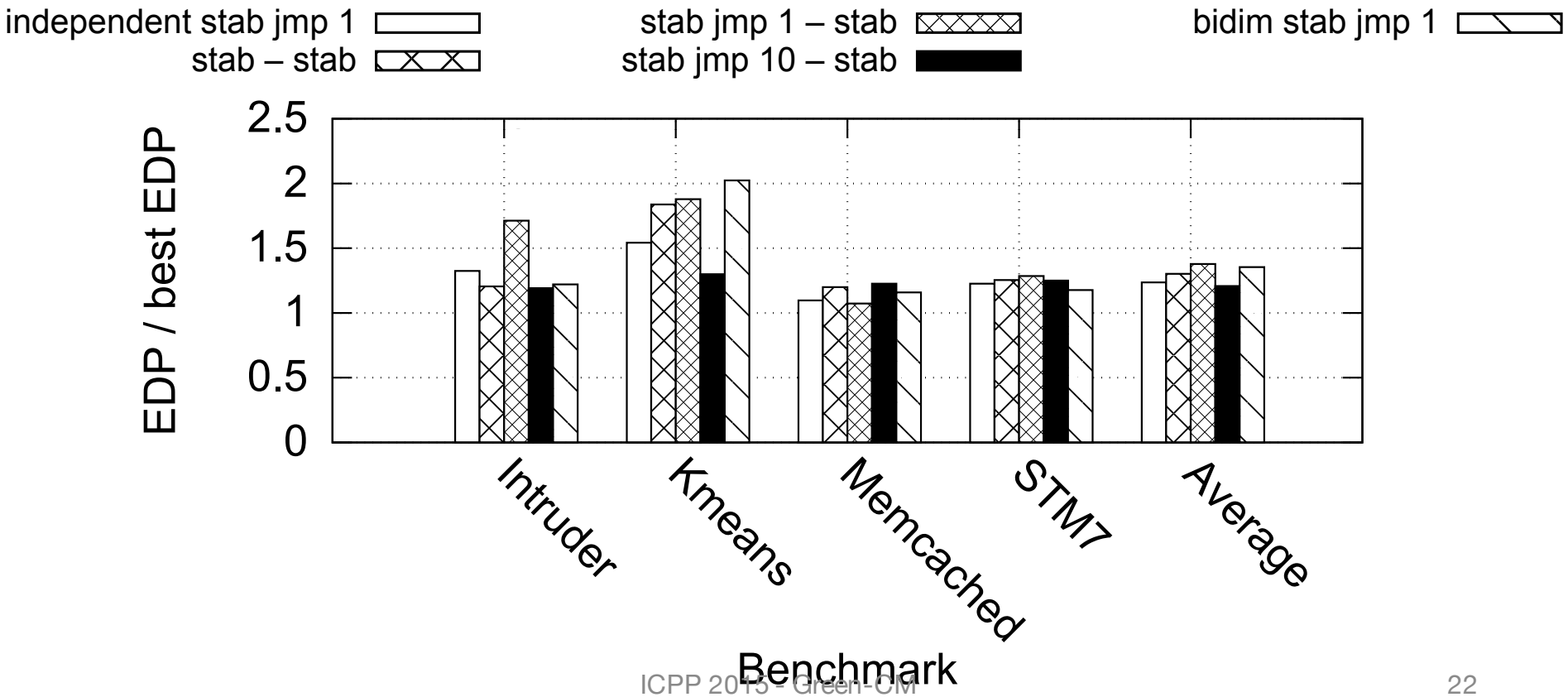
Controller

- Tuning β (no. of boosted threads)

