Transformationen zur parallelisierten und fehlertoleranten Informationsverarbeitung

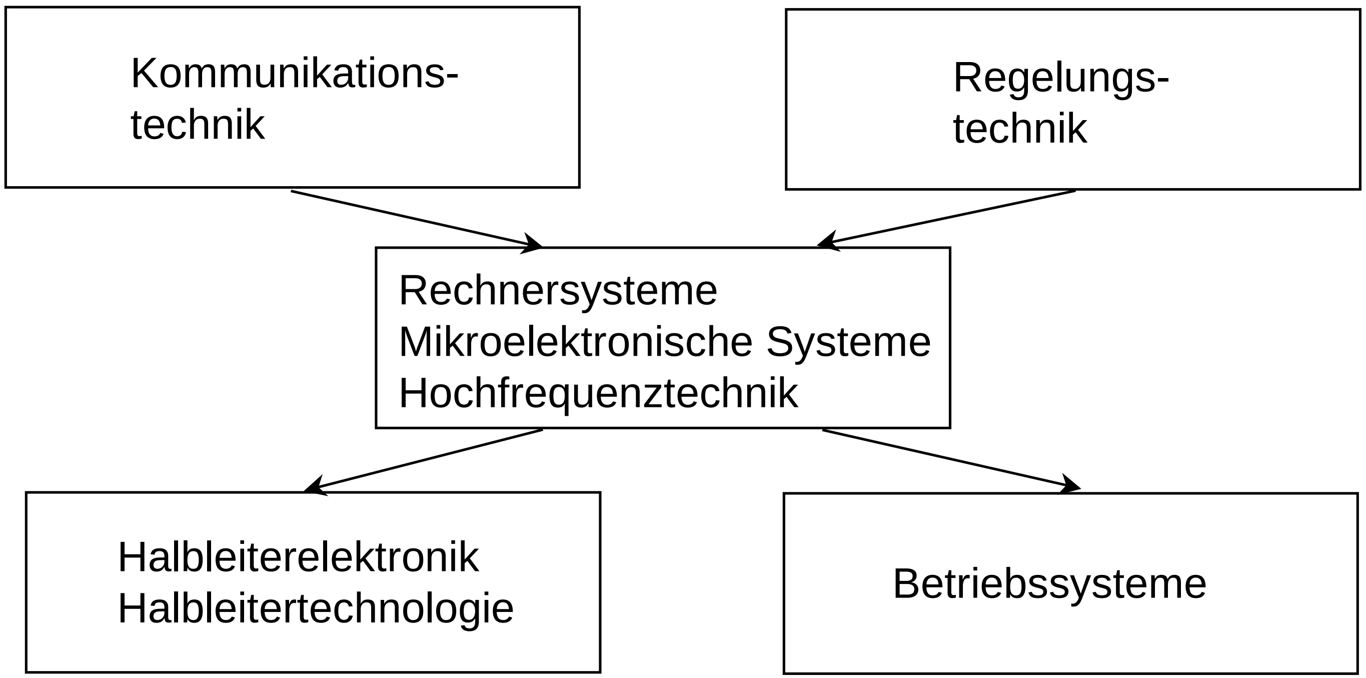
Felix Gärtner, Marc Theisen
Graduiertenkolleg ISIA
Technische Universität Darmstadt
Karlstr. 15
64283 Darmstadt

felix@informatik.tu-darmstadt.de
theisen@mes.tu-darmstadt.de

GK ISIA

TECHNISCHE UNIVERSITÄT DARMSTADT

- Graduiertenkolleg “Intelligente Systeme für die Informations- und Automatisierungstechnik”
- beteiligte Fachbereiche: “Elektrotechnik und Informationstechnik” und “Informatik”



- Ziel: Methoden und Mechanismen zur verbesserten Entwicklung von intelligenten Systemen

Gliederung

TECHNISCHE
UNIVERSITÄT
DARMSTADT

1. Gemeinsames Grundkonzept: Transformationen
2. Parallelisierende Transformation zur Effizienzsteigerung
3. Transformation zur Verbesserung der Fehlertoleranz
4. Zusammenfassung

durchgehendes Beispiel: komplexe Berechnung in der Signalverarbeitung

1. Gemeinsames Grundkonzept

•Ausgangspunkt: Verhaltensbeschreibung eines Systems

•Anforderungen:
- geringe Chipfläche, verlustleistungsarm, hoher Datendurchsatz (Kommunikationstechnik)
- fehlertolerant (Flugsicherheit)

