Minor Thesis

Development and Analysis of Barrier Protocols

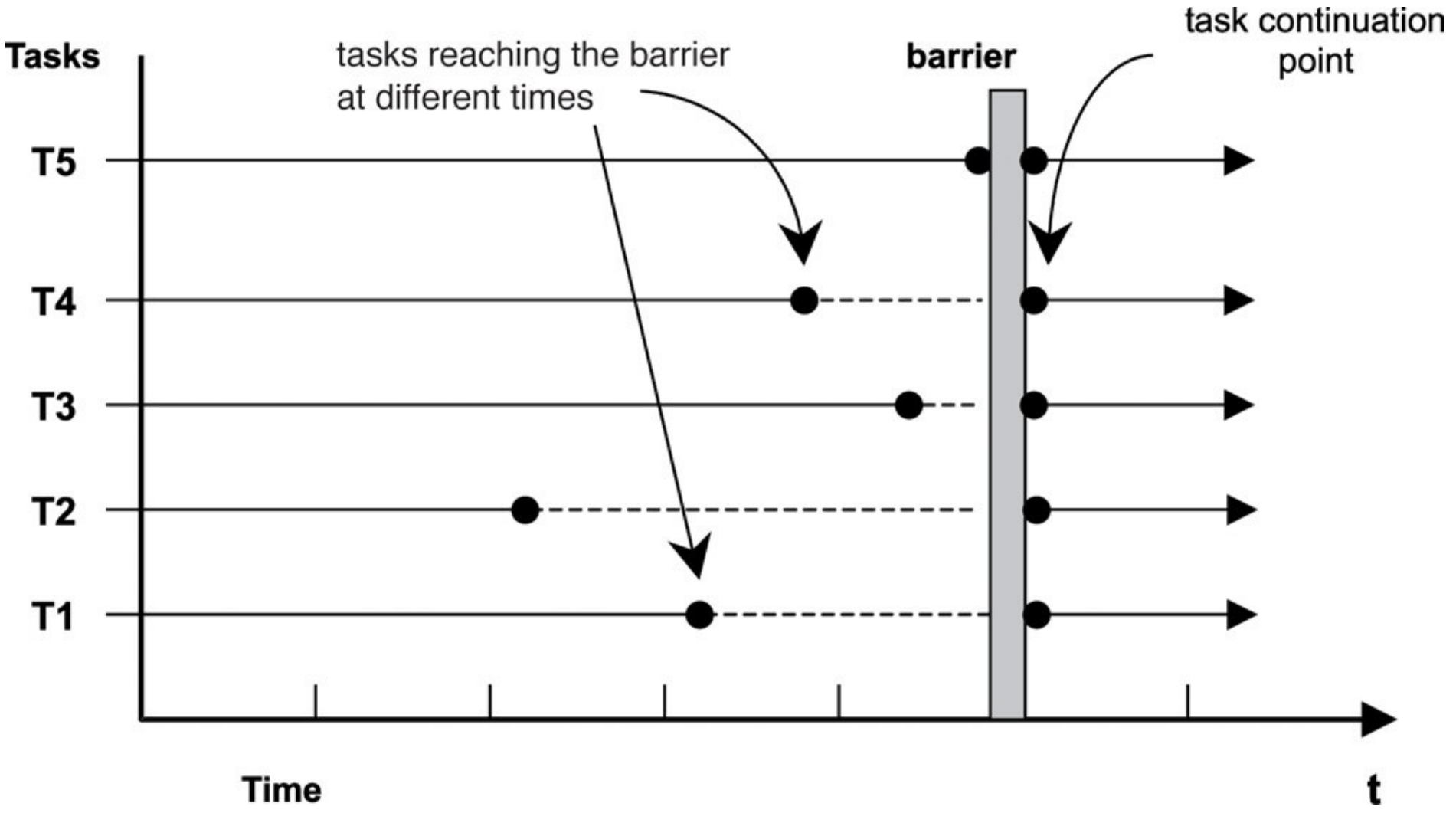
Ronny Brendel (http://automaton2000.com)
Responsible Professor: Prof. Dr. Christel Baier
Supervisor: Dr. Sascha Klüppelholz

Dresden, 2014-01-08

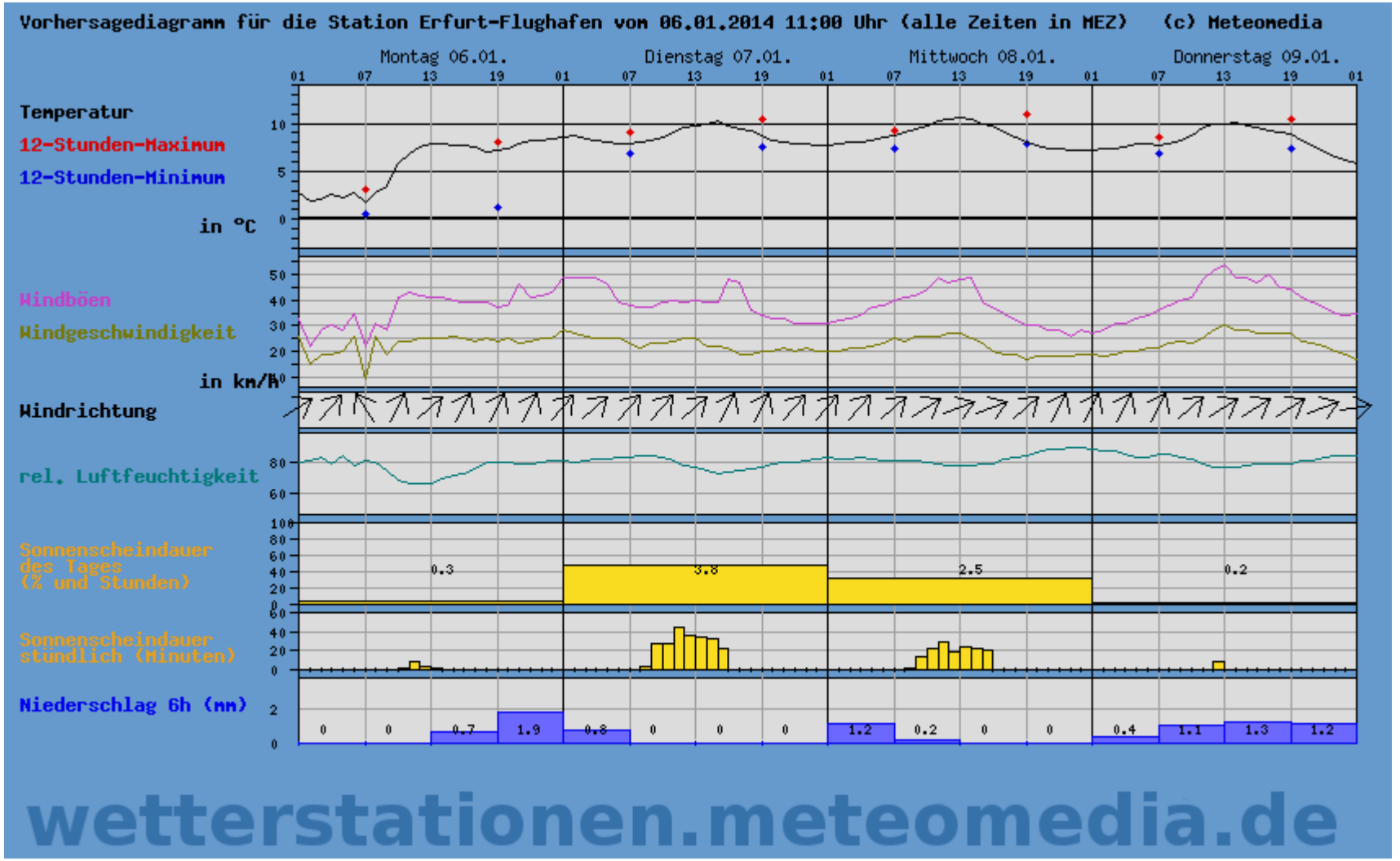
Content

- Introduction
  - Basics, Motivation
- Protocols
  - Central Counter, B1 Barrier
- Modelling
  - Shared Variable, Protocols
- Analysis
  - Functional, Quantitative
- Conclusion, Future Work, Sources

