Brandenburg
University of Technology
Cottbus - Senftenberg

15. März 2016

MYTHOS

VIELKERN-BETRIEBSSYSTEME FÜR HPC

Stefan Wesner, Lutz Schubert, Vladimir Nikolov, Jörg Nolte,
Randolf Rotta, Mai Krüger, Robert Kuban, uvm. . .

Brandenburg
University of Technology
Cottbus - Senftenberg

Ziel dynamische HPC Anwendungen auf Vielkern-Prozessoren

Fokus schnelle Thread-Verwaltung

- Durchsatz statt Echtzeit, Overheads reduzieren
- BMBF Projekt[1] von Uni Ulm, mit Uni Siegen, BTU Cottbus, Alcatel-Lucent Deutschland AG, HLRS Stuttgart

[1] Fördernummer BMBF 01IH13003

AGENDA

Brandenburg
University of Technology
Cottbus - Senftenberg

1. Wozu Vielkern-Betriebssysteme?
- Herausforderungen - Vielkern-Prozessoren
- Anforderungen - Zwei Szenarien
- Konzequenzen - Entwurfsziele

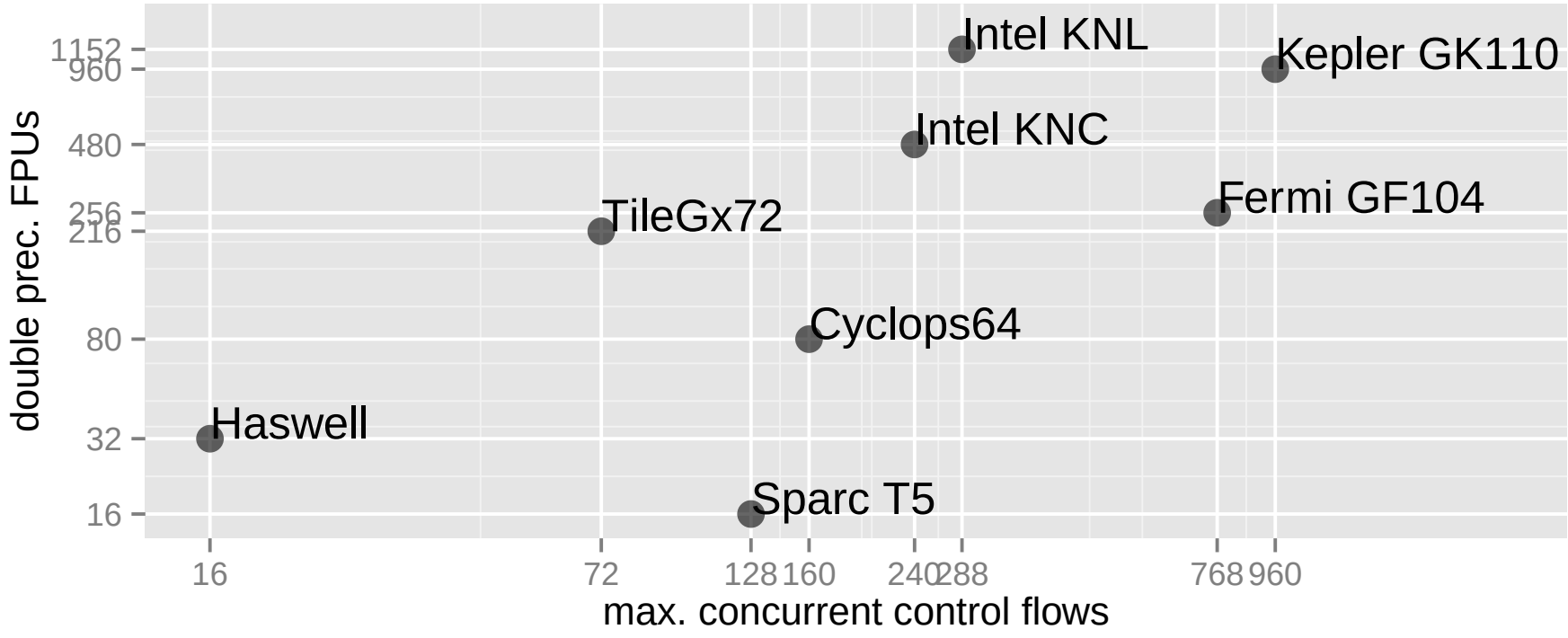
2. Zwischenergebnisse
- Die Kunst der Weglassens
- Wartefreie Aufgabenverwaltung
- Pragmatische Kompromisse