• interaktive / automatische Optimierung beim Entwurf

Spezifikation

1. Gemeinsames Grundkonzept

•Ausgangspunkt: Verhaltensbeschreibung eines Systems

•Anforderungen:
- geringe Chipfläche, verlustleistungsarm, hoher Datendurchsatz (Kommunikationstechnik)
- fehlertolerant (Flugsicherheit)

• interaktive / automatische Optimierung beim Entwurf

$$\begin{array}{c} \boxed{\text{Spezifikation}} \\ \Big\downarrow \text{ Transformation} \\ \boxed{\text{verbesserte Spezifikation}} \end{array}$$

1. Gemeinsames Grundkonzept

- Ausgangspunkt: Verhaltensbeschreibung eines Systems

- Anforderungen:
  - geringe Chipfläche, verlustleistungsarm, hoher Datendurchsatz (Kommunikationstechnik)
  - fehlertolerant (Flugsicherheit)

- interaktive / automatische Optimierung beim Entwurf

Spezifikation

↓ Transformation

verbesserte Spezifikation

↓ Realisierung

ausführbares System

2. Parallelisierung geschachtelter Schleifen

TECHNISCHE UNIVERSITÄT DARMSTADT

• Beispiel:

```
L1: for I1 in 0 to 3 loop
  L2: for I2 in 0 to 3 loop
    H(I1,I2): A(I1,I2) = f(A(I1-1,I2), A(I1,I2-1));
  end loop;
end loop;
```

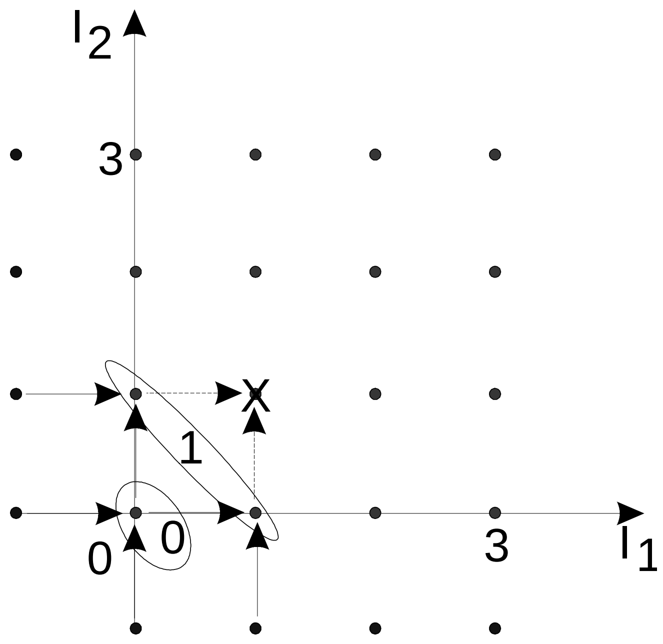
• Ziel:

- •optimaler Kompromiß zwischen Zeit und Fläche

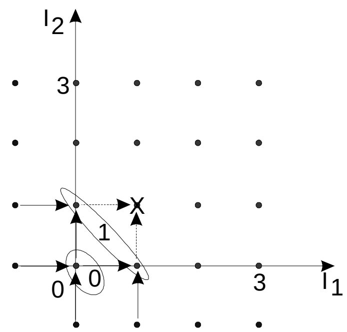
•Lösungsansatz:

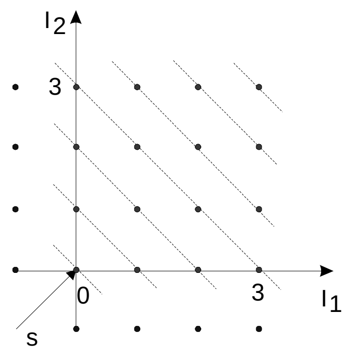
- •aus Compilertechnik bekannte Methode: Parallelisierung