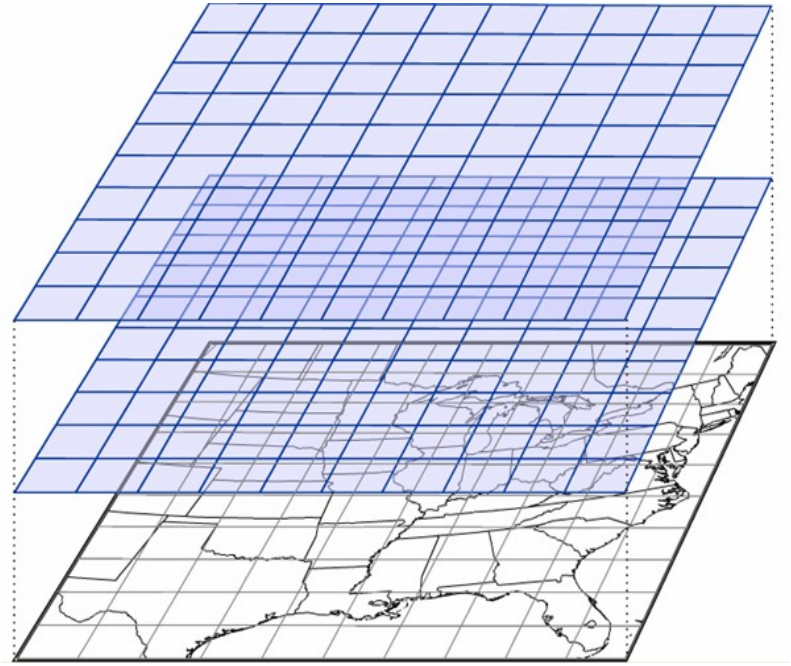
Basics ◁ Introduction

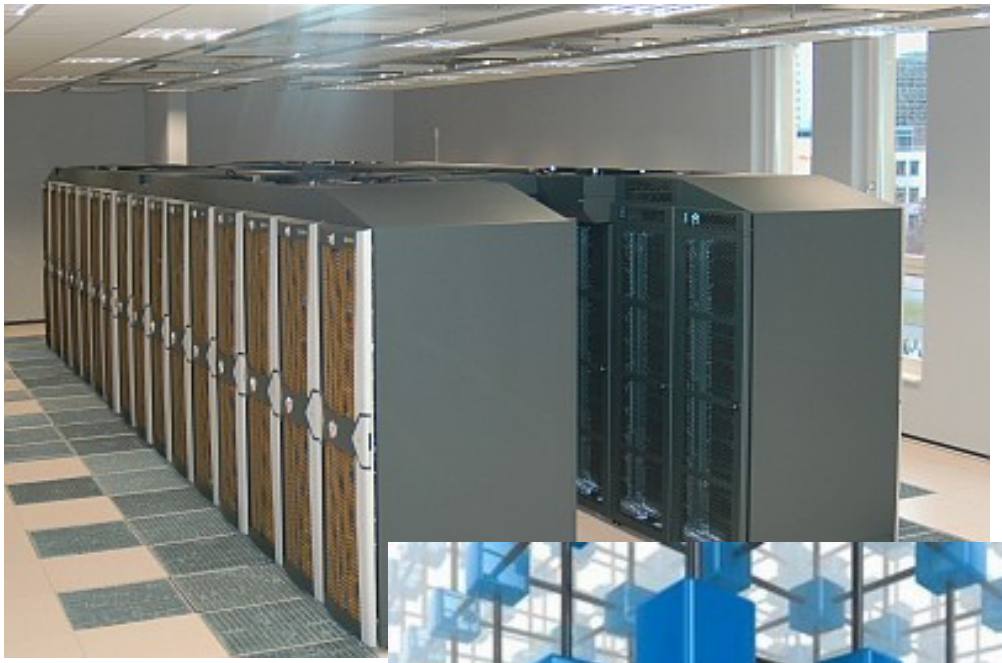


Basics ◁ Introduction

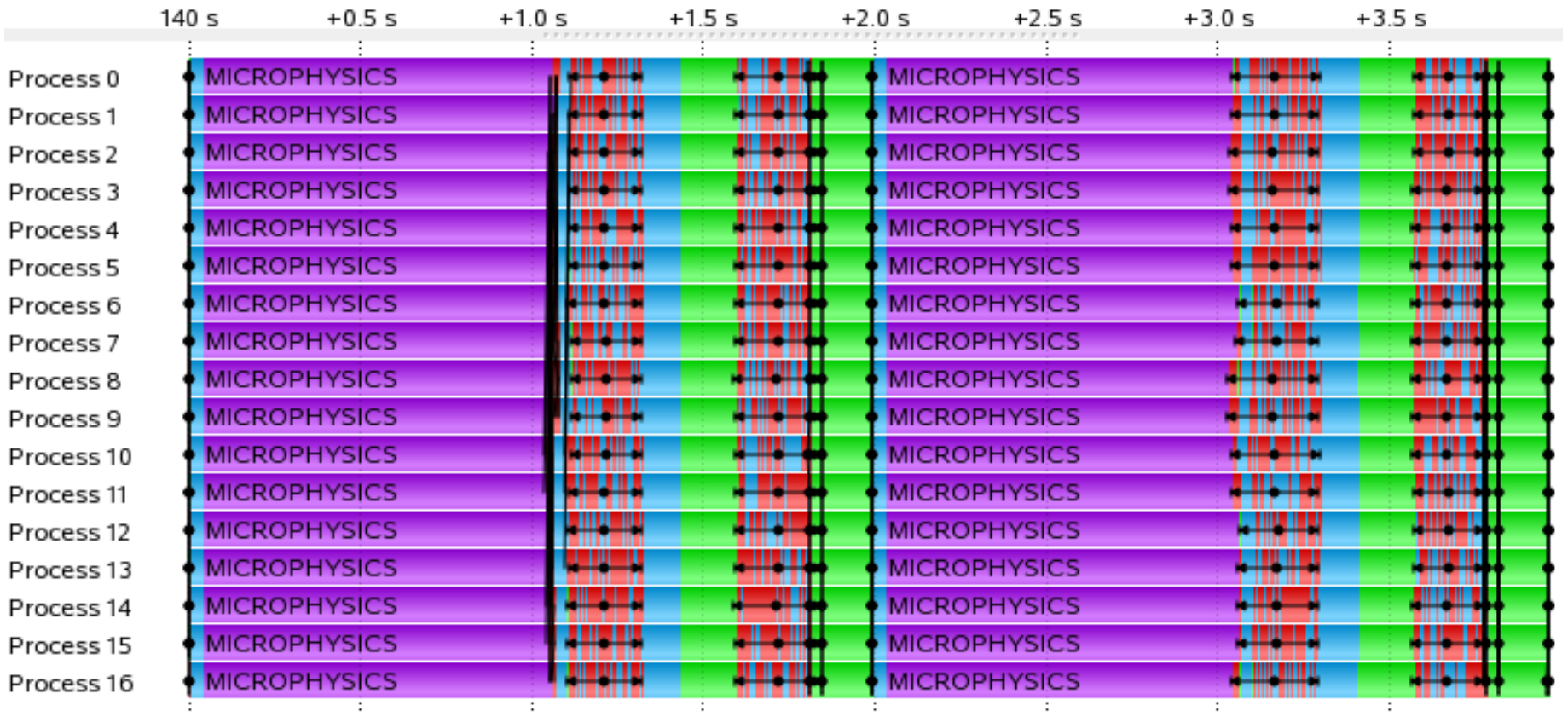


Basics ◁ Introduction

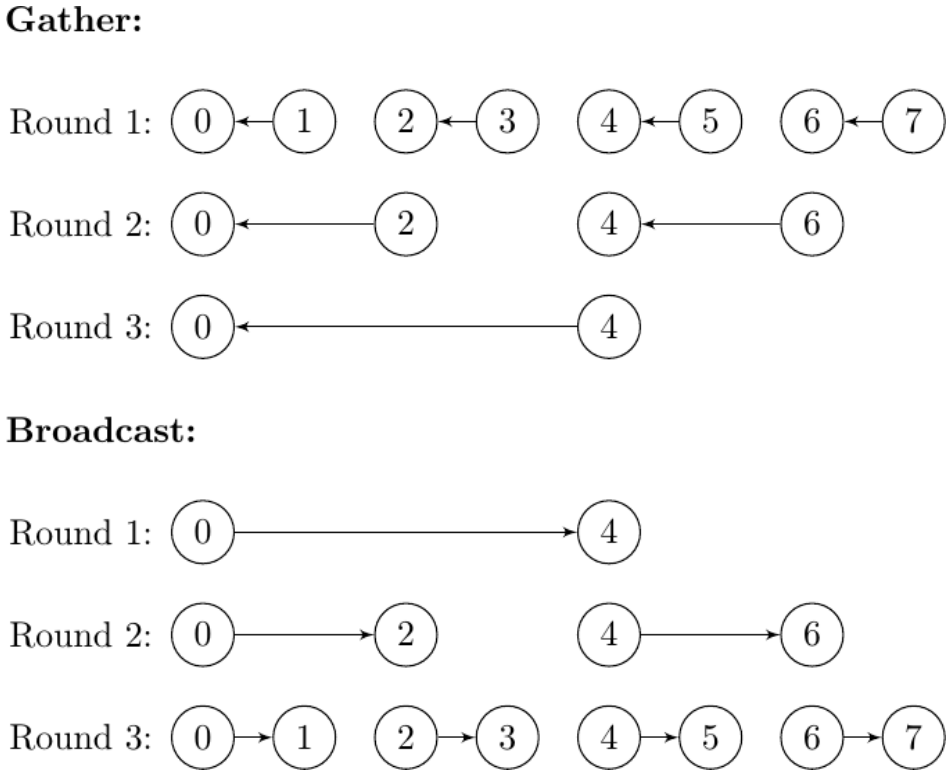




Basics ◁ Introduction



- Usual Implementations include
  - Central Counter Barrier (atomic increment)
  - Hierarchical approaches



see sources [3]

Motivation ◁ Introduction

- Today's Barrier Protocols have been invented long ago
- Probabilistic Write/Copy-Select (pW/CS)
  - Concurrent protocols are unnecessarily strict
  - Relieving strictness can improve performance
  - Complexity of modern computers makes the timing of concurrent interaction effectively random. Employ the tools of probability theory for designing/analysing protocols

see sources [1,2]

- Tests/Benchmarks
  - not exhaustive
  - not arbitrarily fine-grained, incurs overhead
  - testing probabilistic algorithms is hard
- Model checking
  - exhaustive
  - arbitrarily fine-grained, no overhead

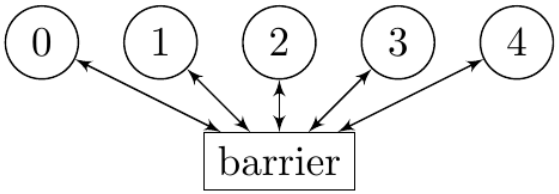
- Improve barrier protocols
- Using the principles behind pW/CS lock
- Analysis through model checking

Central Counter Bar. ◁ Protocols

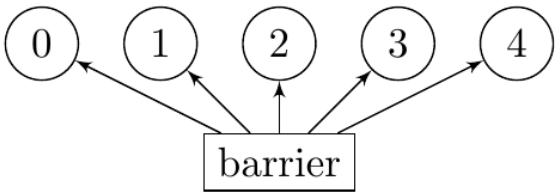
```
shared variables: integer barrier := threadCount
---
atomic{barrier := barrier - 1}

wait until barrier = 0
```

Atomic decrement:



Repeated reading:



B1 Barrier ◁ Protocols

```
shared variables: boolean barrier[threadCount]
local variables:  integer i
initialisation:   barrier[*] := false
```

```
barrier[threadIndex] := true

i := 0
while i < threadCount {
    if barrier[i] = false {
        i := -1
    }
    i = i + 1
}
```

Modelling

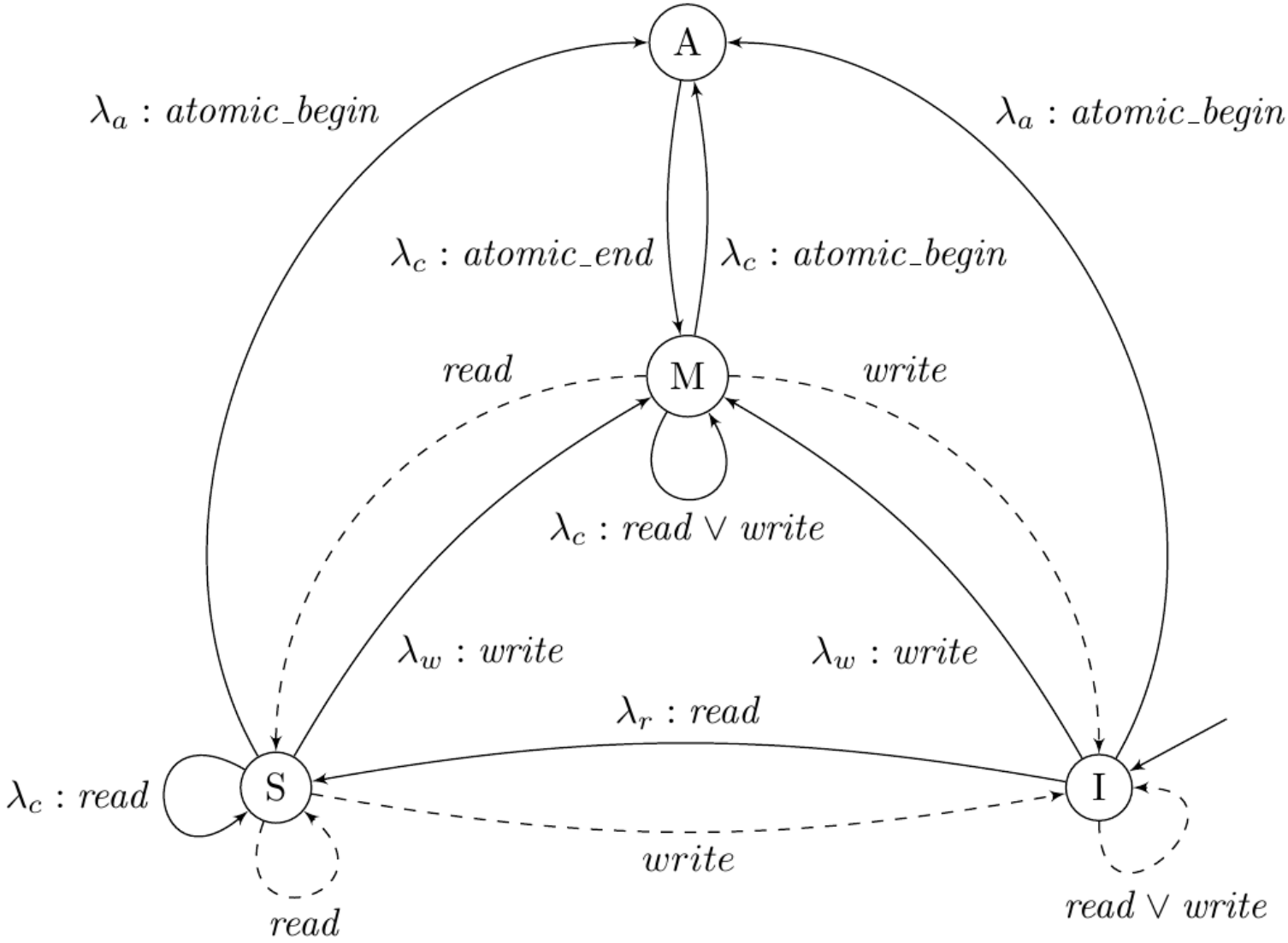
- Functional
  - Non-deterministic transition system + LTL
  - SPIN
  - detailed model to reveal all possible mistakes
- Quantitative
  - CTMC + CSL/CSRL
  - PRISM
  - reduced to just costly/important transitions, no reinitialisation

