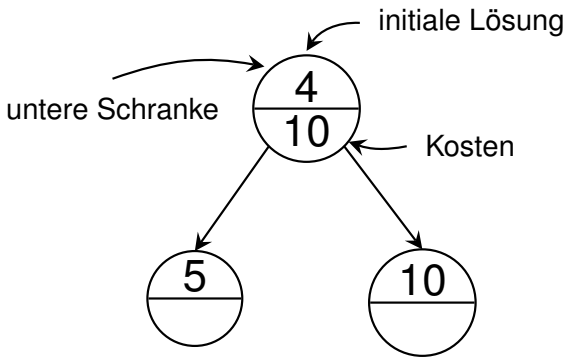
Optimale Suche mittels Branch&Bound

☞ Planungsproblem $\mapsto$ Suchproblem in einem Suchbaum

- Potentiell Betrachtung des kompletten Suchraums $\rightsquigarrow$ exp. Laufzeit

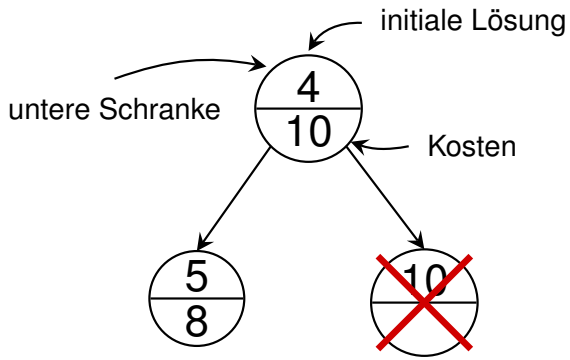


1. Berechne eine initiale Lösung
   - → Ein (evtl. unzulässiger) Ablaufplan
   - Bestimmung der tatsächlichen Kosten
   - Bestimmung einer unteren Schranke
2. Leite verbesserte initiale Lösungen ab
   - → Branch-Schritt

Optimale Suche mittels Branch&Bound

☞ Planungsproblem $\mapsto$ Suchproblem in einem Suchbaum

- Potentiell Betrachtung des kompletten Suchraums $\rightsquigarrow$ exp. Laufzeit

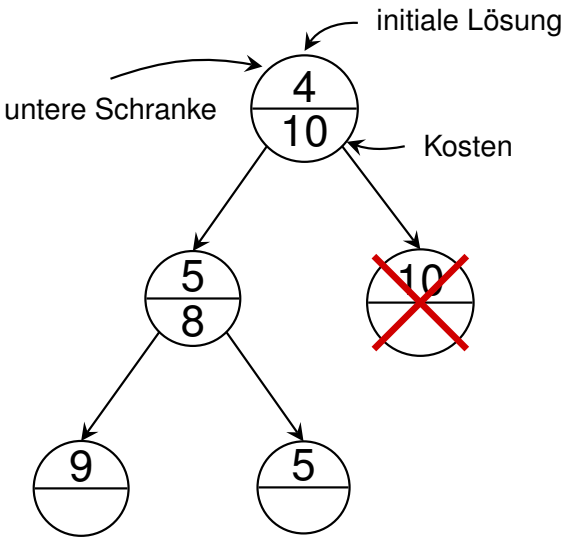


1. Berechne eine initiale Lösung
   - → Ein (evtl. unzulässiger) Ablaufplan
   - Bestimmung der tatsächlichen Kosten
   - Bestimmung einer unteren Schranke
2. Leite verbesserte initiale Lösungen ab
   - → Branch-Schritt
   - Verwerfen ungeeigneter Lösungen
   - → Reduktion des Suchraums: Bound-Schritt

Optimale Suche mittels Branch&Bound

☞ Planungsproblem $\mapsto$ Suchproblem in einem Suchbaum

- Potentiell Betrachtung des kompletten Suchraums $\rightsquigarrow$ exp. Laufzeit