Wellenfronten

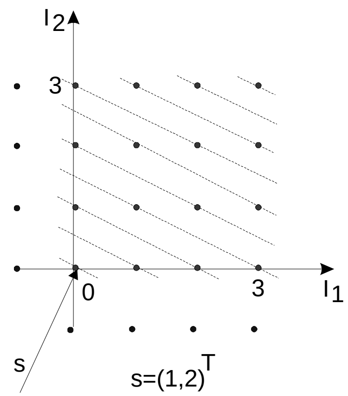


Wellenfronten

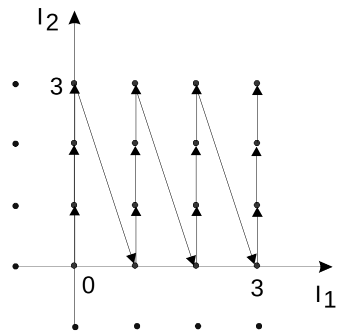




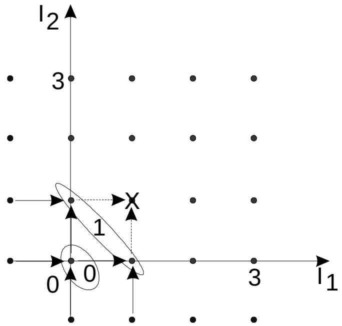
$s=(1,1)^T$

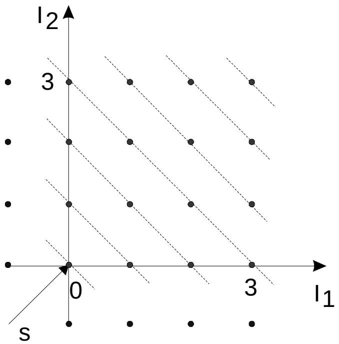


$s=(1,2)^T$

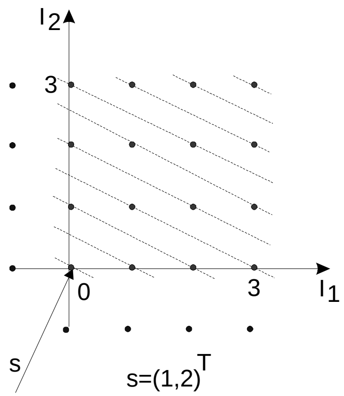