3. Zusammenfassung

OUTLINE

Brandenburg
University of Technology
Cottbus - Senftenberg

MULTI- VERSUS MANY-CORE



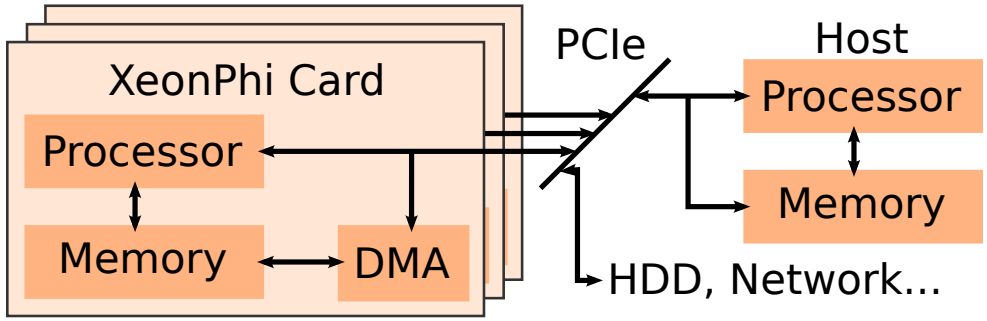
- max. paralleler Durchsatz in geg. Power- & Platz-Budget
- Energie- & Platzeffizienz statt Single-Thread Performance

⇒ Durchsatz nur durch Aufgaben- und Daten-Parallelität

MYTHOS ZIELARCHITEKTUR

Intel Xeon Phi Knights Corner

Brandenburg
University of Technology
Cottbus - Senftenberg



- 60 Kerne mit je 512bit VPU, 4 HW Threads, 1.05GHz
  In-Order, keine Sprungvorhersage, 2 Takte Instruction Decoder
- 8GB kohärenter gemeinsamer Speicher, Streuung über Verzeichnisse
  150ns–1000+ns Latenz, Flaschenhals für Synchronisation
- PCIe2.0 Karten: eigene BS Kernel einfach startbar,
  Kommunikation mit Host über nicht-kohärenten gem. Speicher

$\Rightarrow$ Ideale Worst-Case Plattform!?

SZENARIO I: HPC SIMULATIONEN

Moleküldynamik (HLRS), Strömungen und Elektrodynamik (Uni Siegen)

Brandenburg University of Technology Cottbus - Senftenberg

- eine Anwendung besitzt “alle” Kerne und Speicher
- Aufgabenverwaltung und Kommunikation auf Anwendungsebene
- dynamische Speicherverwaltung

Anforderungen

- effizientes Warten statt Polling (min Energie, min Contention)
  Semaphore/Futex `wait`, `signal`
- viele Threads manipulieren gemeinsamen Adressraum (IO,PGAS,DSM)
  `mmap`, `mremap`, `mprotect`

SZENARIO II: COMPUTE CLOUD

Verteilte Medienverarbeitung (Alcatel/CenoLabs)

Brandenburg University of Technology Cottbus - Senftenberg

- dynamisch konfigurierter Datenfluss-Graph
- Knoten aktiviert durch Input Daten, laufen in Isolation+Zeitschranke
- Cloud Manager bewegt Daten, migriert Knoten für min. Energie

Anforderungen

- viele Threads mit eigenem Adressraum erzeugen und aktivieren
  `create`, `destroy`, `activate` mit Zeitschranke
- Manager manipuliert fremde Adressräume, z.B. zero-copy comm
  `mmap`, `mremap`, `mprotect` auf Kind-Threads
- Weiterleiten der Systemaufrufe, Exceptions etc. an Manager
  `event queue`