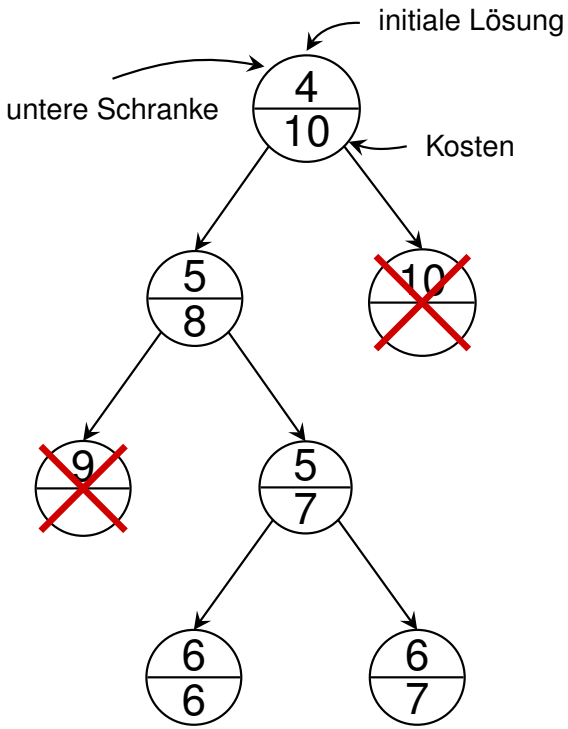


1. Berechne eine initiale Lösung
   - → Ein (evtl. unzulässiger) Ablaufplan
   - Bestimmung der tatsächlichen Kosten
   - Bestimmung einer unteren Schranke
2. Leite verbesserte initiale Lösungen ab
   - → Branch-Schritt
   - Verwerfen ungeeigneter Lösungen
   - → Reduktion des Suchraums: Bound-Schritt
3. Wiederhole diese Schritte . . .

Optimale Suche mittels Branch&Bound

☞ Planungsproblem $\mapsto$ Suchproblem in einem Suchbaum

- Potentiell Betrachtung des kompletten Suchraums $\rightsquigarrow$ exp. Laufzeit

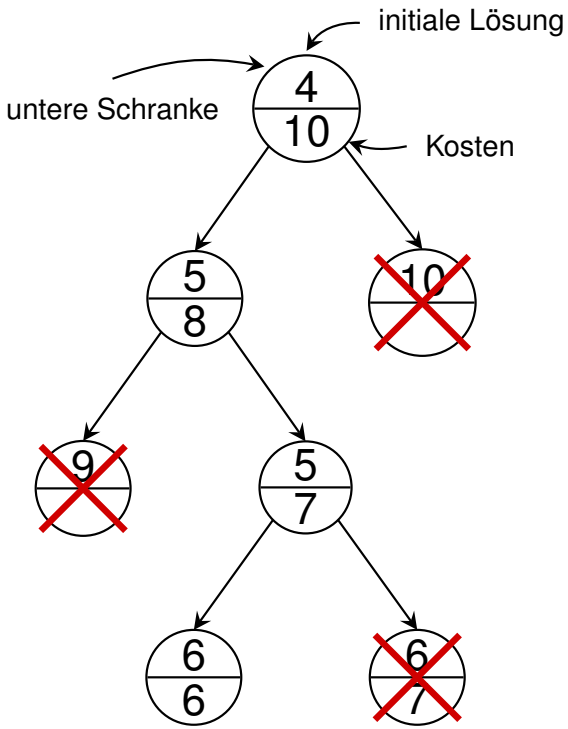


1. Berechne eine initiale Lösung
   - → Ein (evtl. unzulässiger) Ablaufplan
   - Bestimmung der tatsächlichen Kosten
   - Bestimmung einer unteren Schranke
2. Leite verbesserte initiale Lösungen ab
   - → Branch-Schritt
   - Verwerfen ungeeigneter Lösungen
   - → Reduktion des Suchraums: Bound-Schritt
3. Wiederhole diese Schritte . . .

Optimale Suche mittels Branch&Bound

☞ Planungsproblem ↦ Suchproblem in einem Suchbaum

- Potentiell Betrachtung des kompletten Suchraums ⇝ exp. Laufzeit



1. Berechne eine initiale Lösung
   - → Ein (evtl. unzulässiger) Ablaufplan
   - Bestimmung der tatsächlichen Kosten
   - Bestimmung einer unteren Schranke
2. Leite verbesserte initiale Lösungen ab
   - → Branch-Schritt
   - Verwerfen ungeeigneter Lösungen
   - → Reduktion des Suchraums: Bound-Schritt
3. Wiederhole diese Schritte . . .
   bis zur optimalen Lösung
   - oder klar ist, dass keine Lösung existiert

⚠ Um Optimalität zu erreichen, müssen im Branch-Schritt alle Möglichkeiten ausgeschöpft werden, eine Lösung zu verbessern

Branch&Bound – Statische Ablaufplanung

Wo kommen die initiale Lösung, die Kosten und die untere Schranke her?