Wellenfronten

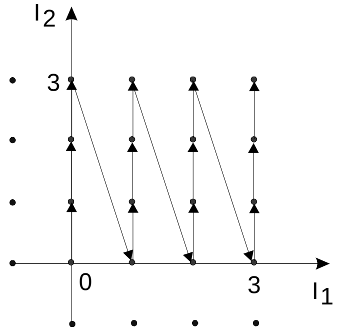




s=(1,1)T

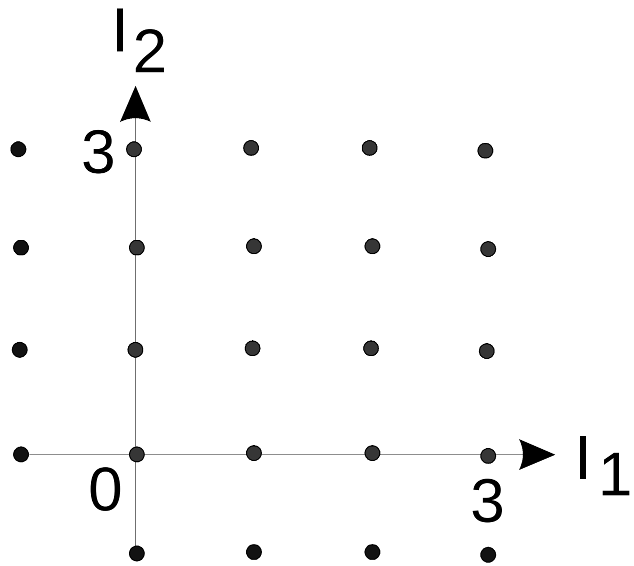


s=(1,2)T

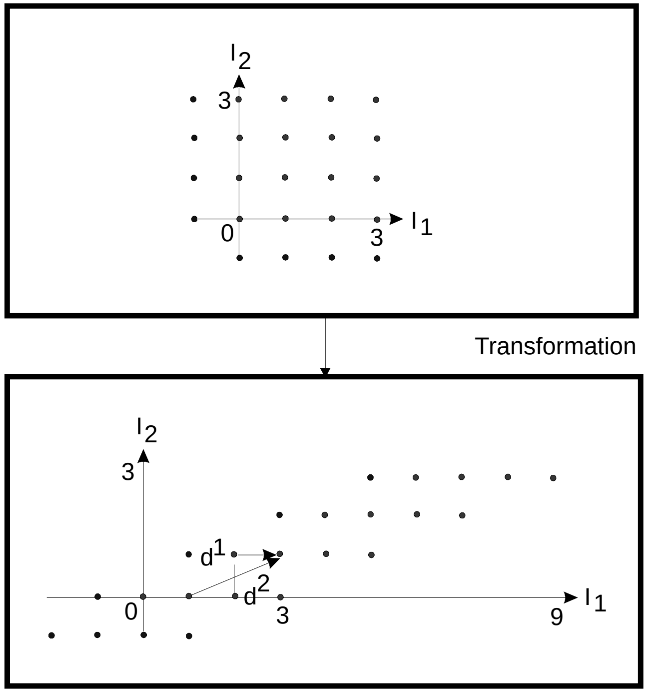


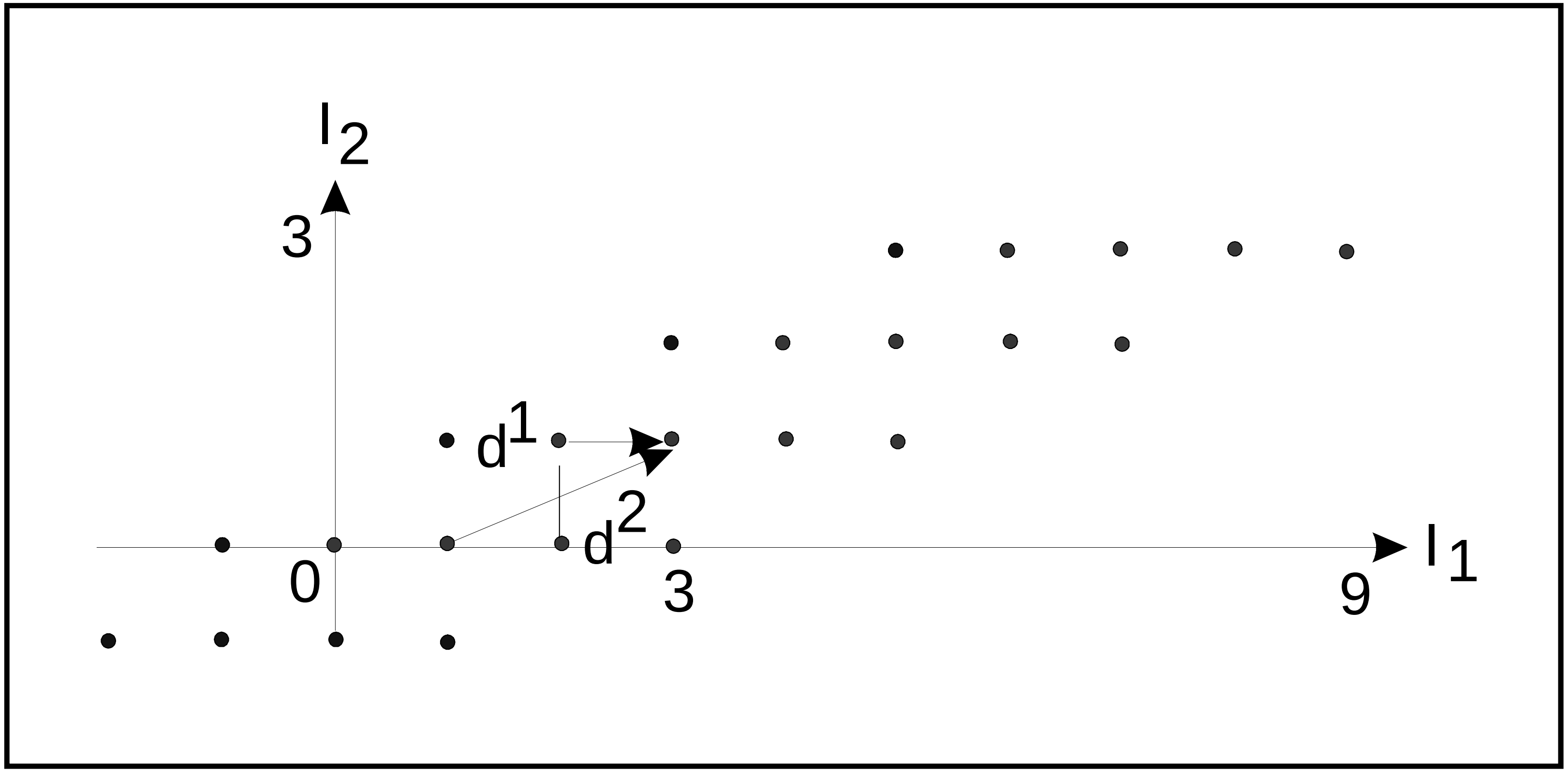
Parallelitätsgrad	Iterationen
1	16
2	10
4	7