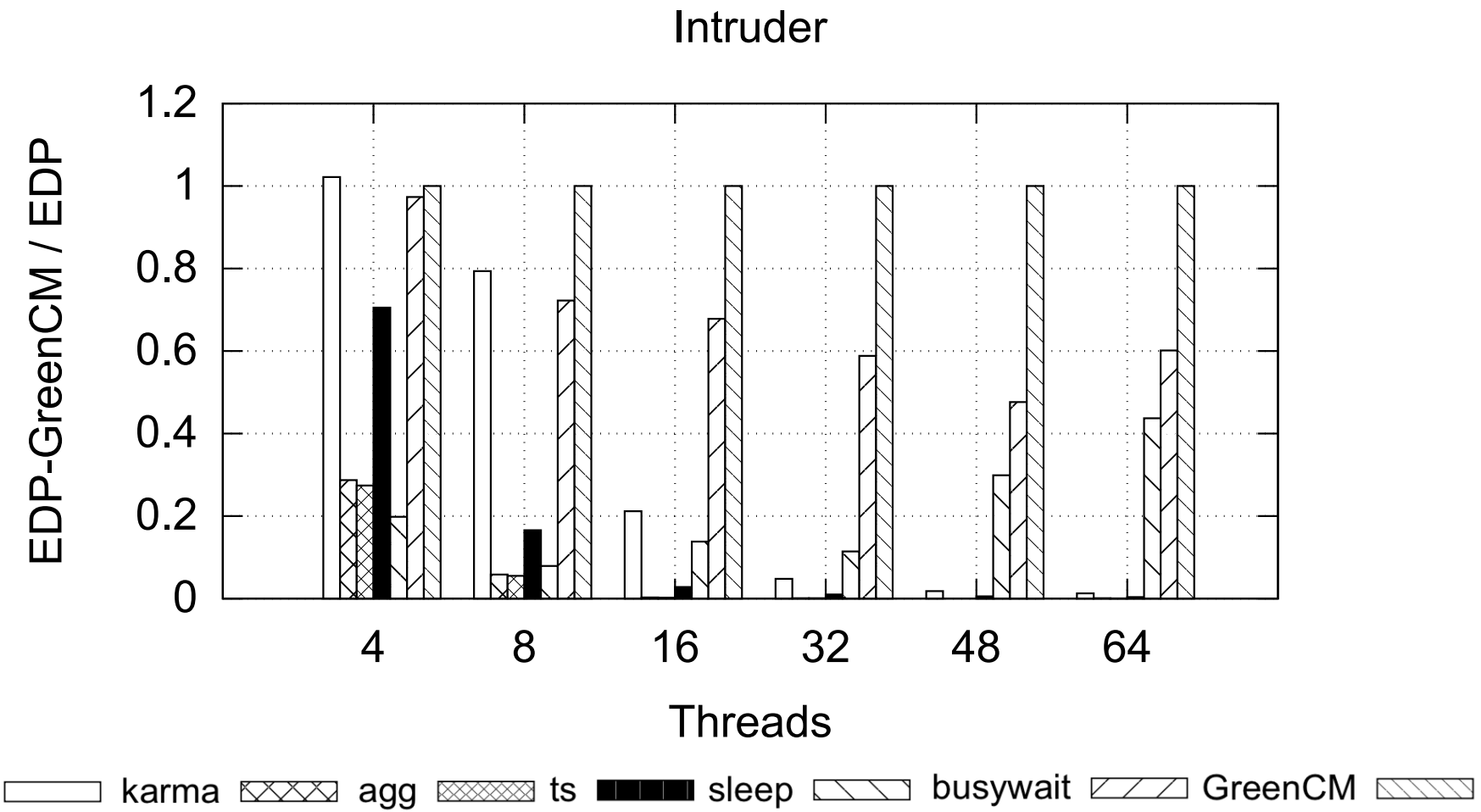


Controller

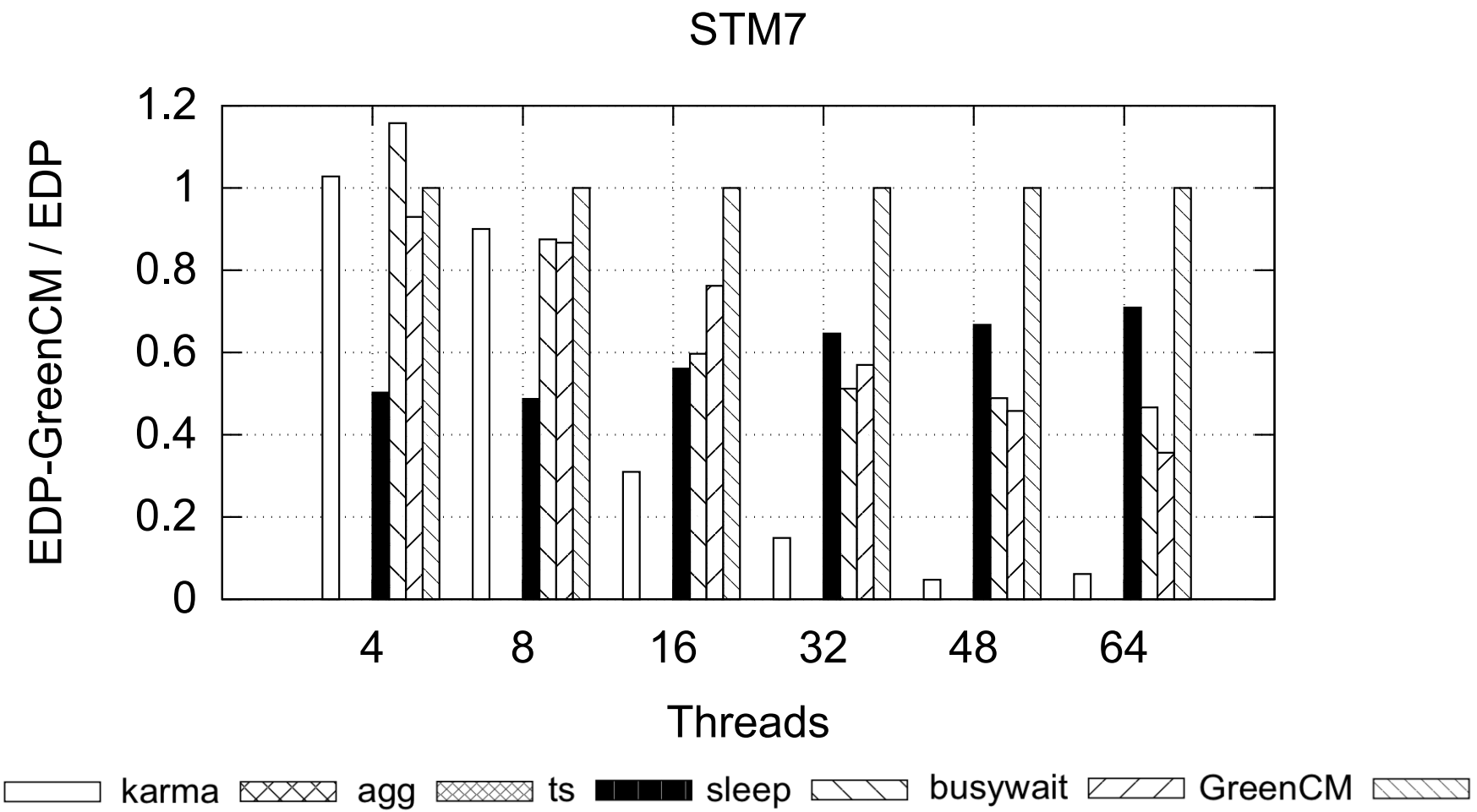