- Initiale Lösungen sind bereits vollständige gültige Ablaufpläne
  - Diese können aber noch Termine verletzen, sind also nicht zulässig

  → Ein Verfahren für deren Bestimmung wird benötigt
- Kosten einer Lösung sind die vorhandenen Verspätungen

  → Maximale Terminüberschreitung aller Jobs
- Untere Schranken durch Vereinfachung des Planungsproblems
  - Ohne die Optimalität des Algorithmus zu verletzten
- Verbesserung durch Manipulation des Planungsproblems
  - Arbeitsauftrag mit der größten Terminüberschreitung früher einplanen

  → Verspätung ohne Verletzung der ursprünglichen Vorgaben reduzieren

Branch&Bound – Statische Ablaufplanung

Wo kommen die initiale Lösung, die Kosten und die untere Schranke her?

- Initiale Lösungen sind bereits vollständige gültige Ablaufpläne
  - Diese können aber noch Termine verletzen, sind also nicht zulässig

  → Ein Verfahren für deren Bestimmung wird benötigt
- Kosten einer Lösung sind die vorhandenen Verspätungen

  → Maximale Terminüberschreitung aller Jobs
- Untere Schranken durch Vereinfachung des Planungsproblems
  - Ohne die Optimalität des Algorithmus zu verletzten
- Verbesserung durch Manipulation des Planungsproblems
  - Arbeitsauftrag mit der größten Terminüberschreitung früher einplanen

  → Verspätung ohne Verletzung der ursprünglichen Vorgaben reduzieren

Ziel: Eine Lösung finden, deren Kosten kleiner oder gleich 0 sind. Der zugehörige Ablaufplan ist daher zulässig.

Der Algorithmus von Abdelzaher und Shin [1]

Ein Beispiel für einen Branch&Bound-Algorithmus für die statische Ablaufplanung

☞ Leistungsumfang des Algorithmus

- Auslösezeiten, Ausführungszeiten, Termine
- Gerichtete und ungerichtete Abhängigkeiten
- Einkern-, Mehrkern- und Mehrprozessorsysteme
- Verteilte Systeme und nachrichtenbasierte Kommunikation

⚠ Der Algorithmus führt jedoch keine Allokation durch!

Der Algorithmus von Abdelzaher und Shin [1]

Ein Beispiel für einen Branch&Bound-Algorithmus für die statische Ablaufplanung

☞ Leistungsumfang des Algorithmus

- Auslösezeiten, Ausführungszeiten, Termine
- Gerichtete und ungerichtete Abhängigkeiten
- Einkern-, Mehrkern- und Mehrprozessorsysteme
- Verteilte Systeme und nachrichtenbasierte Kommunikation

⚠ Der Algorithmus führt jedoch keine Allokation durch!

Initiale Lösung: Globaler EDF-Algorithmus (G-EDF)

- Erweitert um die Behandlung ungerichteter Abhängigkeiten

→ Für obiges Planungsproblem nicht optimal

Kosten: Ablaufplan mittels EDF bestimmen

Untere Schranke : Vereinfachung bis EDF optimal ist!

→ Entfernen ungerichteter oder kernübergreifender Abhängigkeiten

Verbesserung: durch gezieltes Hinzufügen von Abhängigkeiten

Gliederung

1 Entwicklung – Herangehensweise
- Ablaufplanung – Bottom-Up
- Spezifikation – Top-Down

2 Manuelle Einplanung
- Struktur zyklischer Ablaufpläne

3 Algorithmische Einplanung
- List-Scheduling-Verfahren
- Branch&Bound-Algorithmen

4 Moduswechsel

5 Zusammenfassung

Moduswechsel: Flexibilität in zeitgesteuerten Systemen

⚠ Eine für alle Lastsituationen passende statische Ablauftabellen

- In der Praxis nur schwer zu realisieren
- Negativbeispiel: Wartung von Steuergeräten im Auto
  - Diagnosedaten werden in normales Kommunikationsverhalten eingebettet