Optimierung und Realisierung



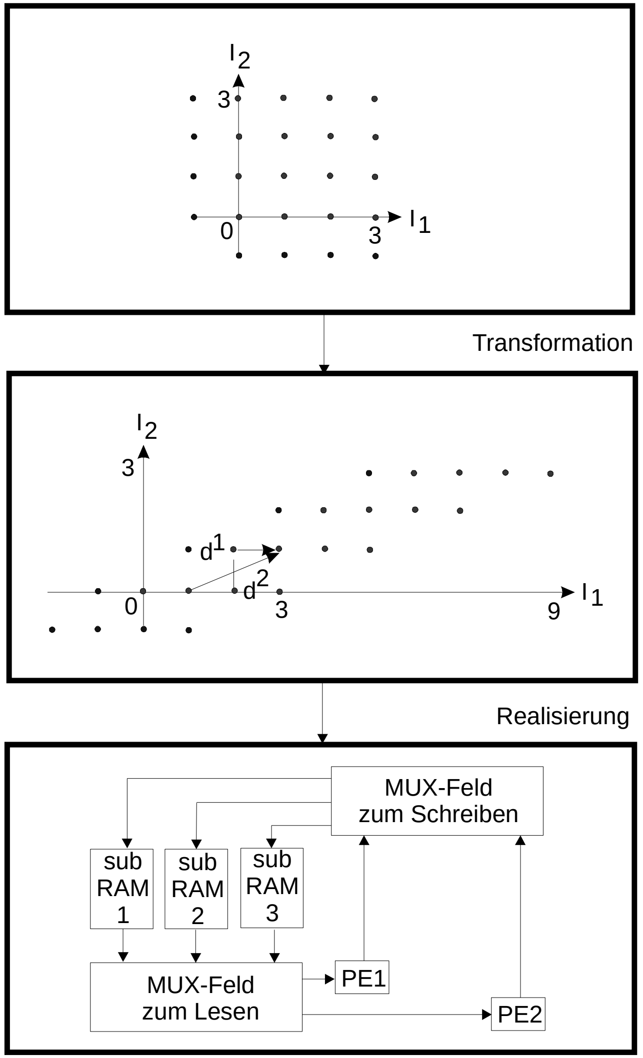
Optimierung und Realisierung



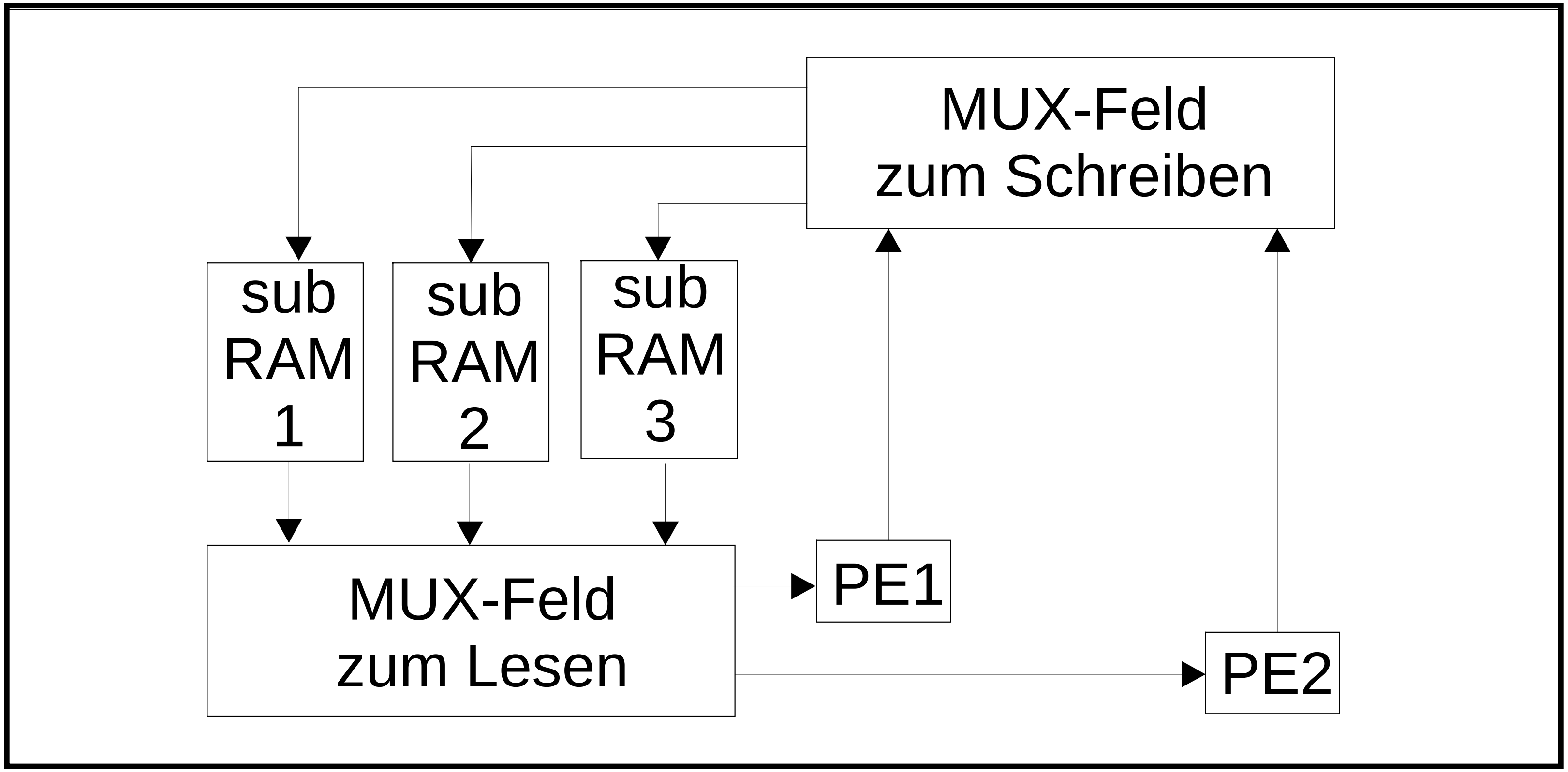


l_2
3
d^1
d^2
0
3
9
l_1

Optimierung und Realisierung



Optimierung und Realisierung



Optimierung und Realisierung

S1 ← Originale Schleifenspezifikation

↓ Effizienztransformation

S2 ← Parallelisierte Schleife

↓ Realisierung

R2 ← Effiziente Hardware

3. Fehlertoleranz-Transformationen

- Motivation: Verbesserung der Zuverlässigkeit

- Ziel jetzt: optimierte Schaltung soll gegen den Ausfall einzelner Komponenten fehlertolerant gemacht werden

- Fehlertoleranz = Korrektheit unter Fehlereinflüssen

- Korrektheit = System erfüllt Spezifikation

Konstruktionsprozeß

1. Festlegung der Fehlerannahme: genau welche Fehler können auftreten

    Kunst: Fehlerannahme so stark wie nötig und so schwach wie möglich

2. Entwurf und Realisierung eines Fehlertoleranzverfahrens

3. Validierung der Fehlertoleranzeigenschaften

Konstruktionsprozeß

TECHNISCHE UNIVERSITÄT DARMSTADT

1. Festlegung der Fehlerannahme: genau welche Fehler können auftreten

   Kunst: Fehlerannahme so stark wie nötig und so schwach wie möglich

2. ausgehend von Fehlerannahme eine verifizierte Systemtransformation durchführen, die Fehlertoleranz einbaut

