School of Computation, Information and Technology
Technische Universität München

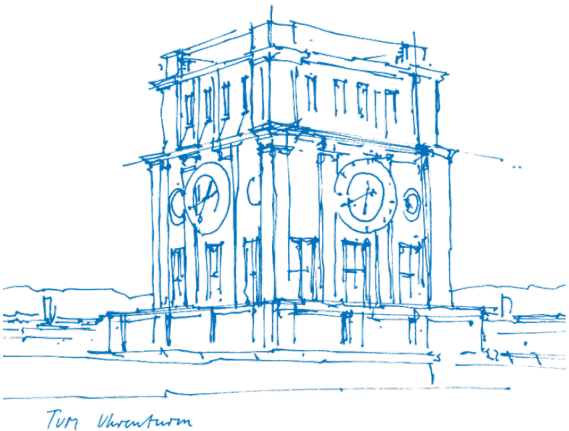
Übung 10: Parallelisierung

Einführung in die Rechnerarchitektur

Niklas Ladurner

School of Computation, Information and Technology
Technische Universität München

3. Januar 2025

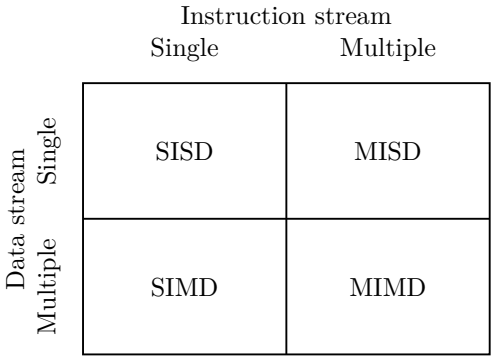


Keine Garantie für die Richtigkeit der Tutorfolien.

Bei Unklarheiten/Unstimmigkeiten haben VL/ZÜ-Folien recht!

Parallelisierungstechniken

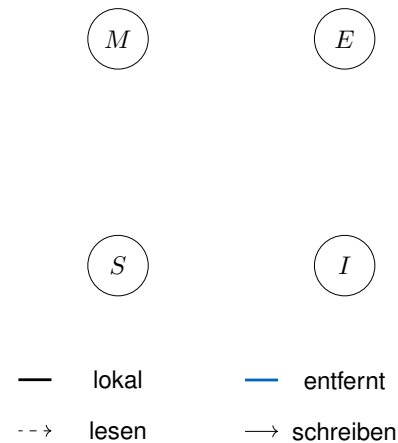
- Single-Threaded Rechenleistung immer weiter durch physikalische Limits eingeschränkt
- Optimierungen: Pipelining, Out-of-Order-Processing, Ausnutzen von Parallelität
- SIMD: Eine Instruktion, die gleichzeitig auf mehrere Daten ausgeführt wird (mehr dazu in GRA)



Quelle: A Taxonomy of Reconfigurable Single-/Multiprocessor Systems-on-Chip

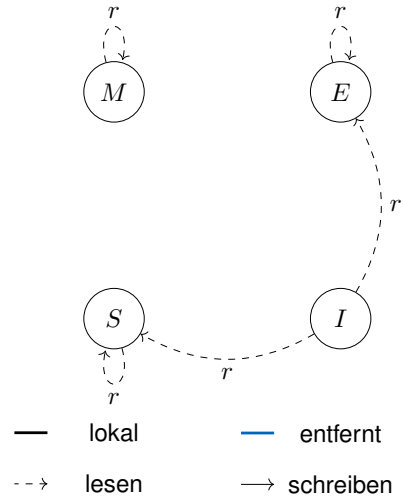
MSI/MESI

- Mehrkernsysteme: Was wenn CPU1 und CPU2 beide ein Datum gecached haben und es modifizieren?
  $\rightarrow$ Cache-Inkonsistenzen
- Einführung von Zuständen für Cachezeilen
- CPUs hören jeweils die Zugriffe der anderen Kerne ab („Bus Snooping“)
- **M**odified, (**E**xclusive), **S**hared, **I**nvalid
- Exclusive-Bit ermöglicht kleineren Overhead wenn CPUs auf verschiedenen Cache-Blöcken arbeiten