  → Niedrige Nutzlast (engl. *payload*) ist die Folge

⚠ Statische Ablauftabellen orientieren sich am schlimmsten Fall

- Arbeitsaufträge belegen immer die zugewiesene WCET
- Auch wenn sie zwar periodisch, aber nur selten ausgelöst werden

Moduswechsel: Flexibilität in zeitgesteuerten Systemen

⚠ Eine für alle Lastsituationen passende statische Ablauftabellen

- In der Praxis nur schwer zu realisieren
- Negativbeispiel: Wartung von Steuergeräten im Auto
  - Diagnosedaten werden in normales Kommunikationsverhalten eingebettet

  → Niedrige Nutzlast (engl. *payload*) ist die Folge

⚠ Statische Ablauftabellen orientieren sich am schlimmsten Fall

  - Arbeitsaufträge belegen immer die zugewiesene WCET
  - Auch wenn sie zwar periodisch, aber nur selten ausgelöst werden

- Entflechtung der Arbeitsaufträge ist das Ziel
  - Arbeitsaufträge befinden sich nur in einer gemeinsamen Ablauftabelle, wenn sie auch zusammen ausgelöst werden können

Moduswechsel: Flexibilität in zeitgesteuerten Systemen

⚠ Eine für alle Lastsituationen passende statische Ablauftabellen

- In der Praxis nur schwer zu realisieren
- Negativbeispiel: Wartung von Steuergeräten im Auto
  - Diagnosedaten werden in normales Kommunikationsverhalten eingebettet

  → Niedrige Nutzlast (engl. *payload*) ist die Folge

⚠ Statische Ablauftabellen orientieren sich am schlimmsten Fall

- Arbeitsaufträge belegen immer die zugewiesene WCET
- Auch wenn sie zwar periodisch, aber nur selten ausgelöst werden

■ Entflechtung der Arbeitsaufträge ist das Ziel

- Arbeitsaufträge befinden sich nur in einer gemeinsamen Ablauftabelle, wenn sie auch zusammen ausgelöst werden können

☞ Gruppierungen von Arbeitsaufträgen definieren Betriebszustände

- Repräsentiert durch eine eigene statische Ablauftabelle

→ Wechsel des Betriebszustands impliziert auch ihren Wechsel

Moduswechsel: Flexibilität in zeitgesteuerten Systemen

⚠ Eine für alle Lastsituationen passende statische Ablauftabellen

- In der Praxis nur schwer zu realisieren
- Negativbeispiel: Wartung von Steuergeräten im Auto
  - Diagnosedaten werden in normales Kommunikationsverhalten eingebettet

  → Niedrige Nutzlast (engl. *payload*) ist die Folge

⚠ Statische Ablauftabellen orientieren sich am schlimmsten Fall

- Arbeitsaufträge belegen immer die zugewiesene WCET
- Auch wenn sie zwar periodisch, aber nur selten ausgelöst werden

■ Entflechtung der Arbeitsaufträge ist das Ziel

- Arbeitsaufträge befinden sich nur in einer gemeinsamen Ablauftabelle, wenn sie auch zusammen ausgelöst werden können

☞ Gruppierungen von Arbeitsaufträgen definieren Betriebszustände

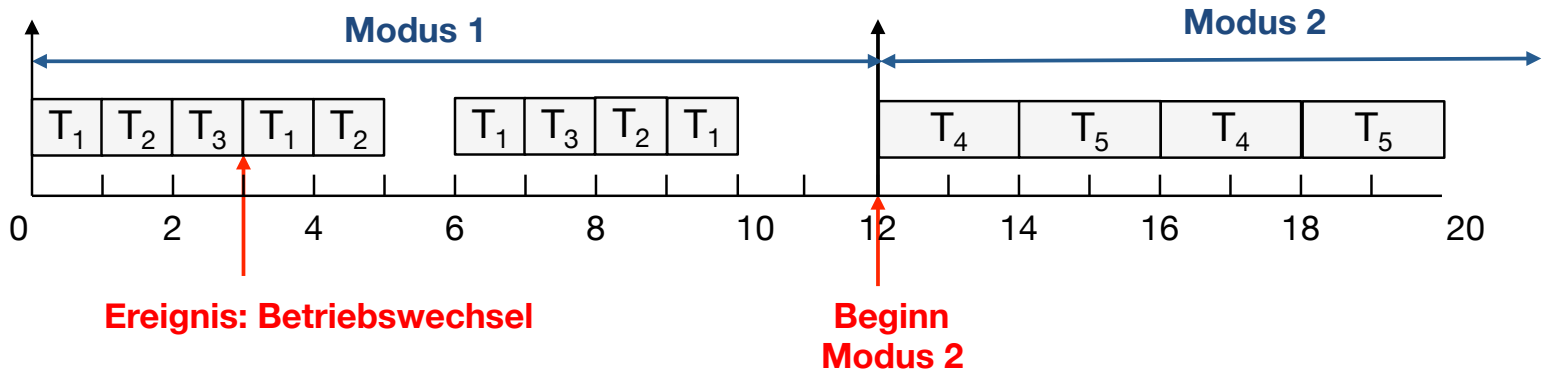
- Repräsentiert durch eine eigene statische Ablauftabelle