Fehlertoleranztheoreme

- Formale Grundlagen der Fehlertoleranz bisher wenig untersucht

- sind aber Schlüssel für effizientere und verifizierte Fehlertoleranzverfahren

Theoreme:

- Redundanz notwendig für Fehlertoleranz

    - Platzredundanz
    - Zeitredundanz

- Fehlertoleranzmechanismus = Fehlererkennung + Fehlerkorrektur

Fortgeführtes Beispiel

S1 ← Originale Schleifenspezifikation

Fortgeführtes Beispiel

$$
\begin{tikzcd}
\text{S1} & \text{Originale Schleifenspezifikation} \arrow[l] \\
\text{S2} \arrow[from=u, "\text{Effizienztransformation}"'] & \text{Parallelisierte Schleife} \arrow[l]
\end{tikzcd}
$$

Fortgeführtes Beispiel

S1 ← Originale Schleifenspezifikation

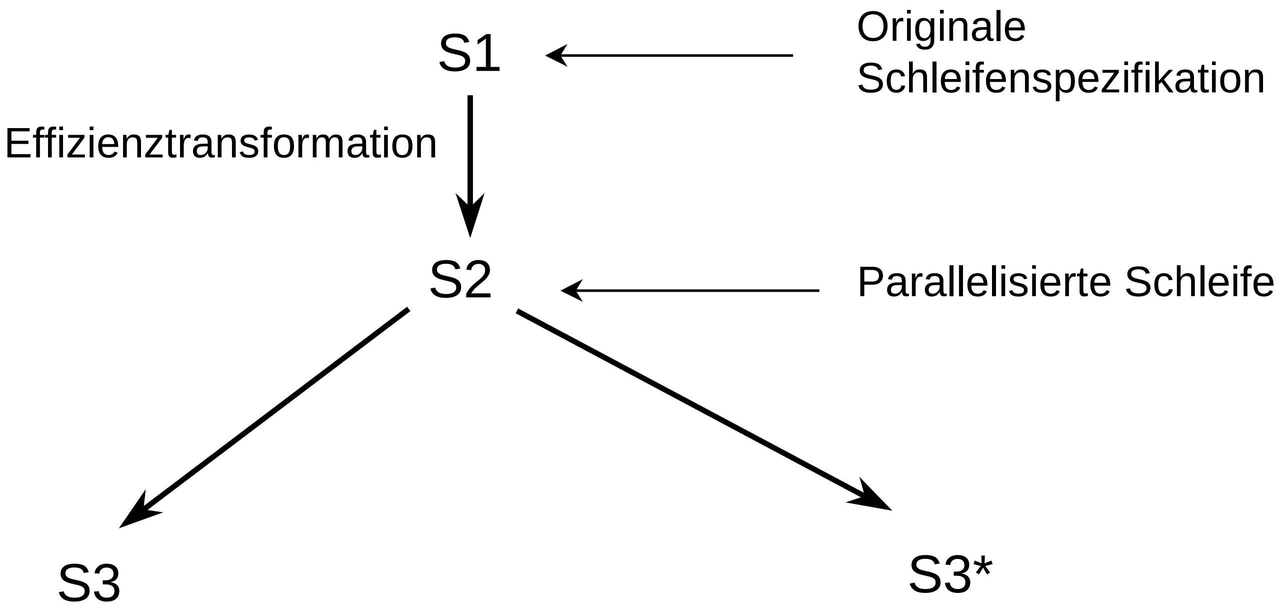
Effizienztransformation

S2 ← Parallelisierte Schleife

S3

- zusätzliches (redundantes) PE
- größerer Flächenverbrauch (für PE und Rekonfigurationslogik)
- unveränderte Ausführungszeit