see sources [4,5]

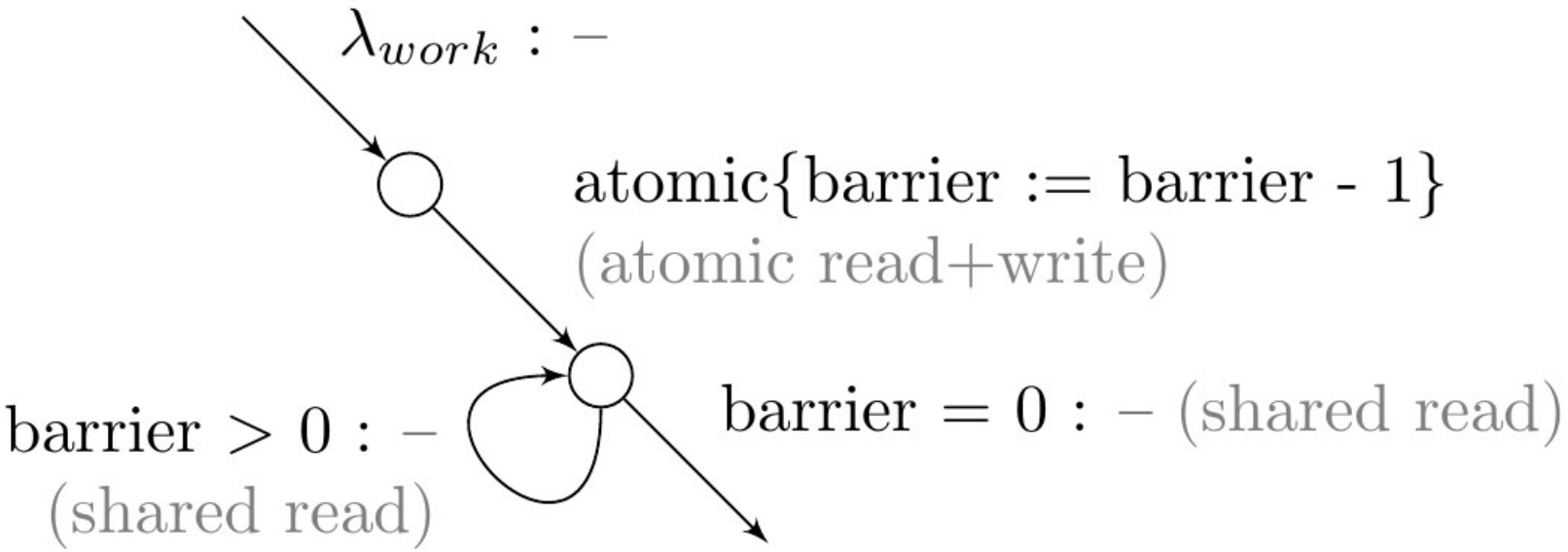
Shared Variable ◁ Modelling

- Synchronisation is about exchanging information, i.e. sharing memory
- Very small information → Timing dominated by memory access latency
- Memory access is cached → We have to model caching
- We identify a shared variable with the cache line it resides on
- MSI protocol + atomic operations

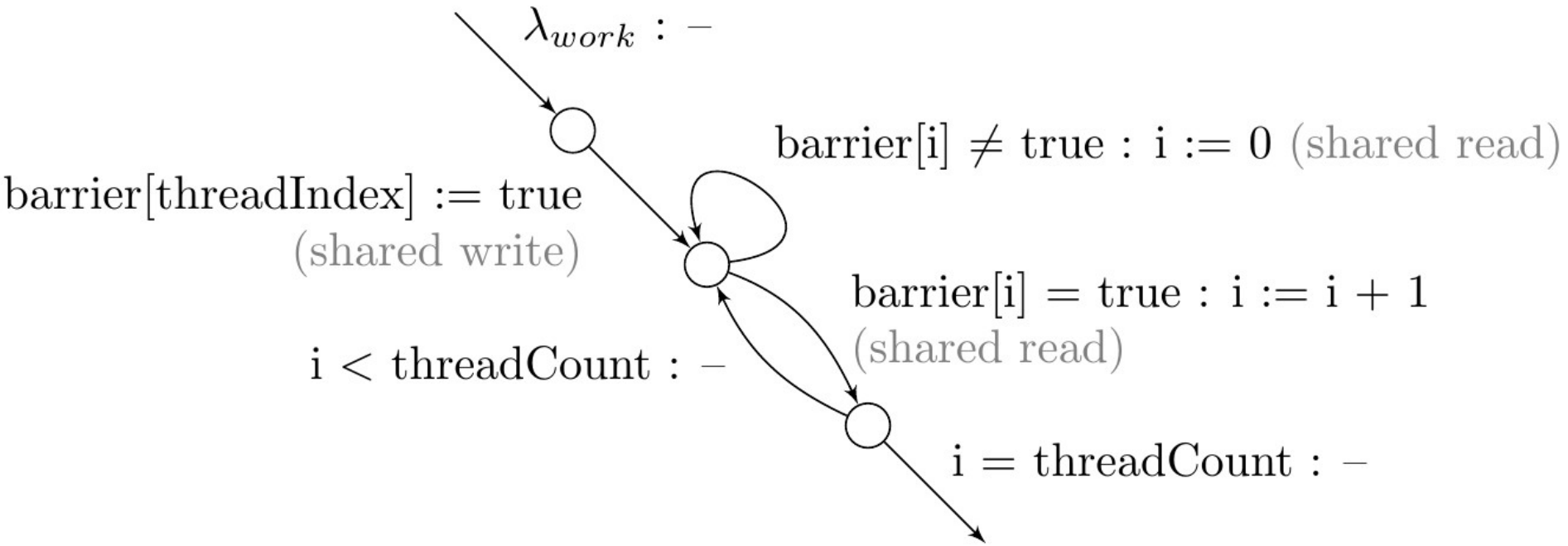
Shared Variable ◁ Modelling



Central Counter Bar. ◁ Modelling



B1 Barrier ◁ Modelling



Functional $\triangleleft$ Analysis

- “A thread may only exit the barrier if all threads have entered it”

$$\Box(one\_left \implies all\_entered)$$

- “If all threads entered the barrier, each one leaves it in a finite amount of time”

$$\Box(all\_entered \implies \Diamond all\_left)$$

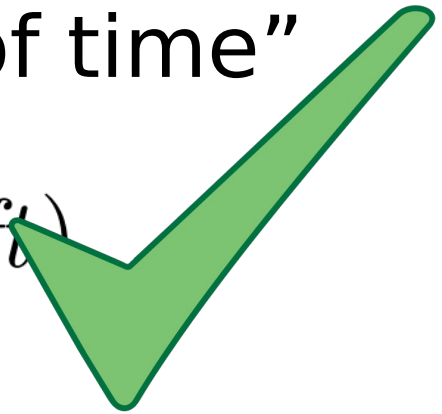
Functional ◁ Analysis

- “A thread may only exit the barrier if all threads have entered it”

$$\Box(one\_left \implies all\_entered)$$

- “If all threads entered the barrier, each one leaves it in a finite amount of time”

$$\Box(all\_entered \implies \Diamond all\_left)$$



Quantitative ◁ Analysis

- Interesting values
  - time in cycles (2.5GHz clock speed)
  - energy consumption in joule / watt
  - implemented using CTMC time and rewards
- Parameters to variate
  - Number of threads
  - Rate of the initial (work) distribution