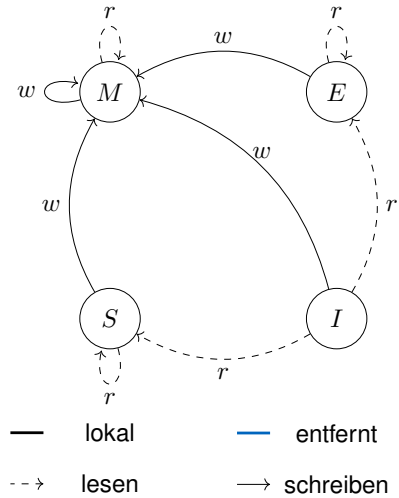
MSI/MESI

- Mehrkernsysteme: Was wenn CPU1 und CPU2 beide ein Datum gecached haben und es modifizieren?
  $\rightarrow$ Cache-Inkonsistenzen
- Einführung von Zuständen für Cachezeilen
- CPUs hören jeweils die Zugriffe der anderen Kerne ab („Bus Snooping“)
- **M**odified, (**E**xclusive), **S**hared, **I**nvalid
- Exclusive-Bit ermöglicht kleineren Overhead wenn CPUs auf verschiedenen Cache-Blöcken arbeiten



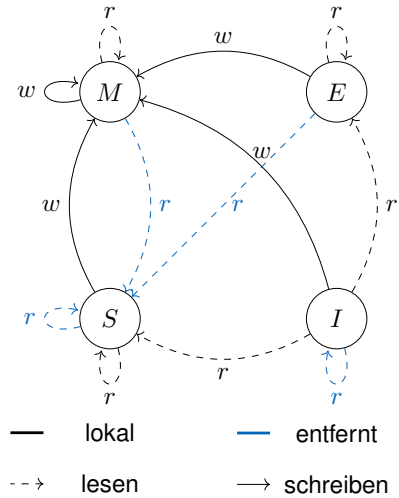
MSI/MESI

- Mehrkernsysteme: Was wenn CPU1 und CPU2 beide ein Datum gecached haben und es modifizieren?
  $\rightarrow$ Cache-Inkonsistenzen
- Einführung von Zuständen für Cachezeilen
- CPUs hören jeweils die Zugriffe der anderen Kerne ab („Bus Snooping“)
- **M**odified, (**E**xclusive), **S**hared, **I**nvalid
- Exclusive-Bit ermöglicht kleineren Overhead wenn CPUs auf verschiedenen Cache-Blöcken arbeiten



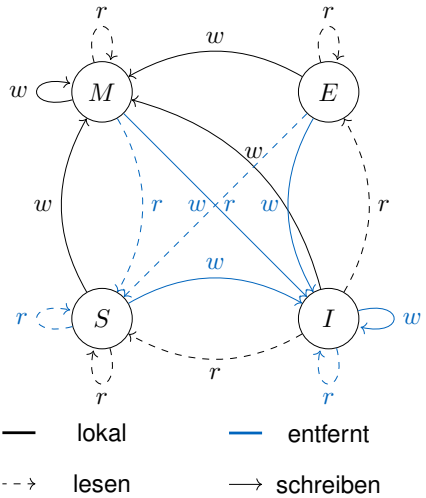
MSI/MESI

- Mehrkernsysteme: Was wenn CPU1 und CPU2 beide ein Datum gecached haben und es modifizieren?
  $\rightarrow$ Cache-Inkonsistenzen
- Einführung von Zuständen für Cachezeilen
- CPUs hören jeweils die Zugriffe der anderen Kerne ab („Bus Snooping“)
- **M**odified, (**E**xclusive), **S**hared, **I**nvalid
- Exclusive-Bit ermöglicht kleineren Overhead wenn CPUs auf verschiedenen Cache-Blöcken arbeiten



MSI/MESI

- Mehrkernsysteme: Was wenn CPU1 und CPU2 beide ein Datum gecached haben und es modifizieren?
  $\rightarrow$ Cache-Inkonsistenzen
- Einführung von Zuständen für Cachezeilen
- CPUs hören jeweils die Zugriffe der anderen Kerne ab („Bus Snooping“)
- **M**odified, (**E**xclusive), **S**hared, **I**nvalid
- Exclusive-Bit ermöglicht kleineren Overhead wenn CPUs auf verschiedenen Cache-Blöcken arbeiten



Speedup durch Parallelisierung

Mit t_s sequentieller Programmteil, t_p paralleler Programmteil, n Anzahl CPU-Kerne, T Ausführungszeit mit $n = 1$:

- Amdahlsches Gesetz: Gleiche Problemgröße, aufgeteilt auf mehrere Kerne $\rightarrow$ begrenzt durch sequentiellen Anteil

$$S_{\text{Amdahl}}(n) = \frac{T}{t_s + \frac{t_p}{n}}$$

- Gustafsons Gesetz: Mehr Kerne können mehr berechnen: Größeres Problem$\rightarrow$ paralleler Anteil wächst mit Problemgröße, t_s proportional kleiner

$$S_{\text{Gustafson}}(n) = \frac{t_s + n \cdot t_p}{T}$$

- Zwei verschiedene Perspektiven, abhängig von Problemszenario verschieden geeignet

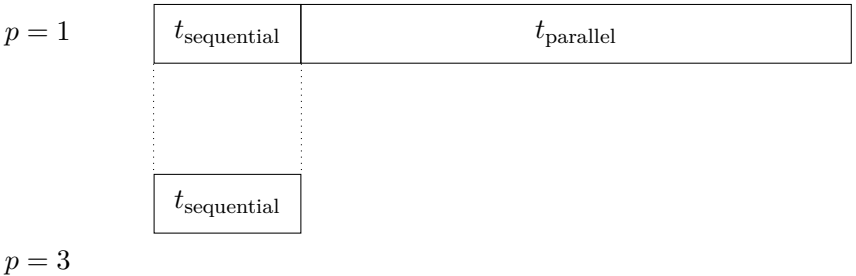
Amdahlsches Gesetz

$p = 1$