- Merging the learners



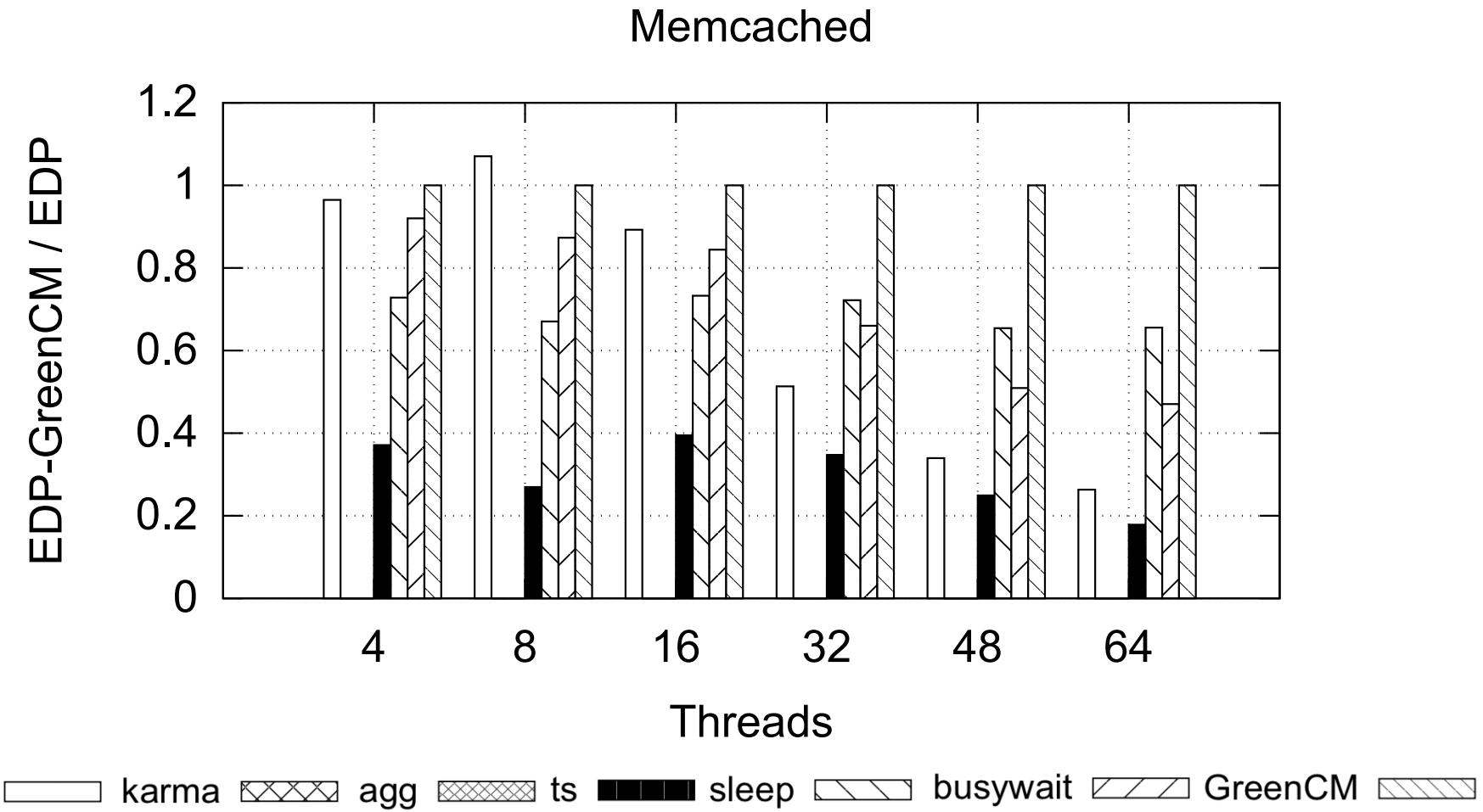
Evaluation


Intruder
EDP-GreenCM / EDP
1.2
1
0.8
0.6
0.4
0.2
0
4
8
16
32
48
64
Threads
suicide
karma
agg
ts
sleep
busywait
GreenCM