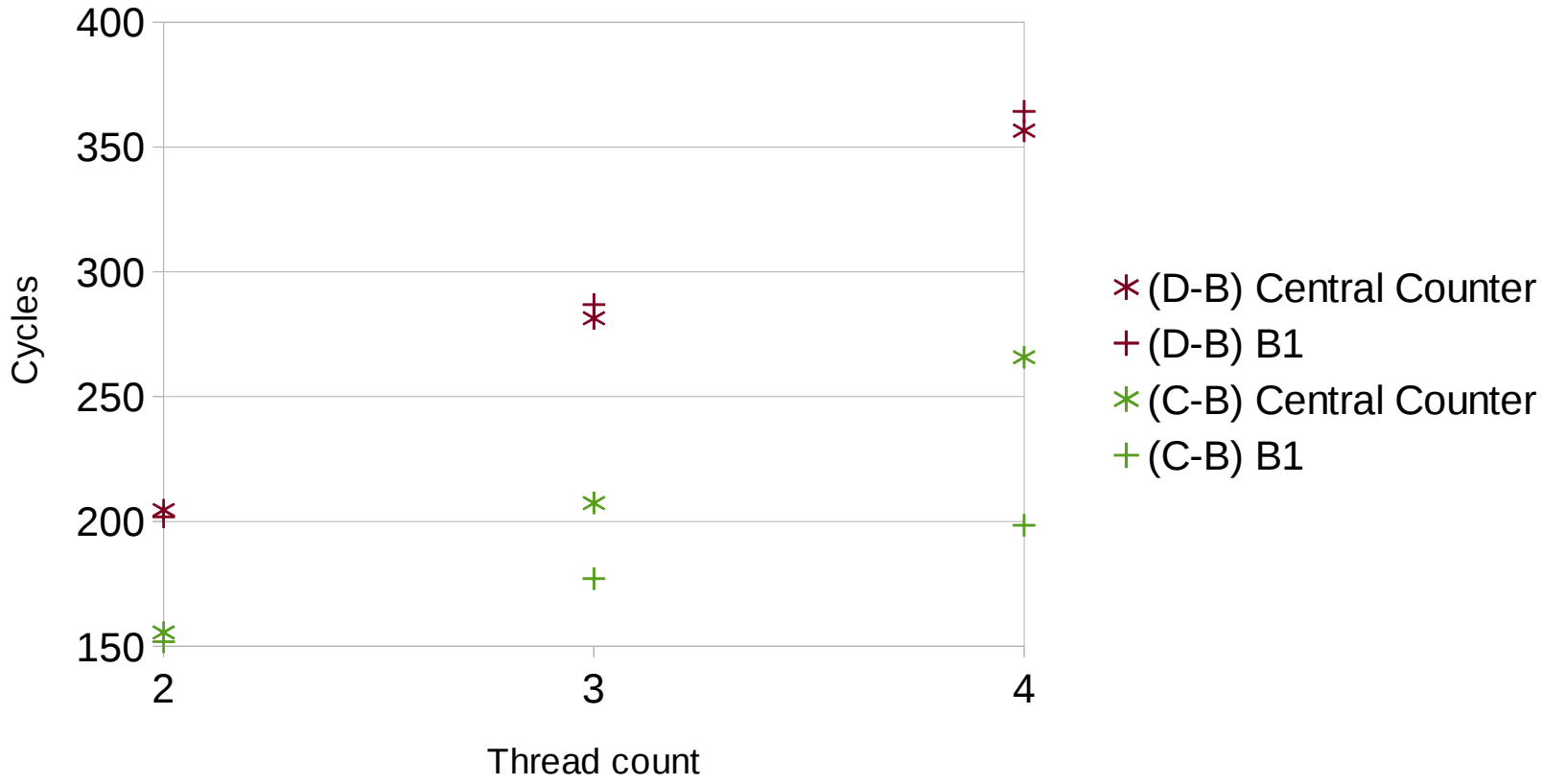
Quantitative ◁ Analysis

- Points in time when to measure
  - `(A)` First thread entered the barrier
  - `(B)` Last thread entered
  - `(C)` First thread left
  - `(D)` Last thread left
  - `(W)` Last thread finished writing

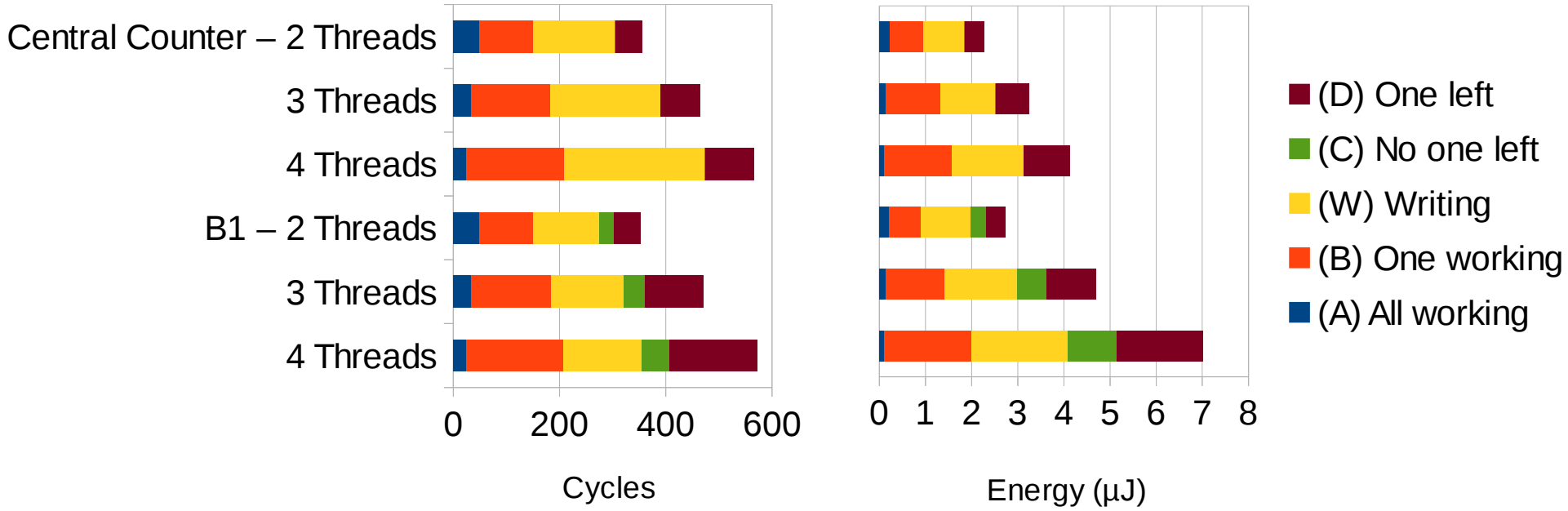
- We take the reachability reward wrt the proper reward functions at these points in time

Quantitative ◁ Analysis

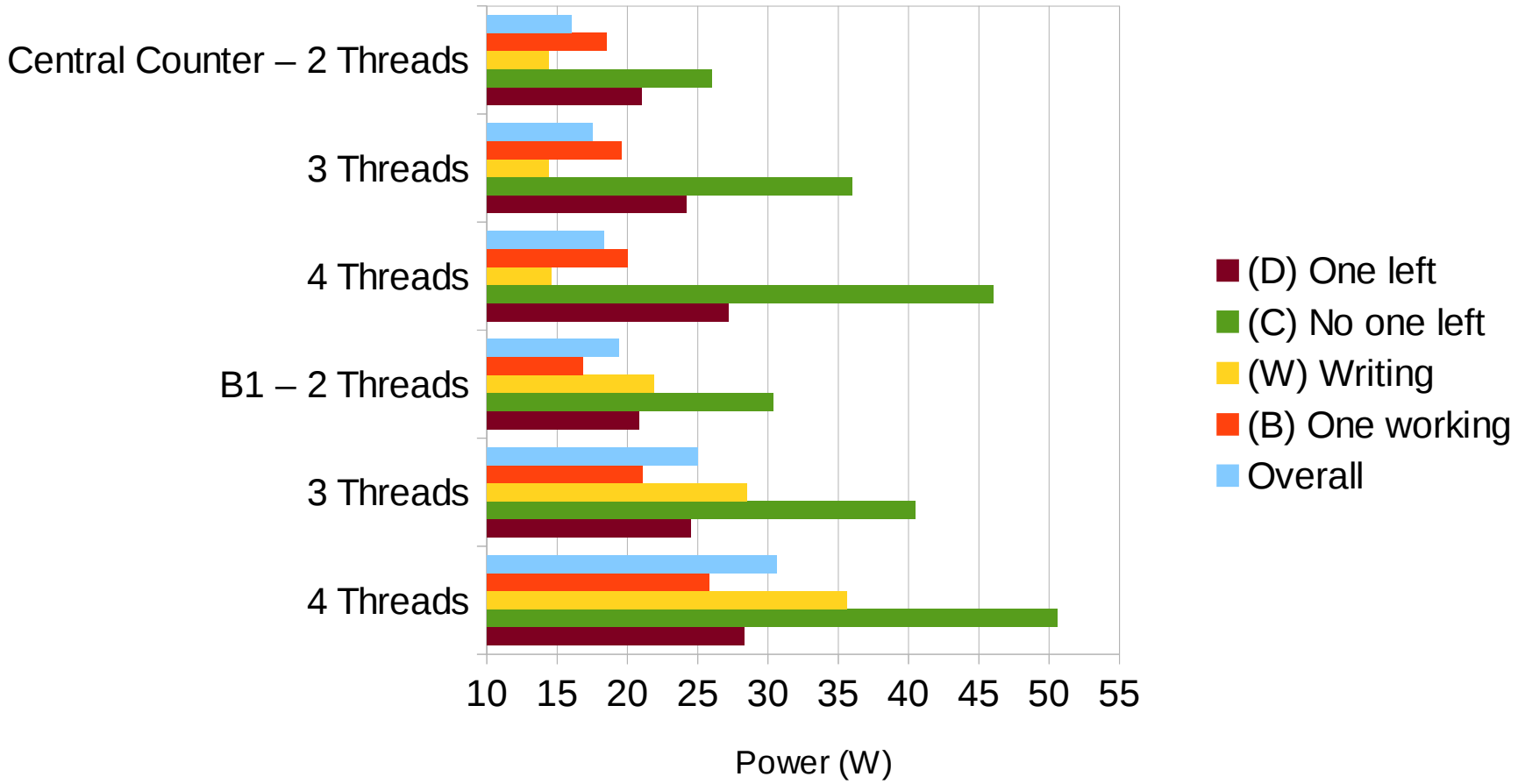
- small work period (100 cycles)