$p = 3$

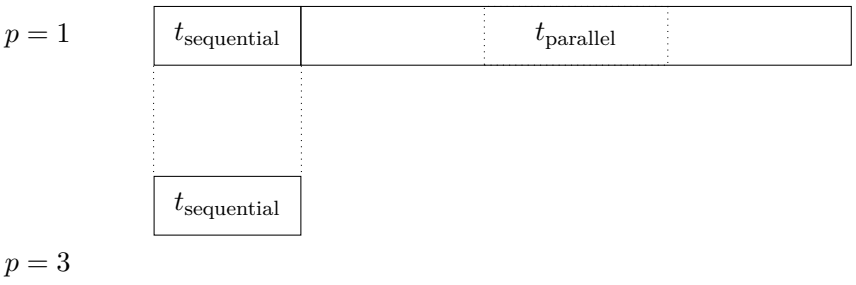
Quelle: Vorlesungsmaterial ERA

Amdahlsches Gesetz



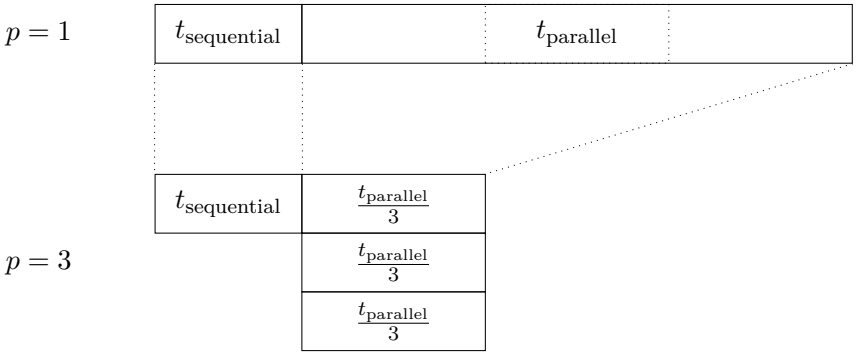
Quelle: Vorlesungsmaterial ERA

Amdahlsches Gesetz



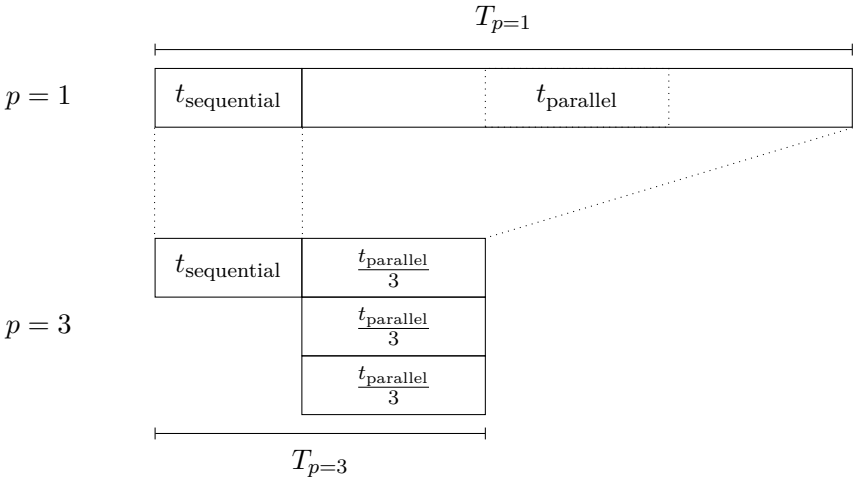
Quelle: Vorlesungsmaterial ERA

Amdahlsches Gesetz



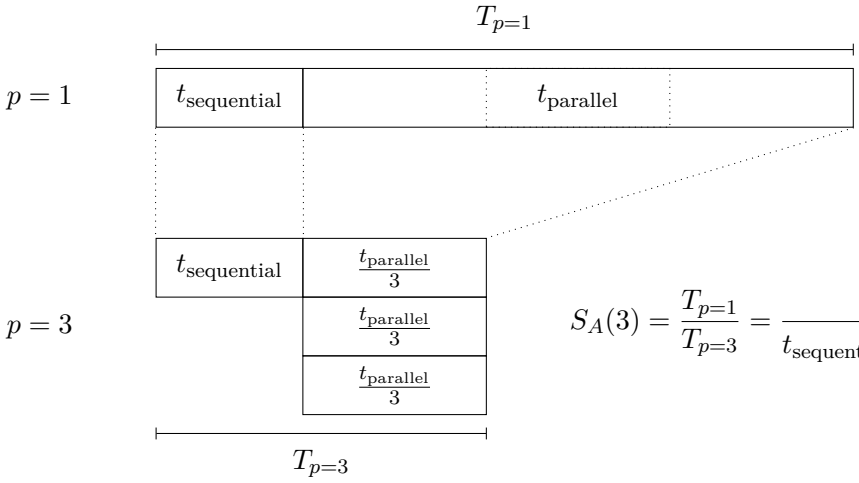
Quelle: Vorlesungsmaterial ERA

Amdahlsches Gesetz



Quelle: Vorlesungsmaterial ERA

Amdahlsches Gesetz



$$S_A(3) = \frac{T_{p=1}}{T_{p=3}} = \frac{T_{p=1}}{t_{\text{sequential}} + \frac{t_{\text{parallel}}}{3}}$$