MYTHOS DESIGN-ZIELE

Brandenburg University of Technology Cottbus - Senftenberg

1. Overheads reduzieren durch Weglassen:
   Policies vereinfachen und auslagern
   z.B. kein Demand-Paging, kein Swapping, nur FIFO Scheduling
2. BS-Aufgaben parallelisieren:
   Aufräumaufgaben über freie Kerne verteilen
   z.B. Rahmen freigeben und nullen
3. Wartezeiten sinnvoll füllen:
   Asynchrone BS-Aufgaben und Kommunikation
   z.B. Active Messages, Delegation Queues, Continuation Passing
4. Kommunikationskosten reduzieren:
   kleine Aufgaben verschicken statt viele Daten bewegen
   lokal gemeinsame Caches ausnutzen

OUTLINE

Brandenburg
University of Technology
Cottbus - Senftenberg

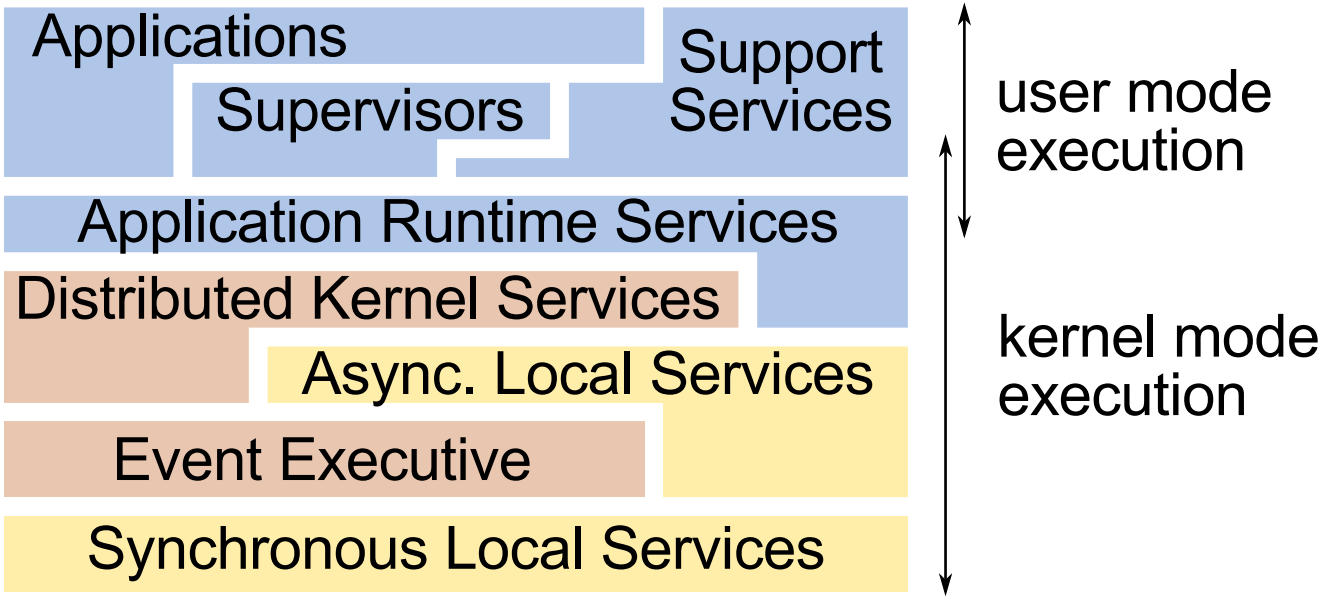
1. Wozu Vielkern-Betriebssysteme?
   - Herausforderungen - Vielkern-Prozessoren
   - Anforderungen - Zwei Szenarien
   - Konzequenzen - Entwurfsziele

2. **Zwischenergebnisse**
   - Die Kunst der Weglassens
   - Wartefreie Aufgabenverwaltung
   - Pragmatische Kompromisse