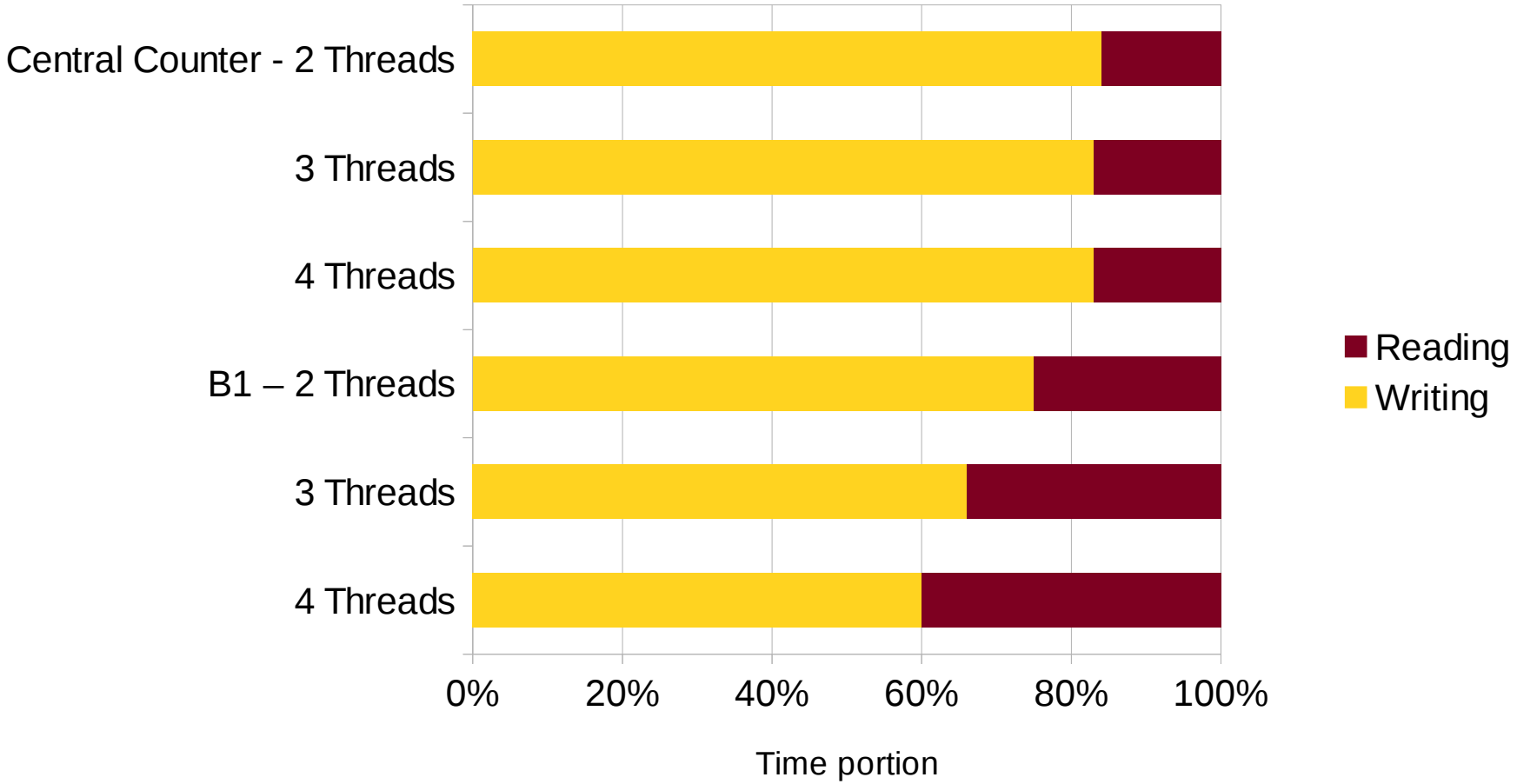
Quantitative ◁ Analysis



Quantitative ◁ Analysis

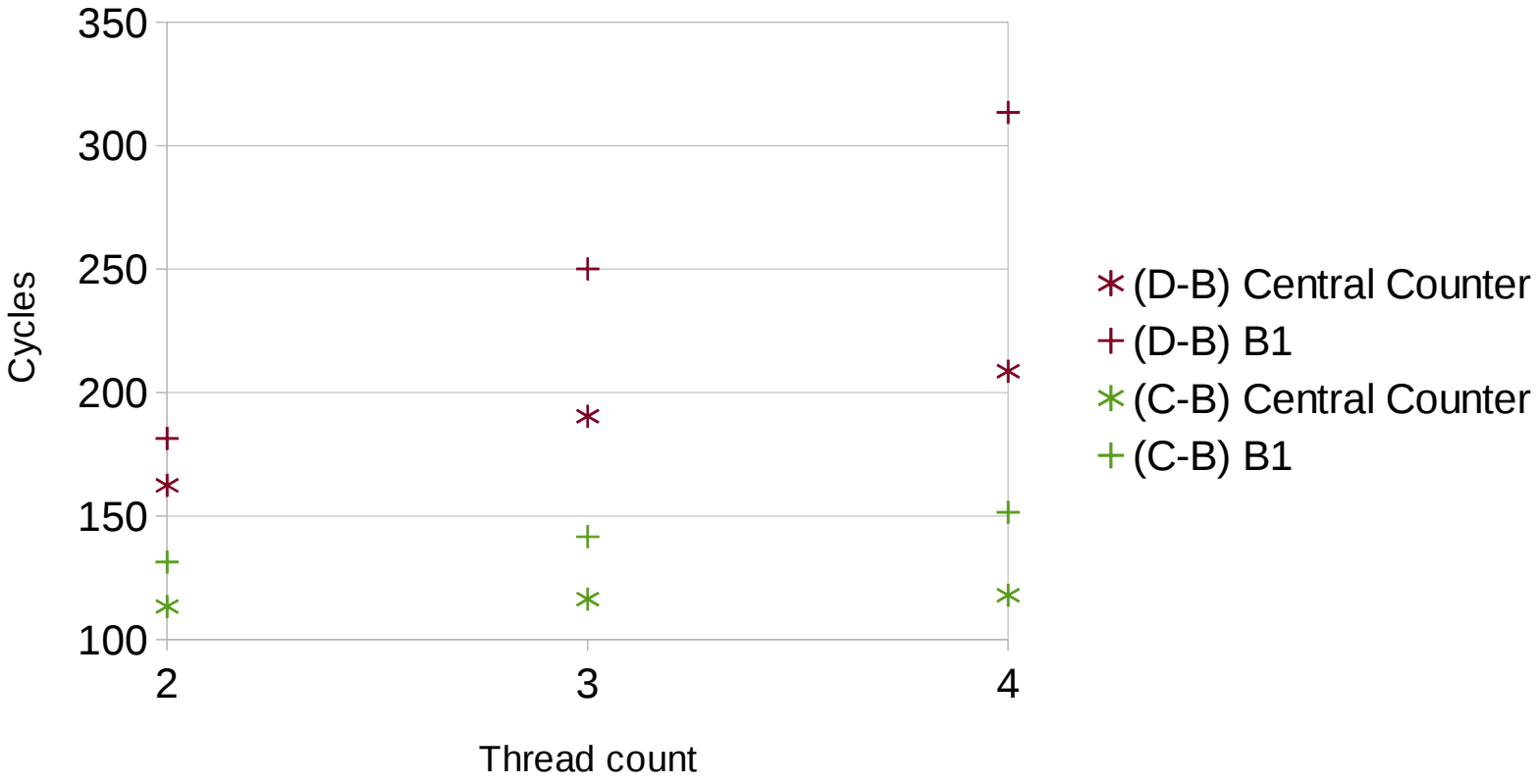


Quantitative ◁ Analysis

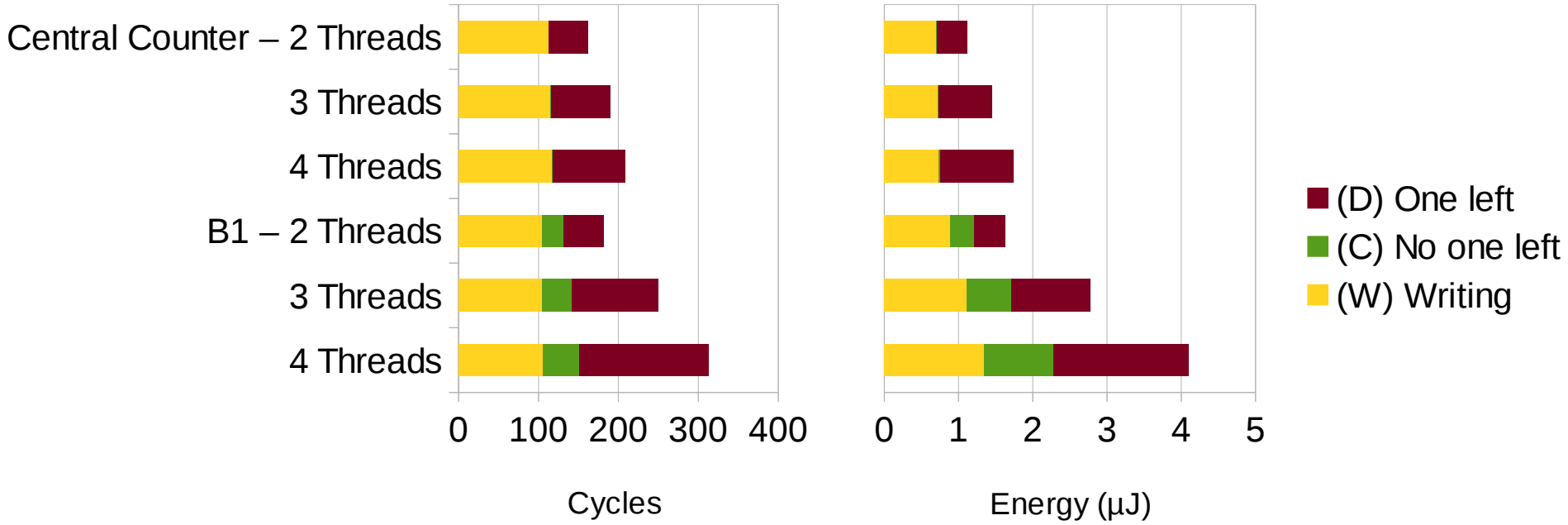


Quantitative ◁ Analysis

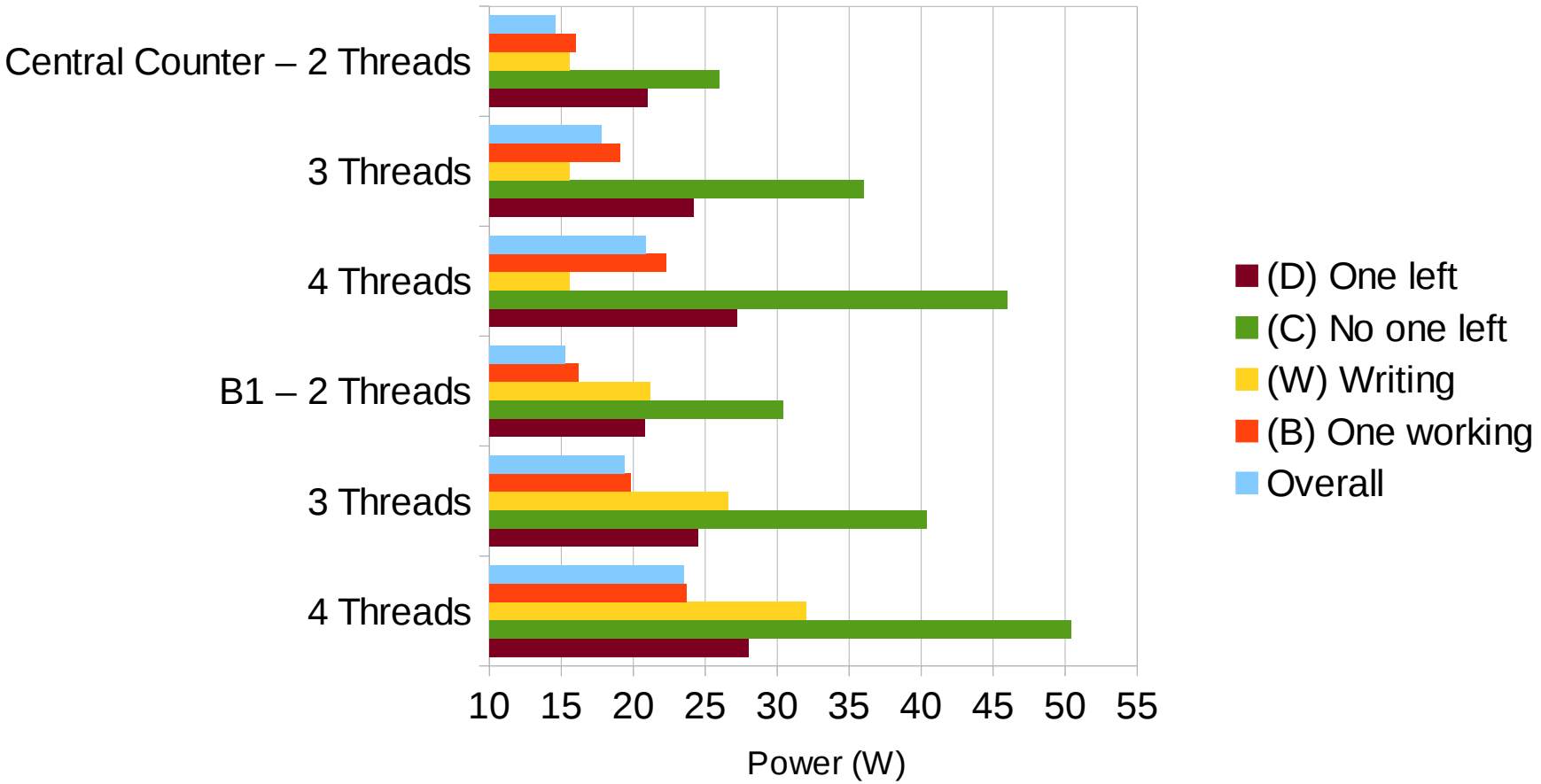
- large work period (1000 cycles)



Quantitative ◁ Analysis



Quantitative ◁ Analysis



Conclusion

- Introduced innovative barrier protocols
  - \+ No atomic operations or locks required
  - \+ Competitive performance
- Principles of pW/CS locks apt to improve synchronisation performance
- Quantitative model checking enables exhaustive, fine-grained analysis beyond the capability of tests and benchmarks

Future Work

- Analyse protocols using measurement
- Invent more protocols
  - Variations of existing
  - Remote write-based
- Extend model checking
  - More processes/threads
  - More detail
    - Cache protocols, cache hierarchies
    - Limited bandwidth and other influences

Sources

- [1] Probabilistic write copy select, Paper,

  In 13th Real-Time Linux Workshop, pages 195–206, Oct. 2011

- [2] A probabilistic quantitative analysis of probabilistic-write/copy-select, Paper,

  In NASA Formal Methods, pages 307–321. Springer, 2013.