→ Wechsel des Betriebszustands impliziert auch ihren Wechsel

⚠ Betriebswechsel erfordert ein systemweit koordiniertes Vorgehen

Rekonfiguration des Aufgabensystems

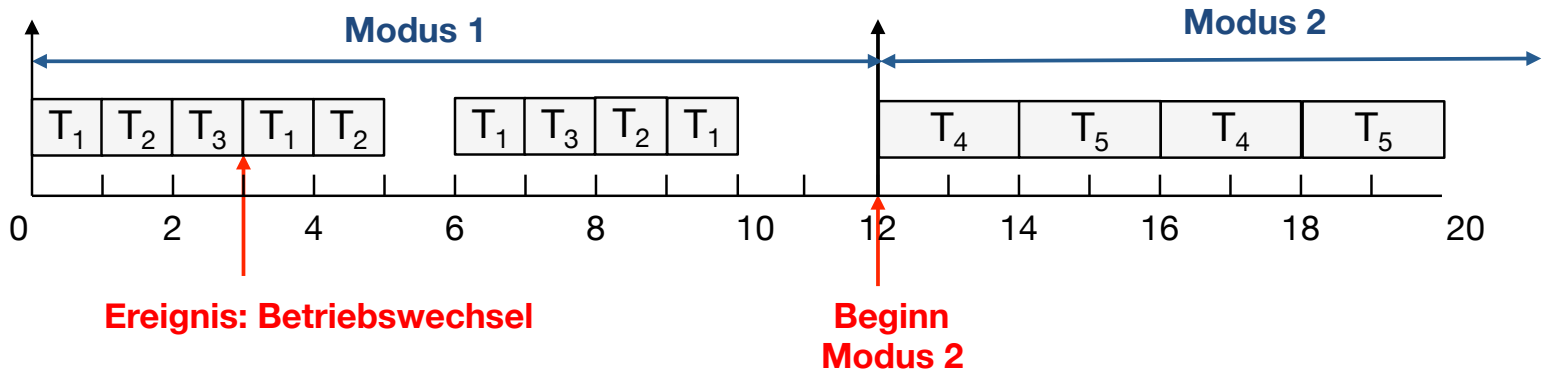
Änderung von Aufgabenanzahl und -parametern



⚠ Umstellen auf einen neuen statischen Ablaufplan bedeutet mehr als nur einen Tabellenwechsel zu vollziehen:

Rekonfiguration des Aufgabensystems

Änderung von Aufgabenanzahl und -parametern



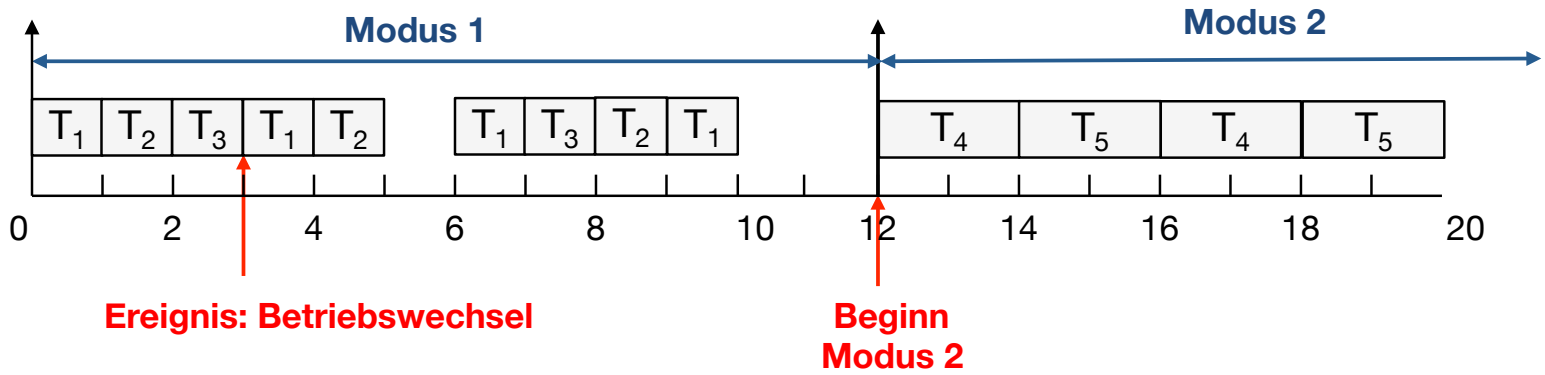
⚠ Umstellen auf einen neuen statischen Ablaufplan bedeutet mehr als nur einen Tabellenwechsel zu vollziehen:

1 Zerstörung und Erzeugung von periodischen Aufgaben

- Einige periodische Aufgaben sind nicht mehr erforderlich
  ⇝ Betriebsmittelfreigabe
- Andere Aufgaben müssen dem System neu hinzugefügt werden
  ⇝ Betriebsmittelanforderung
- Manche Aufgaben überdaueren den Betriebswechsel
  ⇝ Parametererhaltung

Rekonfiguration des Aufgabensystems

Änderung von Aufgabenanzahl und -parametern



⚠ Umstellen auf einen neuen statischen Ablaufplan bedeutet mehr als nur einen Tabellenwechsel zu vollziehen:

1. Zerstörung und Erzeugung von periodischen Aufgaben
   - Einige periodische Aufgaben sind nicht mehr erforderlich
     ⇝ Betriebsmittelfreigabe
   - Andere Aufgaben müssen dem System neu hinzugefügt werden
     ⇝ Betriebsmittelanforderung
   - Manche Aufgaben überdaueren den Betriebswechsel
     ⇝ Parametererhaltung
2. Einlagerung und Aktivierung der neuen Ablauftabelle
   - Taskparameter und neuer Ablaufplan wurden *à priori* bestimmt

Arten von Betriebswechsel

☞ Wechsel vom speziellen Arbeitsauftrag (engl. *mode-change job*) durchführen lassen ↦ nicht-periodische Aufgabe

- Antwortzeit des Betriebswechsels minimieren (Hyperperiode!)
- Verbunden mit einem weichen oder harten Termin

Arten von Betriebswechsel

☞ Wechsel vom speziellen Arbeitsauftrag (engl. *mode-change job*) durchführen lassen ↦ nicht-periodische Aufgabe
- Antwortzeit des Betriebswechsels minimieren (Hyperperiode!)
- Verbunden mit einem weichen oder harten Termin

- Aperiodisch ↦ Betriebswechsel mit weichem Termin
  - Mit höchster Dringlichkeit ausgeführt als aperiodischer Auftrag
    → Kommt vor allen anderen aperiodischen Aufträgen zum Zuge
  - Zerstörung aperiodischer/sporadischer Jobs ist problematisch
    - Ausführung aperiodischer Jobs wird hinausgezögert, bis der Betriebswechsel vollendet worden ist
    - Im Falle sporadischer Jobs stehen zwei Optionen zur Verfügung:
      (a) Betriebswechsel wird unterbrochen und später fortgesetzt
      (b) Übernahmeprüfung berücksichtigt den neuen Ablaufplan
  - Ziel ist Minimierung der Antwortzeit für den Betriebswechsel

Arten von Betriebswechsel

☞ Wechsel vom speziellen Arbeitsauftrag (engl. *mode-change job*) durchführen lassen ↦ nicht-periodische Aufgabe

- Antwortzeit des Betriebswechsels minimieren (Hyperperiode!)
- Verbunden mit einem weichen oder harten Termin