3. Zusammenfassung

MEHRSCHICHTIGE ARCHITEKTUR

Lokale und Verteilte Dienste separieren, Steuerung von Außen



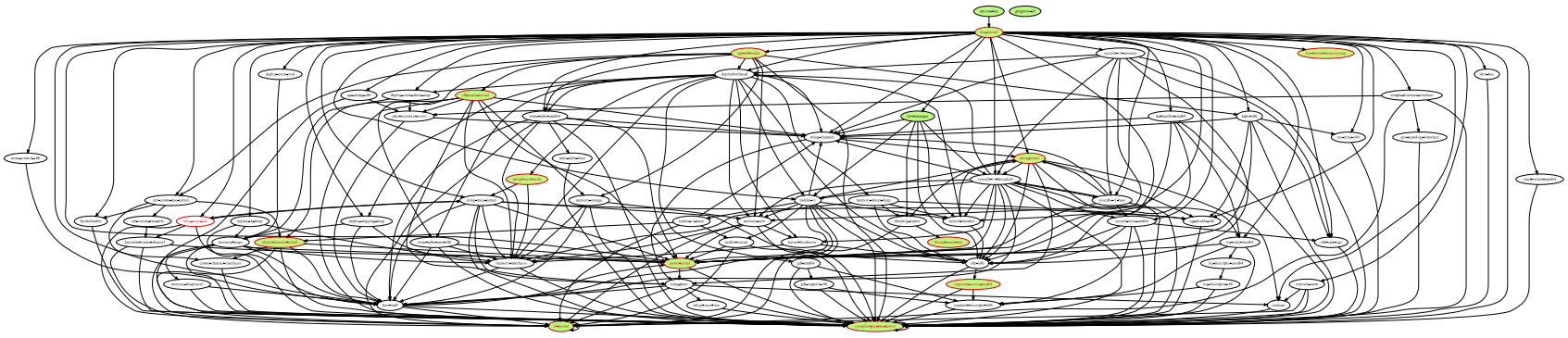
DIE KUNST DER WEGLASSENS

mcconf: Kernel maßschneidern für Platform und Anwendung