- [3] Evaluation of publicly available Barrier-Algorithms and Improvement of the Barrier-Operation for large-scale Cluster-Systems with special Attention on InfiniBand Networks

  http://htor.inf.ethz.ch/publications/index.php?pub=12

- [4] PRISM, Website, 13-03-019

  http://www.prismmodelchecker.org

- [5] SPIN, Website, 13-01-08

  http://spinroot.com

Not Covered in the Presentation

- Survey of means to implement barriers
- Barrier building blocks

- Dissemination Barrier, MGB and B2 Barrier
- more in-depth analysis of the protocols

Thank you!

Slides and report are available at
http://automaton2000.com/barrier-slides.pdf
http://automaton2000.com/barrier-minor-thesis.pdf

Minor Thesis

Development and Analysis of Barrier Protocols

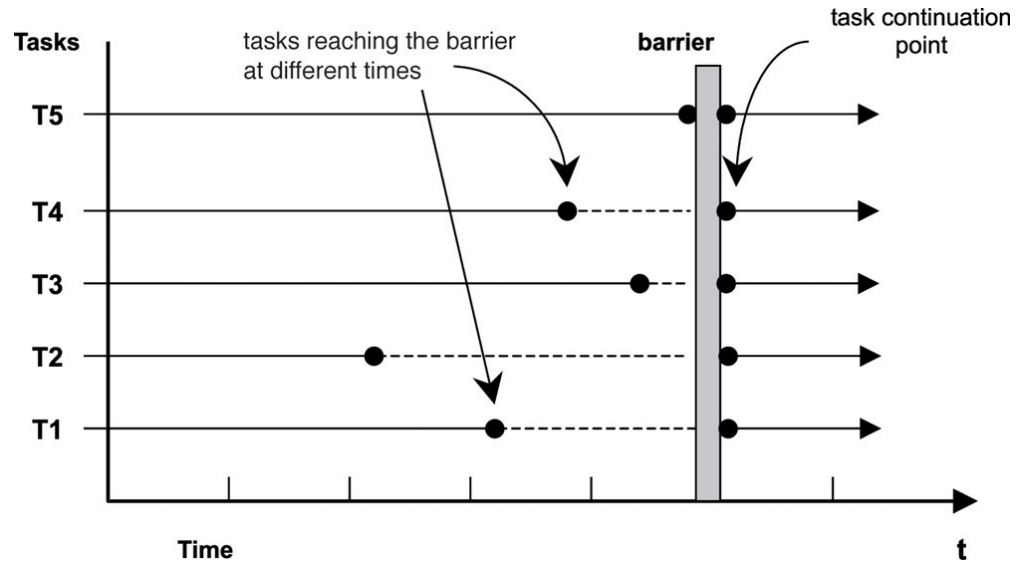
Ronny Brendel (http://automaton2000.com)
Responsible Professor: Prof. Dr. Christel Baier
Supervisor: Dr. Sascha Klüppelholz

Dresden, 2014-01-08

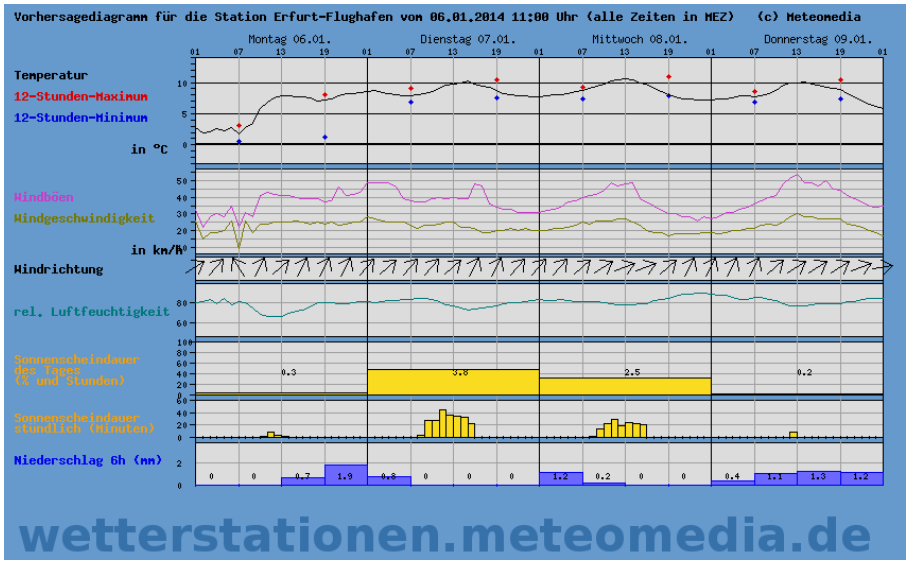
Content

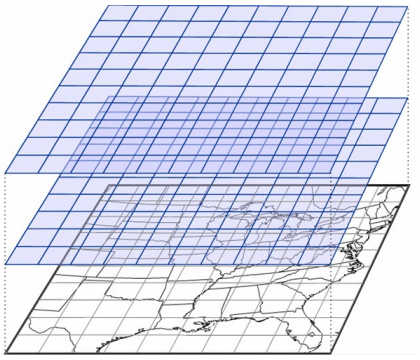
- Introduction
  - Basics, Motivation
- Protocols
  - Central Counter, B1 Barrier
- Modelling
  - Shared Variable, Protocols
- Analysis
  - Functional, Quantitative
- Conclusion, Future Work, Sources

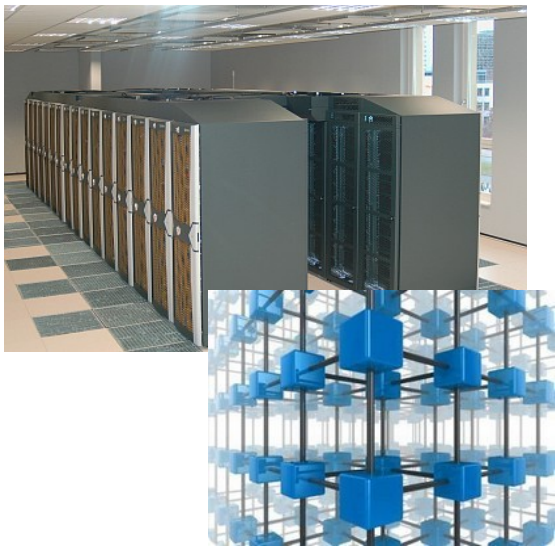
Basics ◁ Introduction



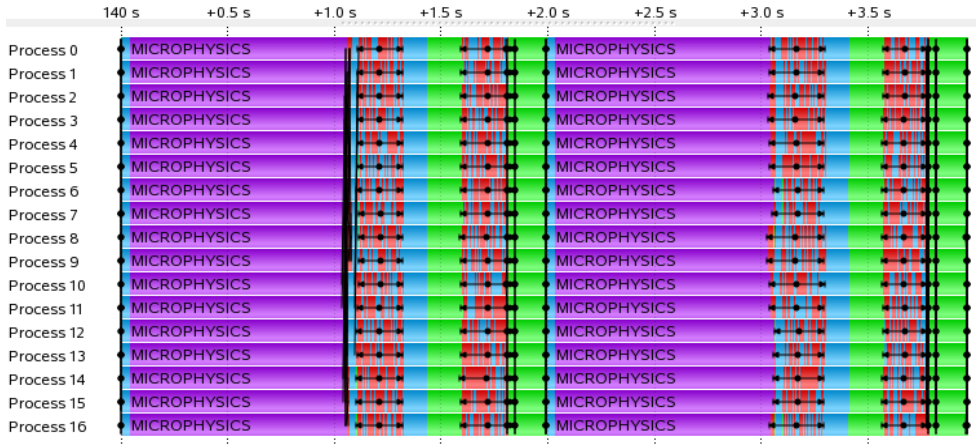
Basics ◁ Introduction





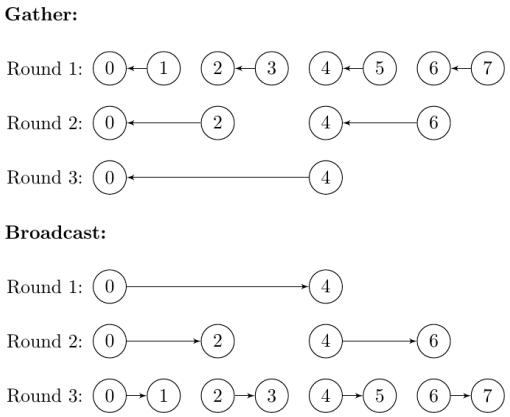


Basics ◁ Introduction



Lock-step / Gleichschritt

- Usual Implementations include
  - Central Counter Barrier (atomic increment)
  - Hierarchical approaches



see sources [3]

Motivation ◁ Introduction

- Today's Barrier Protocols have been invented long ago
- Probabilistic Write/Copy-Select (pW/CS)
  - Concurrent protocols are unnecessarily strict
  - Relieving strictness can improve performance
  - Complexity of modern computers makes the timing of concurrent interaction effectively random. Employ the tools of probability theory for designing/analysing protocols

see sources [1,2]

- Dissemination 1988
- Central Counter noch aelter
- Neue Varationen und andere Technologien, aber im Kern das selbe (RMA Dissemination, n-way Dissemination)

- Neueres workshop paper von Nicholas Mc Guire
    - locking ist kompliziert
    - locking/atomic ops skalieren schlecht
    - pW/CS lock = Alternatives locking Technik
        - Replikation
        - Fehlertoleranz
        - Wahrscheinlichkeit des Fehlers so gering dass im Durchschnit die performance besser ist.