Quelle: Vorlesungsmaterial ERA

Gustafsons Gesetz

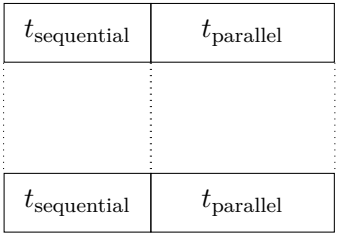
$p = 1$

$t_{\text{sequential}}$	t_{parallel}

$p = 3$

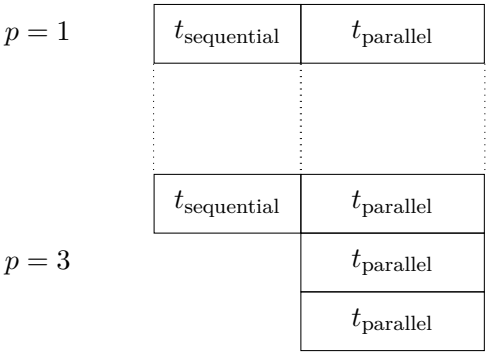
Gustafsons Gesetz

$p = 1$

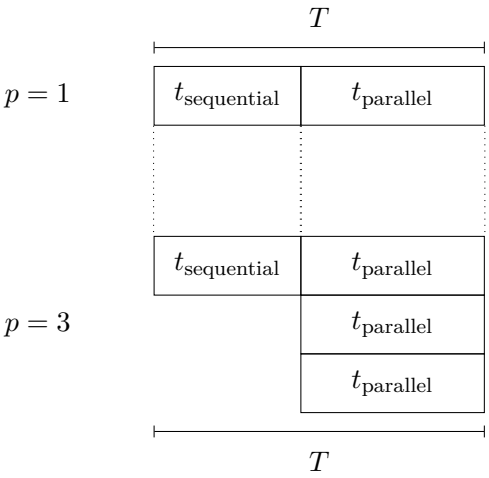


$p = 3$

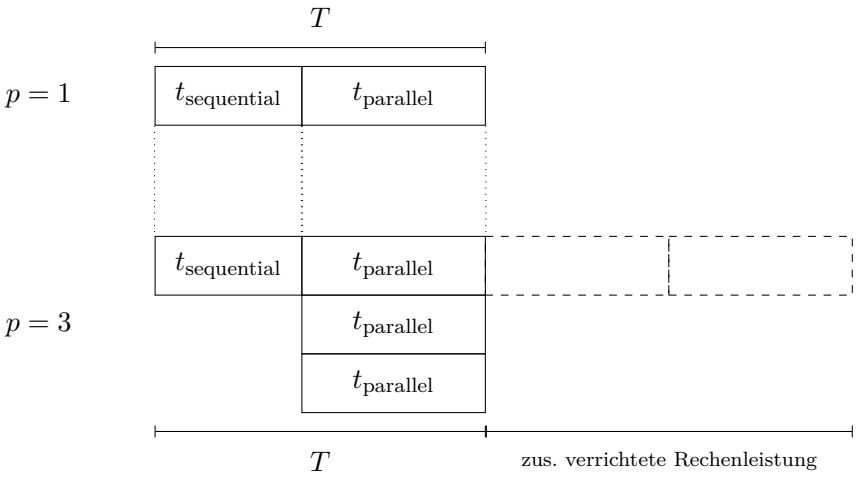
Gustafsons Gesetz



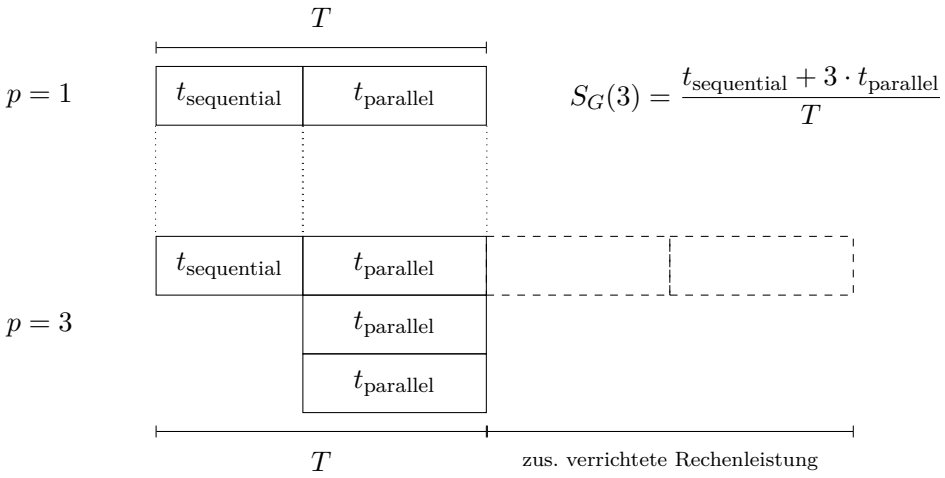
Gustafsons Gesetz



Gustafsons Gesetz



Gustafsons Gesetz



Artemis-Hausaufgaben

- „H10 — MESI“ bis 12.01.2025 23:59 Uhr
- Durchlaufen der MESI-Zustände für 4 parallel arbeitende CPUs

- „B01 — RFC 9402“ bis 26.01.2025 23:59 Uhr
- Wiederholungsaufgabe RISC-V Assembly: Strings zusammenkopieren
- Erste (und vsl. letzte) Bonusaufgabe: 10 Punkte Bonuspunkte!

Links

- Zulip: „ERA Tutorium - Do-1600-1“ bzw. „ERA Tutorium - Fr-1500-2“
- ERA-Moodle-Kurs
- ERA-Artemis-Kurs
- Wikipedia zu MESI
- Amdahlsches und Gustafsons Gesetz

School of Computation, Information and Technology
Technische Universität München

Übung 10: Parallelisierung

Einführung in die Rechnerarchitektur

Niklas Ladurner

School of Computation, Information and Technology
Technische Universität München

3. Januar 2025