- Aperiodisch ↦ Betriebswechsel mit weichem Termin
  - Mit höchster Dringlichkeit ausgeführt als aperiodischer Auftrag
    → Kommt vor allen anderen aperiodischen Aufträgen zum Zuge
  - Zerstörung aperiodischer/sporadischer Jobs ist problematisch
    - Ausführung aperiodischer Jobs wird hinausgezögert, bis der Betriebswechsel vollendet worden ist
    - Im Falle sporadischer Jobs stehen zwei Optionen zur Verfügung:
      (a) Betriebswechsel wird unterbrochen und später fortgesetzt
      (b) Übernahmeprüfung berücksichtigt den neuen Ablaufplan
  - Ziel ist Minimierung der Antwortzeit für den Betriebswechsel
- Sporadisch ↦ Betriebswechsel mit hartem Termin
  - Anwendung muss die evtl. Abweisung des Wechsels behandeln
    - Betriebswechsel muss ggf. hinausgezögert werden

Gliederung

1 Entwicklung – Herangehensweise
- Ablaufplanung – Bottom-Up
- Spezifikation – Top-Down

2 Manuelle Einplanung
- Struktur zyklischer Ablaufpläne

3 Algorithmische Einplanung
- List-Scheduling-Verfahren
- Branch&Bound-Algorithmen

4 Moduswechsel

5 Zusammenfassung

Resümee

- Entwicklungsprozesse führen verschiedenste Akteure zusammen
  - Firmen/Arbeitsgruppen sind u.U. über den ganzen Globus verstreut

  → Eine zeitliche Spezifikation der Abläufe ist wünschenswert
    - Sie ermöglicht die Entwicklung top-down zu strukturieren
    - Wird durch eine manuelle, statische Ablaufplanung unterstützt

Resümee

- Entwicklungsprozesse führen verschiedenste Akteure zusammen
  - Firmen/Arbeitsgruppen sind u.U. über den ganzen Globus verstreut

  → Eine zeitliche Spezifikation der Abläufe ist wünschenswert
  - Sie ermöglicht die Entwicklung top-down zu strukturieren
  - Wird durch eine manuelle, statische Ablaufplanung unterstützt

- Struktur zyklischer Ablaufpläne $\rightsquigarrow$ gute Anordnung, Determinismus
  - Rahmen, Rahmenlänge, Scheiben; *major/minor cycle*

Resümee

- Entwicklungsprozesse führen verschiedenste Akteure zusammen
  - Firmen/Arbeitsgruppen sind u.U. über den ganzen Globus verstreut

  → Eine zeitliche Spezifikation der Abläufe ist wünschenswert
  - Sie ermöglicht die Entwicklung top-down zu strukturieren
  - Wird durch eine manuelle, statische Ablaufplanung unterstützt

- Struktur zyklischer Ablaufpläne $\rightsquigarrow$ gute Anordnung, Determinismus
  - Rahmen, Rahmenlänge, Scheiben; *major/minor cycle*

- Algorithmische Einplanung ordnet gerichtete, azyklische Graphen

  → Entlastung bei der Lösung eines komplexen Problems
  - List-Scheduling- und Branch&Bound-Algorithmen

Resümee

- Entwicklungsprozesse führen verschiedenste Akteure zusammen
  - Firmen/Arbeitsgruppen sind u.U. über den ganzen Globus verstreut

  → Eine zeitliche Spezifikation der Abläufe ist wünschenswert
  - Sie ermöglicht die Entwicklung top-down zu strukturieren
  - Wird durch eine manuelle, statische Ablaufplanung unterstützt
- Struktur zyklischer Ablaufpläne $\rightsquigarrow$ gute Anordnung, Determinismus
  - Rahmen, Rahmenlänge, Scheiben; *major/minor cycle*
- Algorithmische Einplanung ordnet gerichtete, azyklische Graphen

  → Entlastung bei der Lösung eines komplexen Problems
  - List-Scheduling- und Branch&Bound-Algorithmen
- Moduswechsel durch aperiodischen oder sporadischen Auftrag
  - Tabellenwechsel, Betriebsmittelfreigabe/-anforderung, Nachladen

Literaturverzeichnis

[1] Abdelzaher, T. F. ; Shin, K. G.:
Combined Task and Message Scheduling in Distributed Real-Time Systems.
In: *IEEE Transactions on Parallel and Distributed Systems* 10 (1999), Nr. 11, S. 1179–1191.
http://dx.doi.org/10.1109/71.809575. –
DOI 10.1109/71.809575