Motivation ◁ Introduction

- Tests/Benchmarks
  - not exhaustive
  - not arbitrarily fine-grained, incurs overhead
  - testing probabilistic algorithms is hard
- Model checking
  - exhaustive
  - arbitrarily fine-grained, no overhead

- Nachteile
  - Aufwand. Messen vs Simulieren/Analysieren
  - Skaliert schlecht
  - Ergebnisqualitaet ist abhaengig von der Modellqualitaet

Motivation ◁ Introduction

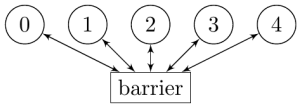
- Improve barrier protocols
- Using the principles behind pW/CS lock
- Analysis through model checking

- ein Gaengiger Algorithmus vs einen Neuen

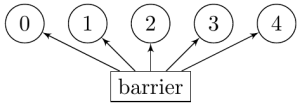
Central Counter Bar. ◁ Protocols

```
shared variables: integer barrier := threadCount
---
atomic{barrier := barrier - 1}
wait until barrier = 0
```

Atomic decrement:



Repeated reading:



!!! Wir behandeln im Vortrag nur shared memory !!!

ceil(log(2, n)) bytes

B1 Barrier ◁ Protocols

```
shared variables: boolean barrier[threadCount]
local variables:  integer i
initialisation:   barrier[*] := false
---
barrier[threadIndex] := true

i := 0
while i < threadCount {
    if barrier[i] = false {
        i := -1
    }
    i = i + 1
}
```

12

- n * 64 bytes - linear

- aehnlich wie central counter vom ablauf her
    - einmal schreiben
    - oft lesen

- schnelleres schreiben, weil keine atomic ops, gleichzeitig moeglich
- langsameres lesen, weil mehrere variablen gelesen werden muessen.
- Da nach dem schreiben einer variable nie wieder in die gleiche Variable geschrieben wird, wird es in jedem thread gecacht und ist dann schnell verfuegbar

Modelling

- Functional
  - Non-deterministic transition system + LTL
  - SPIN
  - detailed model to reveal all possible mistakes
- Quantitative
  - CTMC + CSL/CSRL
  - PRISM
  - reduced to just costly/important transitions, no reinitialisation

see sources [4,5]

- Folgend hauptsaechlich quantitatives Modelling
- Funktionales ist aehnlich, aber einfacher

Shared Variable ◁ Modelling

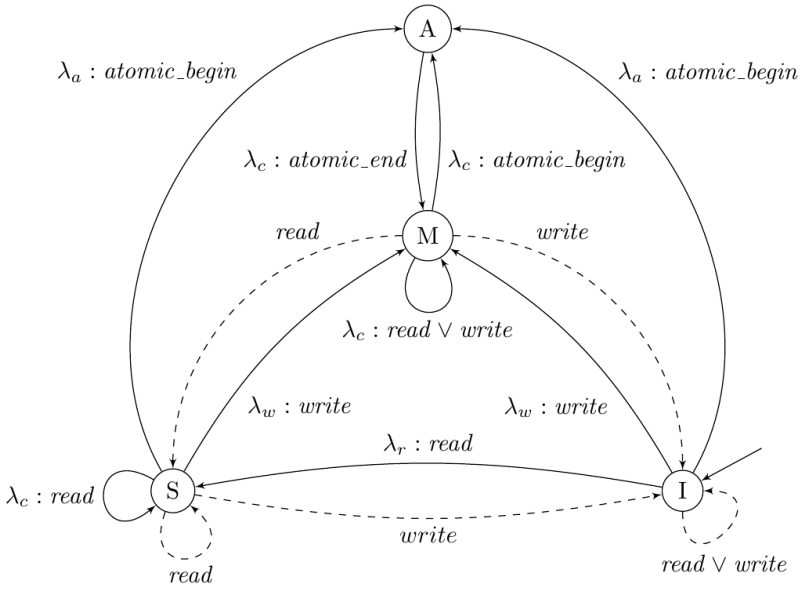
- Synchronisation is about exchanging information, i.e. sharing memory
- Very small information → Timing dominated by memory access latency
- Memory access is cached → We have to model caching
- We identify a shared variable with the cache line it resides on
- MSI protocol + atomic operations

- kein/wenig “Rechnen”

- Cache line Erklaerung
    - 64 byte Einheit von Speicher die am Stueck eingelagert und verdraengt werden.
    - Adressen genau auf 64byte ausgerichtet

    - Eine Kopie pro Core/Cache. Synchronisation zwischen den Kopien noetig damit keine Veralteten Daten benutzt werden. Beispiel: Jemand schreibt und du benutzt ein altes Datum.

Shared Variable ◁ Modelling

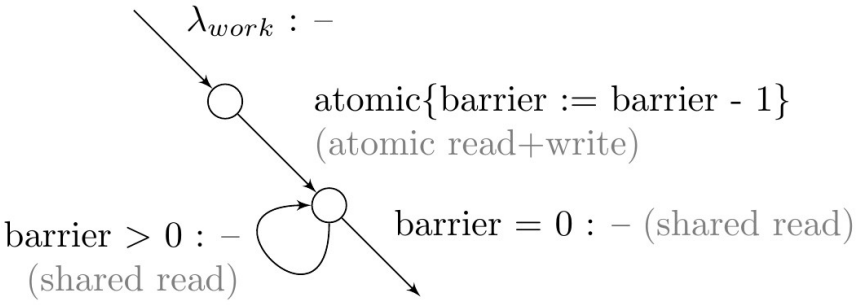


- CTMC Modul
- Ein Model setzt sich aus mehreren CTMC Modulen, die aus ueberlappt ausgefuehrt werden. Synchronisation zwischen Modulen ueber action labels.
- Eine cache line Kopie pro thread und pro Variable
- Synchronisation zwischen allen Modulen der gleichen Variable ueber action labels

- λ_c = 1 cycle
- λ_r = 50 cycles
- λ_w = 100 cycles

- gestrichelte Linien bekommen ihre Rate von der ausloesenden Aktion auf der ausloesenden cache line Kopie

Central Counter Bar. ◁ Modelling

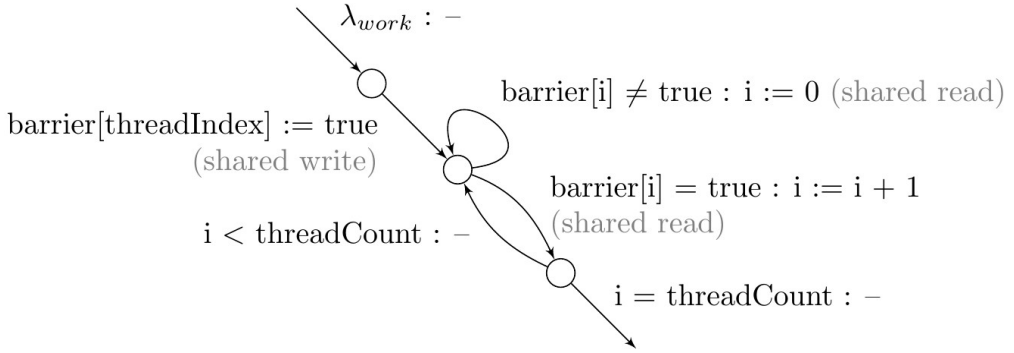


- Ein Protokoll-Modul pro thread
- Ein cache line Kopie Modul pro Thread, weil nur eine Variable

- Rate von shared memoryOperationen sind abhaengig vom cache line Kopie Zustand der jeweiligen Variable

- work ist Verteilung der Ankunftszeiten der threads an der Barriere
- Ungleichgewicht da Arbeit unterschiedlich schnell fertig gestellt wird

B1 Barrier ◁ Modelling



- Ein cache line Kopie Modul fuer jedes Array Element
    - n Protokol Module
    - n*n cache line Kopie Module

Functional $\triangleleft$ Analysis

- “A thread may only exit the barrier if all threads have entered it”

$$\Box(one\_left \implies all\_entered)$$

- “If all threads entered the barrier, each one leaves it in a finite amount of time”

$$\Box(all\_entered \implies \Diamond all\_left)$$

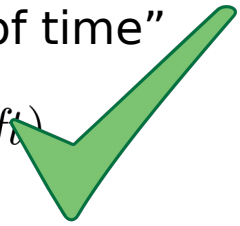
Functional ◁ Analysis

- “A thread may only exit the barrier if all threads have entered it”

$$\Box(one\_left \implies all\_entered)$$

- “If all threads entered the barrier, each one leaves it in a finite amount of time”

$$\Box(all\_entered \implies \Diamond all\_left)$$



Quantitative ◁ Analysis

- Interesting values
  - time in cycles (2.5GHz clock speed)
  - energy consumption in joule / watt
  - implemented using CTMC time and rewards
- Parameters to variate
  - Number of threads
  - Rate of the initial (work) distribution

- kurz rewards erklaeren
  - transitions system bekommt 2 neue Funktionen
  - state, trans reward ...

Quantitative ◁ Analysis

- Points in time when to measure
  - `(A)` First thread entered the barrier
  - `(B)` Last thread entered
  - `(C)` First thread left
  - `(D)` Last thread left
  - `(W)` Last thread finished writing
- We take the reachability reward wrt the proper reward functions at these points in time

- (A), (B) bezogen auf Arbeitsperiode ueberwinden

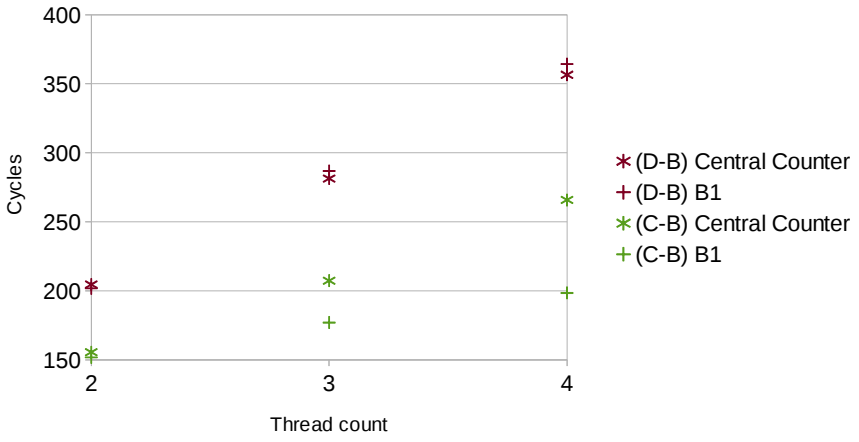
- Average reward

- “Proper” = reward functions die die richtigen Werte repraesentieren fuer Zeit und Energieverbrauch

11 Watt baseline. 2 nJ for fast shared mem ops. 200 nJ for slow shared mem ops

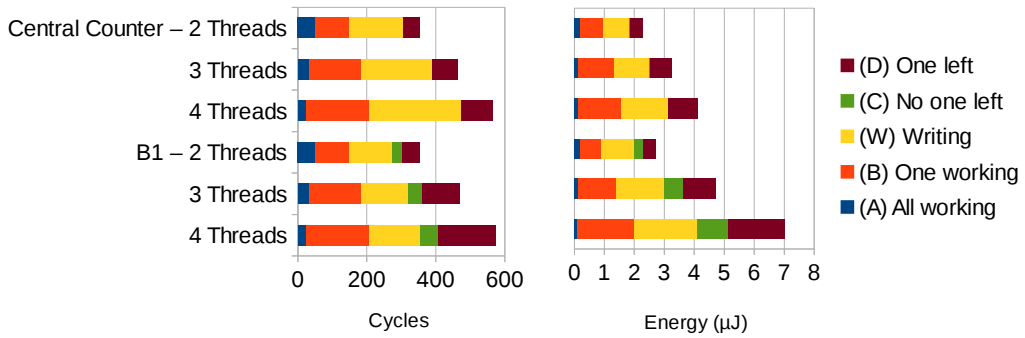
Quantitative ◁ Analysis

- small work period (100 cycles)