Brandenburg
University of Technology
Cottbus - Senftenberg

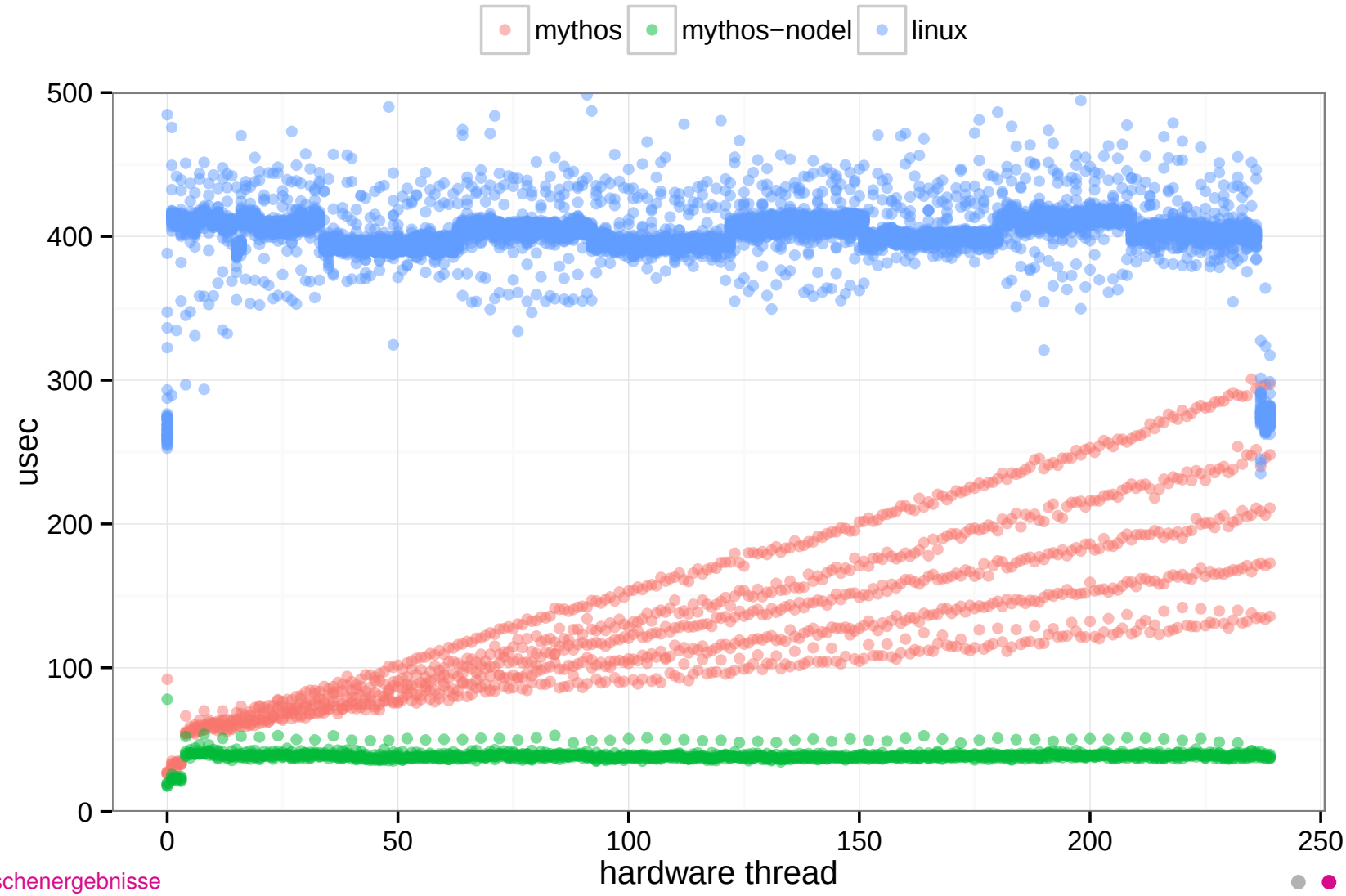
Komponenten Vielfalt durch Laufzeit-Komposition
einfaches Replizieren & Platzieren auf Kerne