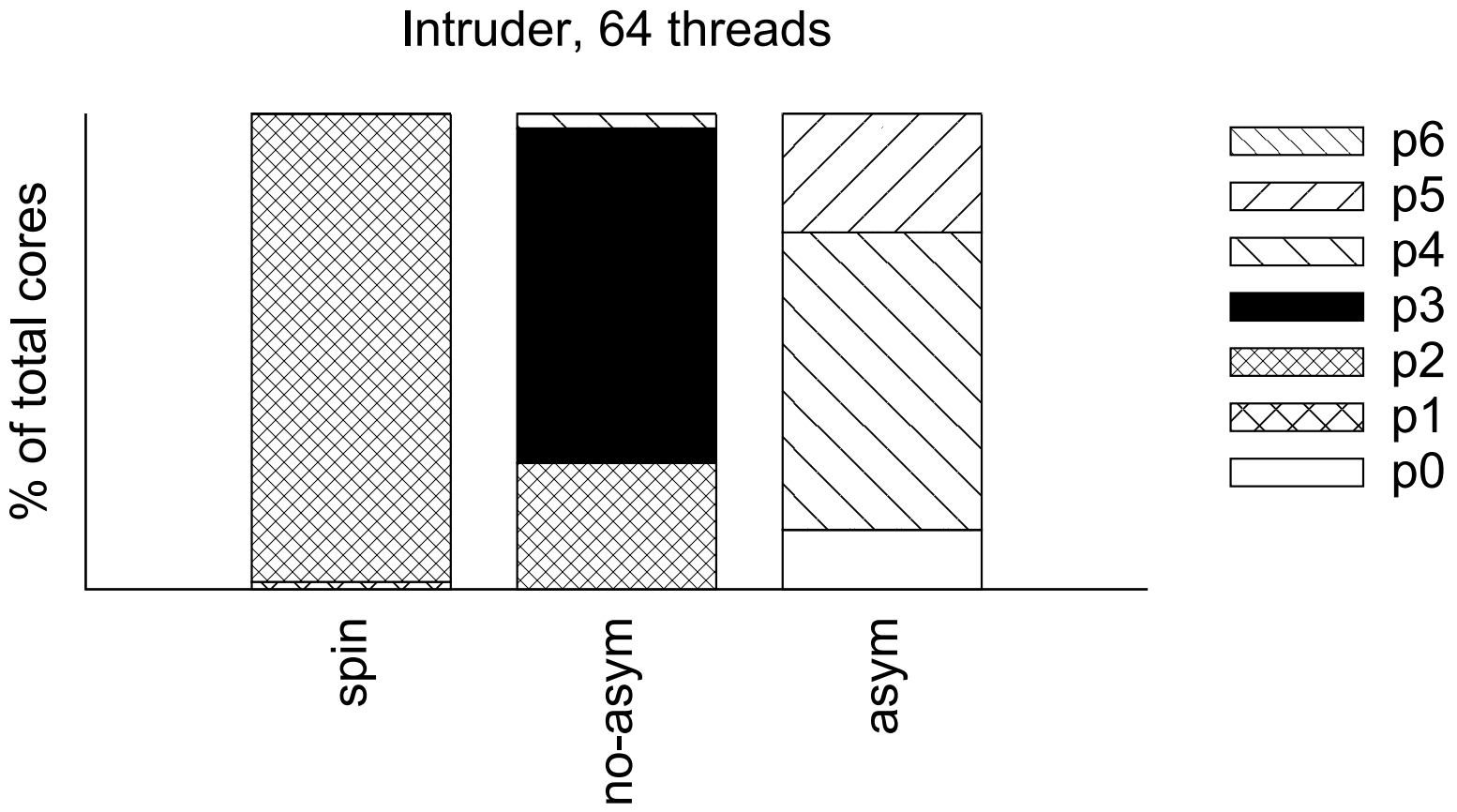
Evaluation


STM7
EDP-GreenCM / EDP
1.2
1
0.8
0.6
0.4
0.2
0
4
8
16
32
48
64
Threads
suicide
karma
agg
ts
sleep
busywait
GreenCM

Evaluation


Memcached
EDP-GreenCM / EDP
1.2
1
0.8
0.6
0.4
0.2
0
4
8
16
32
48
64
Threads
suicide
karma
agg
ts
sleep
busywait
GreenCM