[2] Adam, T. L. ; Chandy, K. M. ; Dickson, J. R.:
A comparison of list schedules for parallel processing systems.
In: *Communications of the ACM* 17 (1974), Nr. 12, S. 685–690.
http://dx.doi.org/10.1145/361604.361619. –
DOI 10.1145/361604.361619. –
ISSN 0001–0782

[3] Baker, T. P. ; Shaw, A. C.:
The Cyclic Executive Model and Ada.
In: *Proceedings of the 9th IEEE Real-Time Systems Symposium (RTSS '88)*, IEEE Computer Society Press, 1988, S. 120–129

[4] Casavant, T. L. ; Kuhl, J. G.:
A taxonomy of scheduling in general-purpose distributed computing systems.
In: *IEEE Transactions on Software Engineering* 14 (1988), Nr. 2, S. 141–154.
http://dx.doi.org/10.1109/32.4634. –
DOI 10.1109/32.4634. –
ISSN 0098–5589

Literaturverzeichnis (Forts.)

[5] Kruatrachue, B. ; Lewis, T. G.:
Duplication Scheduling Heuristics (DSH): A New Precedence Task Scheduler for Parallel Processor Systems / Oregon State University.
Corvallis, OR, USA, 1976. –
Forschungsbericht

[6] Kwok, Y.-K. ; Ahmad, I. :
Static scheduling algorithms for allocating directed task graphs to multiprocessors.
In: *ACM Computing Surveys* 31 (1999), Nr. 4, S. 406–471.
http://dx.doi.org/10.1145/344588.344618. –
DOI 10.1145/344588.344618. –
ISSN 0360–0300

[7] Schild, K. ; Würtz, J. :
Off-Line Scheduling of a Real-Time System.
In: *Proceedings of the 13th ACM Symposium on Applied Computing (SAC '98)*.
New York, NY, USA : ACM Press, 1998. –
ISBN 0–89791–969–6, S. 29–38

Literaturverzeichnis (Forts.)

[8] Sih, G. C. ; Lee, E. A.:
A Compile-Time Scheduling Heuristic for Interconnection-Constrained Heterogeneous Processor Architectures.
In: *IEEE Transactions on Parallel and Distributed Systems* 4 (1993), Nr. 2, S. 175–187.
http://dx.doi.org/10.1109/71.207593. –
DOI 10.1109/71.207593. –
ISSN 1045–9219

EZS – Cheat Sheet

Typographische Konvention

Der erste Index gibt die Aufgabe an (z. B. D_i), der Zweite (optional) bezieht sich auf den Arbeitsauftrag (z. B. $d_{i,j}$). Exponenten zeigen verschiedene Varianten einer Eigenschaft an (z. B. T^{HI},T^{MED}, T^{LO}). Funktionen beschreiben zeitlich variierende Eigenschaften (z. B. $P(t)$).

Eigenschaften

t (Real-)Zeit
d Zeitverzögerung (engl. delay)

Strukturelemente

E_i Ereignis (engl. event)
R_i Ergebnis (engl. result)
T_i Aufgabe (engl. task)
$J_{i,j}$ Arbeitsauftrag (engl. job) der Aufgabe T_i

Temporale Eigenschaften

Allgemein

r_i Auslösezeitpunkt (engl. release time)
e_i Maximale Ausführungszeit (WCET)
D_i Relativer Termin (engl. deadline)
d_i Absoluter Termin
ω_i Antwortzeit (engl. response time)
σ_i Schlupf (engl. slack)

Periodische Aufgaben

p_i Periode (engl. period)
ϕ_i Phase (engl. phase)

Aufgaben – Tupel

$T_p = (p, e, D, \phi)$ Periodische Aufgabe ohne Priorität (zeitgesteuert oder dynamische Taskpriorität), $D = p$ und $\phi = 0$ sind der Reihe nach optional

$J_{i,j} = (r_{i,j}, e_{i,j}, d_{i,j})$ Arbeitsauftrag

Ablaufplanung

P_i Priorität (engl. priority) der Aufgabe T_i
Ω_i Prioritätsebenen (engl. number of priorities)
$h_{\Delta t}$ Rechenzeitbedarf (engl. demand)
$u_{\Delta t}$ CPU-Auslastung (engl. utilisation)
U Absolute CPU-Auslastung
H Hyperperiode (großer Durchlauf, engl. major cycle)
f Rahmenlänge (kleiner Durchlauf, engl. minor cycle)
e_i^f WCET aller Aufträge im Rahmen i