Module Effizienz durch Compile-Time-Komposition
Abhängigkeitsverwaltung mit `#include`,
Varianten filtern anhand Prozessor, Umgebung. . .



FORK-JOIN BENCHMARK

Einen Thread erzeugen und aktivieren; Freigeben kostet noch. . .



WARTEFREIE AUFGABENVERWALTUNG

Asynchrones Ausführungsmodell mit lokaler Balanzierung

Brandenburg University of Technology Cottbus - Senftenberg

- Ereignisse, Aktive Nachrichten, Kritische Abschnitte, Interrupt-Epiloge, Continuations...

⇒ “Tasklets” basierend auf C++11 Lambdas

- pro HW-Thread: Kernel-, User-, Idle-Warteschlangen
- Thread-Gruppen mit Work Sharing/Stealing, Delegation Locks regeln lokale Nebenläufigkeit

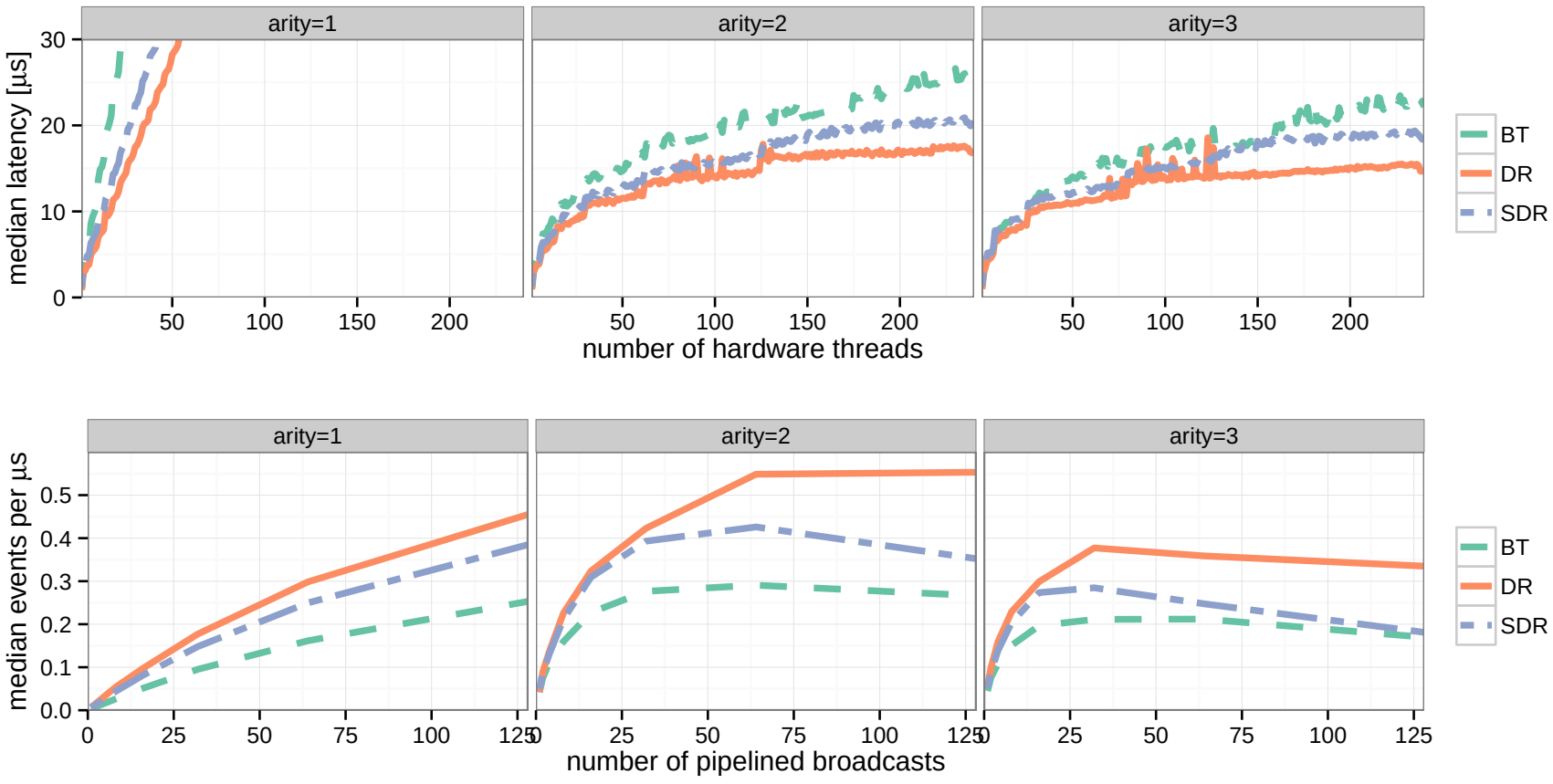
```
threads[40]->send( [](){ mlog::kernel.info("Hello"); } );

Mutex* m;
m->protect(
  [=](){ process_event(args...); ... }
);
```