Evaluation

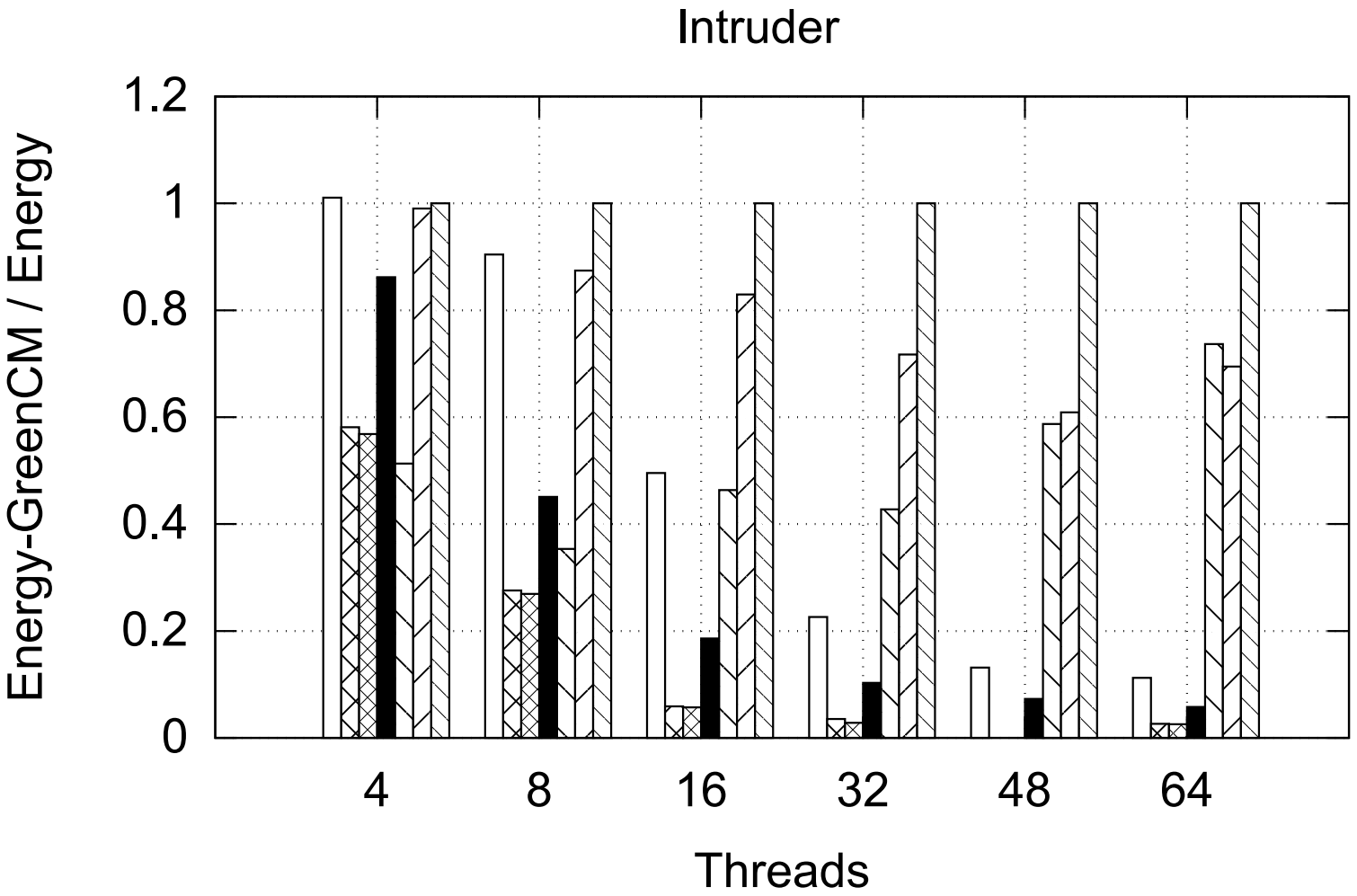

Intruder, 64 threads
% of total cores
spin
no-asym
asym
p6
p5
p4
p3
p2
p1
p0

Conclusion

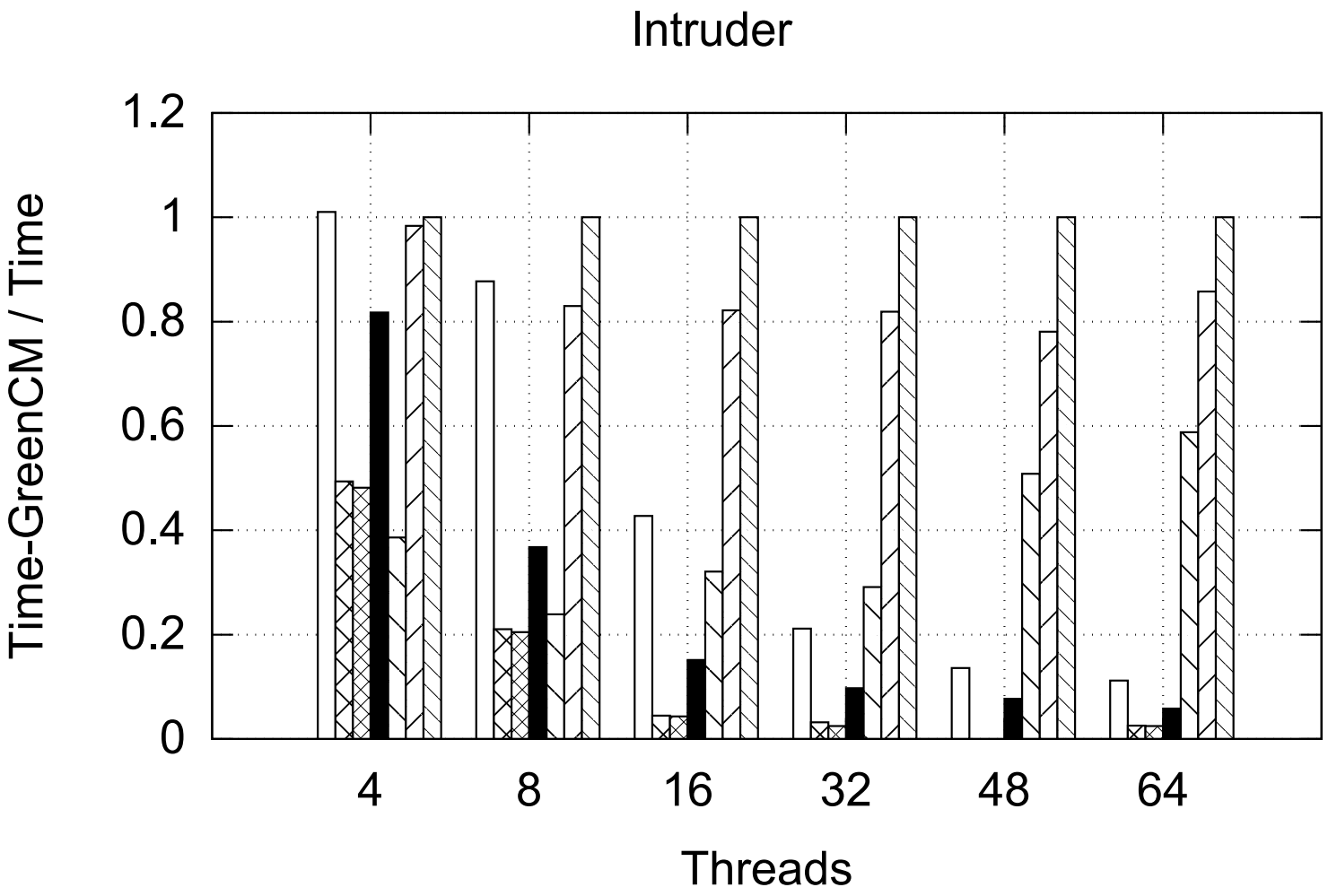
- Implementation of waits has a significant impact on energy efficiency
- Experimental results (obtained on real system) contradict previously published ones based on simulation
- Exploiting DVFS enhances energy efficiency
- Self-tuning is needed to adapt to different workloads

THANK YOU

Evaluation


Intruder
Energy-GreenCM / Energy
1.2
1
0.8
0.6
0.4
0.2
0
4
8
16
32
48
64
Threads

Evaluation


Intruder
Time-GreenCM / Time
1.2
1
0.8
0.6
0.4
0.2
0
4
8
16
32
48
64
Threads