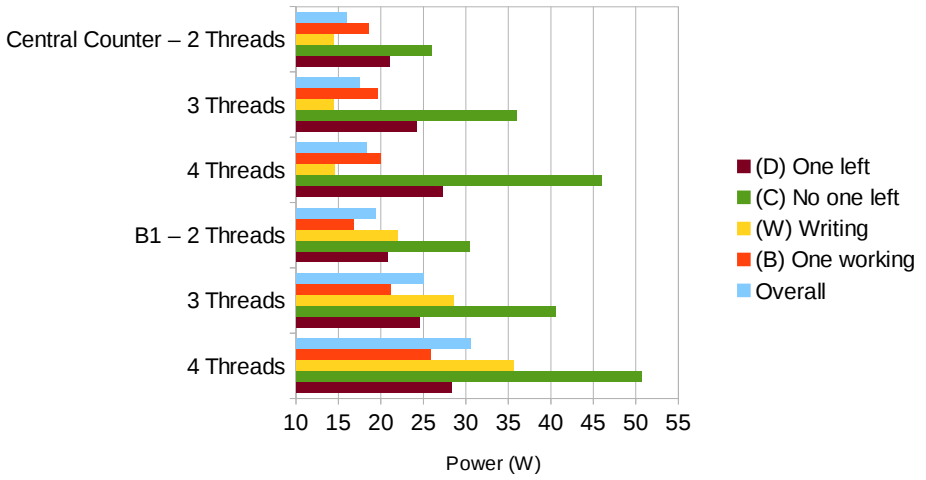
- C-B letzter der betritt bis zum ersten der verlaesst
- D-B letzter der betritt bis zum letzten der verlaesst
- 150 cycles minimum (2 shared writes)

Quantitative ◁ Analysis



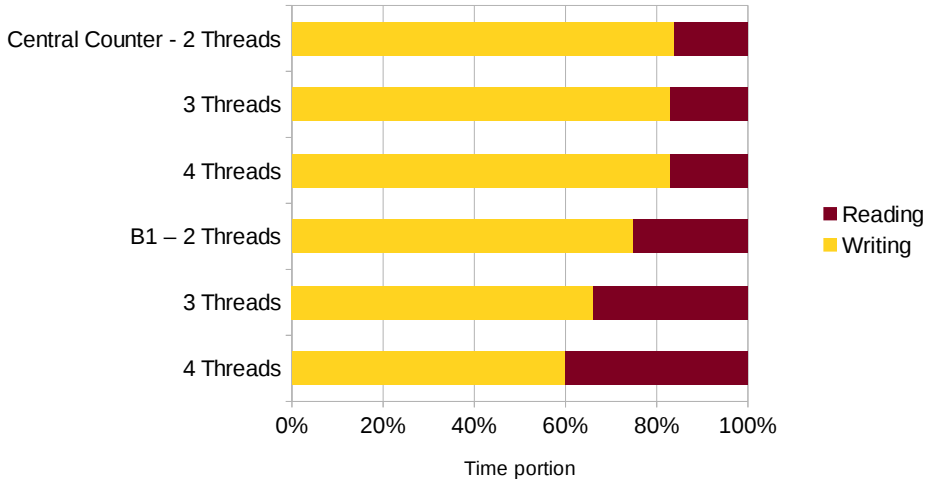
- A B W C D nochmal erklaeren (andere Wortwahl in der Legende)
- Energie aehnlich wie Timing
- atomic op blockieren spart Energie
- A, B nur abhaengig von der thread Anzahl
- kein Gruen fuer CC, weil letzter ankommer verlaesst die barrier (mit hoher wahrschl.) sofort
- Mit steigender thread-Anzahl steigt die Laufzeit aller Operationen wegen der CTMC Semantik zu gleichzeitig aktivierten Transitionen
    - semantik erklaeren
    - zB CC Phase (D) 50, 75, 92 cycles
    - Erwartung waere 50 fuer alle
    - zB Phase A = 100 / n, B = sum_{i=1}^{n} 100/i
    - deswegen dauert (W) 125, 138, 146 Takte statt 100 ueberall wie erwartet
- Lesen / Schreiben ... Vermutung reiterieren.

Quantitative ◁ Analysis



- (A) has the 11 watts baseline
- overall CC 16-18.3 watts, B1 19.4-30 watts
- (B) wie erwartet ... busy waiting / concurrent write / atomic ops
- (W) atomic ops + busy wait vs concurrent write + busy wait
- (C) sehr kurz / viele shared reads (verbrauchen mehr energie)
- (D) aehnlich, weil bei beiden nur lesen

Quantitative ◁ Analysis



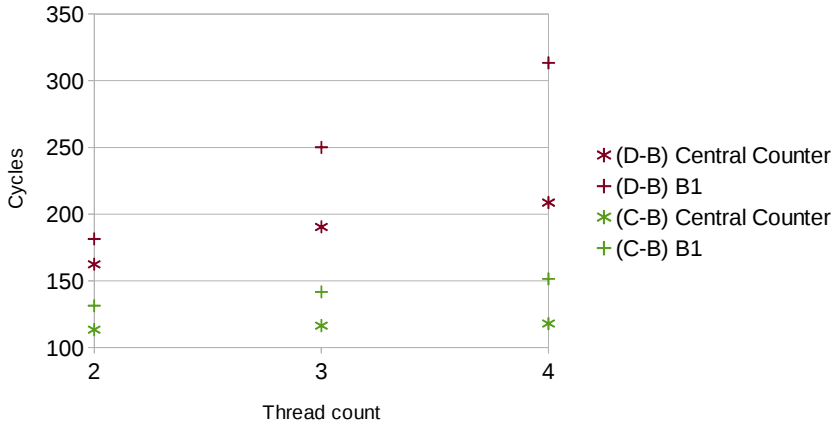
25

- W = (W) = first enters to last wrote
- R = (D-W) = after W until finish

- Erwartung CC schreiben wird groesser ... passiert nicht wegen CTMC Semantik ((D) wird groesser mit steigendem core count wie vorher erklaert)
- B1 wie erwartet

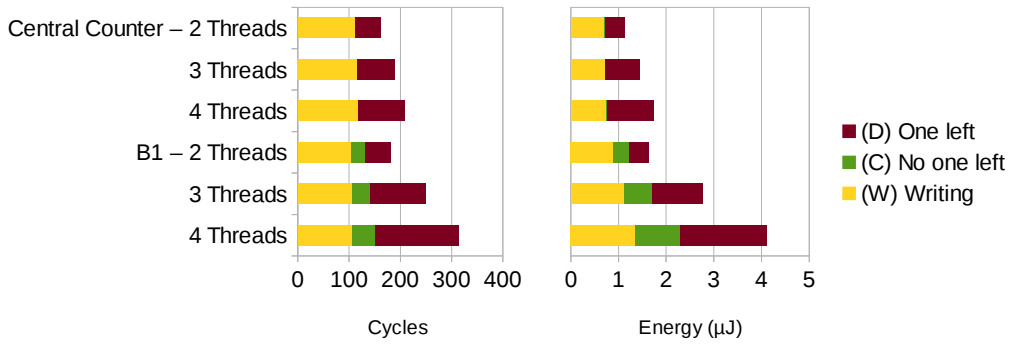
Quantitative ◁ Analysis

- large work period (1000 cycles)



- keine/wenige ueberlappende schreib operationen
+ langsames lesen -> alles langsamer

Quantitative ◁ Analysis



27

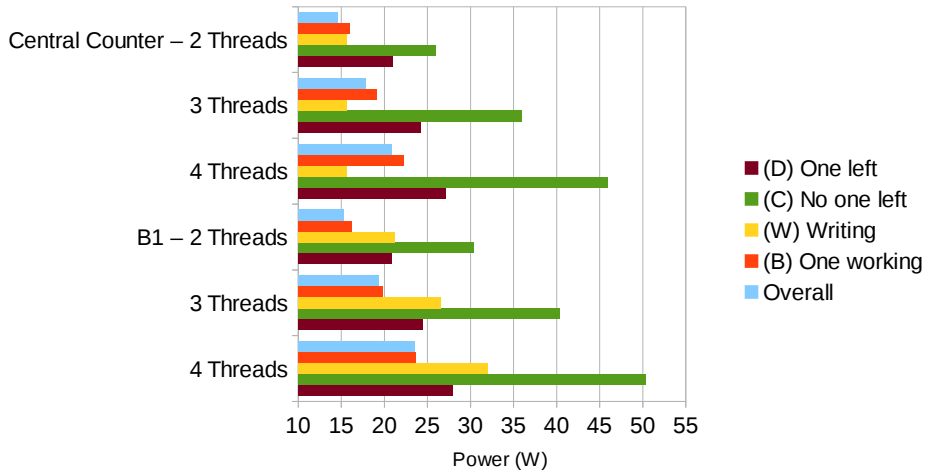
- (W) CC = 1 atomic op
- (W) B1 = 1 write

- (C-W) and (D-C) gleich wie bei 100 Takten, weil diese unabhaengig von der Eintrittszeit ist.

(Sobald alle drinnen sind faengt ist W zu Ende und C faengt an)

- Energie (W) B1 steigt (Zeit nicht) weil mehr threads -> mehr busy waiting

Quantitative ◁ Analysis



28

- overall ausgeglichener weil grosse Arbeitsperiode mit 11 watt
    - CC = 14,6 - 20.9
    - B1 = 15.3 - 23.5
- B, C, D sehr aehnlich zwischen beiden
- W anders weil lesen und schreiben blockiert ist waehrend atomic ops -> nicht so bei B1

- je mehr Arbeit desto weiter naehern sich der Energieverbrauch zwischen den beiden Protokollen an

Conclusion

- Introduced innovative barrier protocols
  - \+ No atomic operations or locks required
  - \+ Competitive performance
- Principles of pW/CS locks apt to improve synchronisation performance
- Quantitative model checking enables exhaustive, fine-grained analysis beyond the capability of tests and benchmarks

Future Work

- Analyse protocols using measurement
- Invent more protocols
  - Variations of existing
  - Remote write-based
- Extend model checking
  - More processes/threads
  - More detail
    - Cache protocols, cache hierarchies
    - Limited bandwidth and other influences

Sources

- [1] Probabilistic write copy select, Paper,

  In 13th Real-Time Linux Workshop, pages 195–206, Oct. 2011

- [2] A probabilistic quantitative analysis of probabilistic-write/copy-select, Paper,

  In NASA Formal Methods, pages 307–321. Springer, 2013.

- [3] Evaluation of publicly available Barrier-Algorithms and Improvement of the Barrier-Operation for large-scale Cluster-Systems with special Attention on InfiniBand Networks

  http://htor.inf.ethz.ch/publications/index.php?pub=12

- [4] PRISM, Website, 13-03-019

  http://www.prismmodelchecker.org

- [5] SPIN, Website, 13-01-08

  http://spinroot.com

Not Covered in the Presentation

- Survey of means to implement barriers
- Barrier building blocks

- Dissemination Barrier, MGB and B2 Barrier
- more in-depth analysis of the protocols

Thank you!

Slides and report are available at
http://automaton2000.com/barrier-slides.pdf
http://automaton2000.com/barrier-minor-thesis.pdf