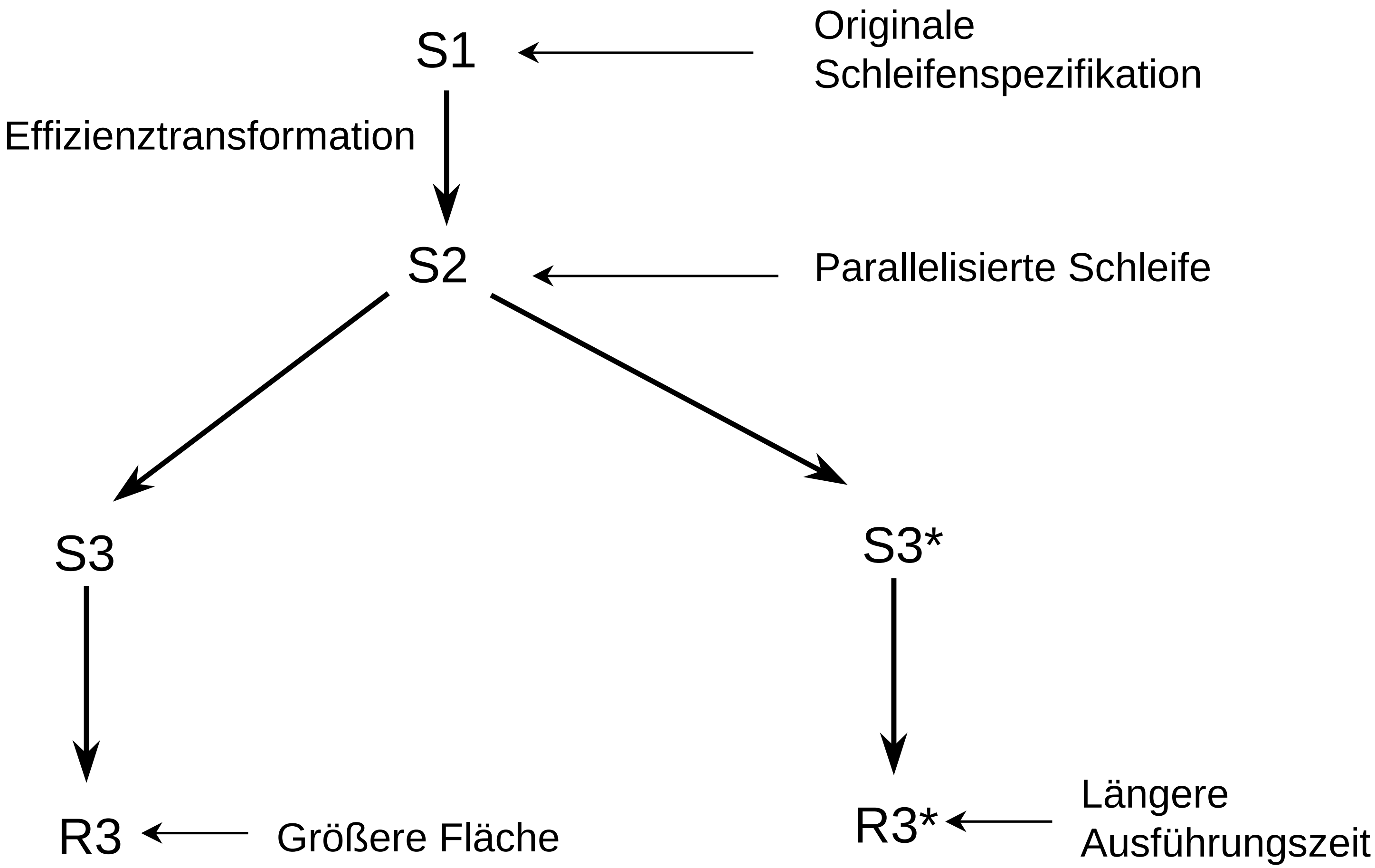
Fortgeführtes Beispiel



- zusätzliches (redundantes) PE
- größerer Flächenverbrauch (für PE und Rekonfigurationslogik)
- unveränderte Ausführungszeit

- PEs sind bereits „redundant“
- (fast) unveränderte Fläche
- ggf. längere Ausführungszeit

Fortgeführtes Beispiel



Zusammenfassung

- Transformation als Konzept zur automatischen / interaktiven Optimierung von Systemeigenschaften bei der Entwicklung

- Beispielhaft gezeigt:
  1. Effizienz
  2. Fehlertoleranz

- Probleme: Transformationen ändern möglicherweise bereits im System vorhandene (gewünschte) Systemeigenschaften ab

- weitere Arbeit: Kompositionen von Transformationen untersuchen