BROADCAST BENCHMARK

Latenz und Durchsatz auf dem XeonPhi

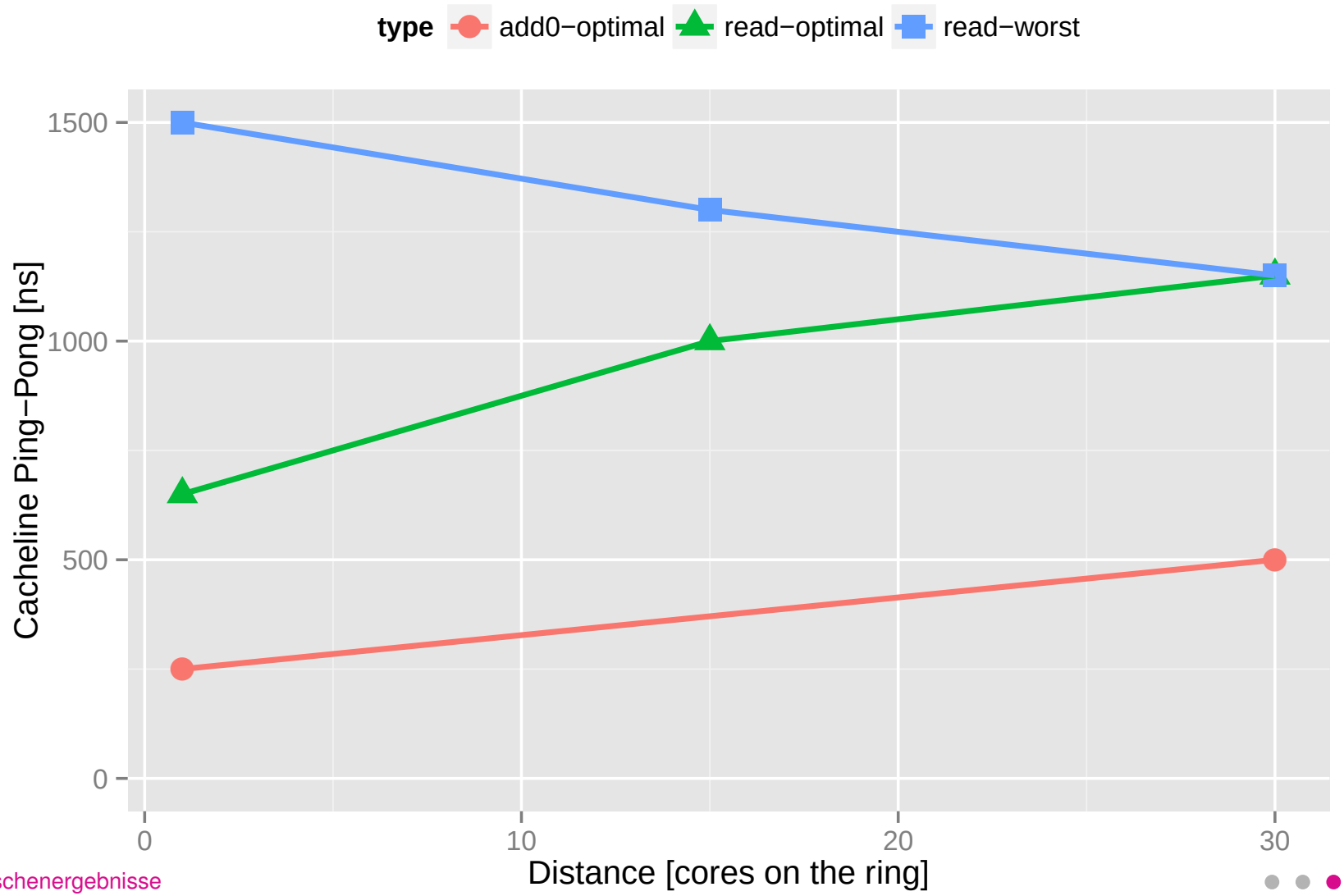
Brandenburg University of Technology Cottbus - Senftenberg



SYNCHRONISATIONSLATENZ

Geschickte Platzierung der Synchronisationsvariablen

Brandenburg University of Technology Cottbus - Senftenberg



PRAGMATISCHE KOMPROMISSE...

- keine echten Prioritäten, keine Flusskontrolle
  $\Rightarrow$ zentrale Dienste beginnen neue Aufgaben statt alte abzuschließen
- doch viele Dienste und Policies im Kernel implementiert
  $\Rightarrow$ duplizierte Infrastruktur kernel- vs. user mode
- 8GiB/240 = 34MiB pro HW-Thread
  $\Rightarrow$ kein reiner Multikernel, gemeinsamer Speicherpool
- aktives Work-Stealing stört andere Threads

Einsichten

- statische Resourcenplanung wie in Echtzeitsystemen
- Microkernel mit optionalen kernel-mode Tasks
- wartefreie Datenstrukturen für Speicherpools

OUTLINE

Brandenburg
University of Technology
Cottbus - Senftenberg

1. Wozu Vielkern-Betriebssysteme?
   - Herausforderungen - Vielkern-Prozessoren
   - Anforderungen - Zwei Szenarien
   - Konzequenzen - Entwurfsziele

2. Zwischenergebnisse
   - Die Kunst der Weglassens
   - Wartefreie Aufgabenverwaltung
   - Pragmatische Kompromisse

3. Zusammenfassung

ZUSAMMENFASSUNG

Brandenburg
University of Technology
Cottbus - Senftenberg

- Vielkern-Prozessoren steigern die Herausforderungen
- MyThOS als Experiment-Plattform für parallele BS
- Zeigt Potentiale für dynamische Anwendungen

ZUSAMMENFASSUNG

Brandenburg
University of Technology
Cottbus - Senftenberg

- Vielkern-Prozessoren steigern die Herausforderungen
- MyThOS als Experiment-Plattform für parallele BS
- Zeigt Potentiale für dynamische Anwendungen

Vielen Dank für Ihre Fragen!