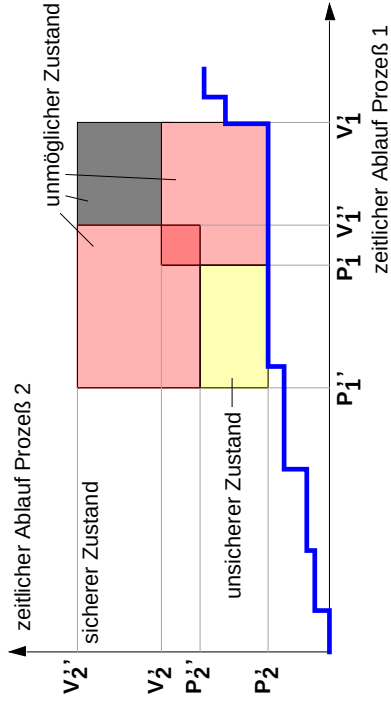


1 Sichere und unsichere Zustände (4)

- Beispiel von Folie H.13:



- ◆ Prozess 2 darf P_2' nicht durchführen und muß warten

SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

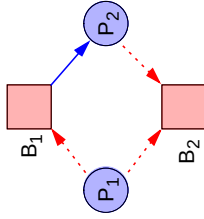
H-Deadline.doc: 1998-01-21 09:27

Reproduzieren oder Abdruck ist ohne Genehmigung der Universität Erlangen-Nürnberg strafbar. Bei der Zustimmung des Autors.

H.17

2 Betriebsmittelgraph

- Annahme: eine Instanz pro Betriebsmitteltyp
- ◆ Einsatz von Betriebsmittelgraphen zur Erkennung unsicherer Zustände



- ◆ zusätzliche Kanten zur Darstellung möglicher Anforderungen (Ansprüche, *Claims*)
- ◆ Anspruchskanten werden gestrichelt dargestellt und bei Anforderung in Anforderungskanten umgewandelt
- ◆ Anforderung und Belegung von B_2 durch P_1 führt in einen unsicheren Zustand (siehe Beispiel von Folie H.13)

SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

H-Deadline.doc: 1998-01-21 09:27

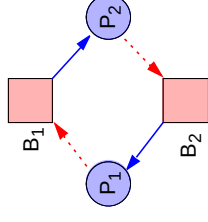
Reproduzieren oder Abdruck ist ohne Genehmigung der Universität Erlangen-Nürnberg strafbar. Bei der Zustimmung des Autors.

H.18

2 Betriebsmittelgraph (2)

- Erkennung des unsicheren Zustands an Zyklen im erweiterten Betriebsmittelgraph

- ◆ Anforderung und Belegung von B_2 durch P_1 führt zu:



- ◆ Zyklerkennung hat einen Aufwand von $O(n^2)$
- ▲ Betriebsmittelgraph nicht anwendbar bei mehreren Instanzen eines Betriebsmitteltyps
- ◆ Banker's Algorithmus (siehe Betriebsprogrammierung I)

SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

H-Deadline.doc: 1998-01-21 09:27

Reproduzieren oder Abdruck ist ohne Genehmigung der Universität Erlangen-Nürnberg strafbar. Bei der Zustimmung des Autors.

H.19

H.5 Erkennung von Verklemmungen

- Systeme ohne Mechanismen zur Vermeidung oder Verhinderung von Verklemmungen
- ◆ Verklemmungen können auftreten
- ◆ Verklemmung sollte als solche erkannt werden
- ◆ Auflösung der Verklemmung sollte eingeleitet werden (Algorithmus nötig)

1 Wartegraphen

- Annahme: nur eine Instanz pro Betriebsmitteltyp
- ◆ Einsatz von Wartegraphen, die aus dem Betriebsmittelgraphen gewonnen werden können

SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

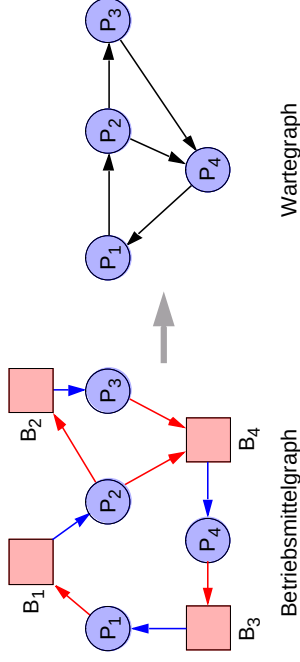
H-Deadline.doc: 1998-01-21 09:27

Reproduzieren oder Abdruck ist ohne Genehmigung der Universität Erlangen-Nürnberg strafbar. Bei der Zustimmung des Autors.

H.20

1 Wartegraphen (2)

- Wartegraphen
 - ◆ Betriebsmittel und Kanten werden aus Betriebsmittelgraph entfernt
 - ◆ zwischen zwei Prozessen wird eine „wartet auf“-Kante eingeführt, wenn es Kanten vom ersten Prozeß zu einem Betriebsmittel und von diesem zum zweiten Prozeß gabe



1 Wartegraphen (3)

- Erkennung von Verklemmungen
 - ◆ Wartegraph enthält Zyklen: System ist verklemmt
- ▲ Betriebsmittelgraph nicht für Systeme geeignet, die mehrere Instanzen pro Betriebsmitteltyp zulassen
 - ◆ kompliziertere Algorithmen ähnlich dem Banker's Algorithmus nötig

2 Erkennung durch Reduktionsverfahren

- Annahmen:
 - ◆ m Betriebsmitteltypen; Typ i verfügt über b_i Instanzen
 - ◆ n Prozesse
- Definitionen
 - ◆ B ist der Vektor $(b_1, b_2, \dots, b_m)$ der vorhandenen Instanzen
 - ◆ R ist der Vektor $(r_1, r_2, \dots, r_m)$ der noch verfügbaren Restinstanzen
 - ◆ C_j sind die Vektoren $(c_{j,1}, c_{j,2}, \dots, c_{j,m})$ der aktuellen Belegung durch den Prozeß j
- Es gilt: $\sum_{i=1}^n c_{i,j} + r_j = b_j$ für alle $1 \leq j \leq m$

2 Erkennung durch Reduktionsverfahren (2)

- Weitere Definitionen
 - ◆ A_j sind die Vektoren $(a_{j,1}, a_{j,2}, \dots, a_{j,m})$ der aktuellen Anforderungen durch den Prozeß j
 - ◆ zwei Vektoren A und B stehen in der Relation $A \leq B$, falls die Elemente der Vektoren jeweils paarweise in der gleichen Relation stehen
- Algorithmus
 - (1) alle Prozesse sind zunächst unmarkiert
 - (2) wähle einen Prozeß i , so daß $A_i \leq R$
(Prozeß ist ohne Verklemmung ausführbar)
 - (3) falls ein solcher Prozeß i existiert, addiere C_i zu R , markiere Prozeß i und beginne wieder bei Punkt (2)
(Bei Terminierung wird der Prozeß alle Betriebsmittel freigegeben)
 - (4) falls ein solcher Prozeß nicht existiert, terminiere Algorithmus
- ◆ alle nicht markierten Prozesse sind an einer Verklemmung beteiligt

2 Erkennung durch Reduktionsverfahren

■ Beispiel

- ◆ $m = 4$; $B = (4, 2, 3, 1)$
- ◆ $n = 3$; $C_1 = (0, 0, 1, 0)$; $C_2 = (2, 0, 0, 1)$; $C_3 = (0, 1, 2, 0)$
- ◆ daraus ergibt sich $R = (2, 1, 0, 0)$
- ◆ Anforderungen der Prozesse lauten:
 $A_1 = (2, 0, 0, 1)$; $A_2 = (1, 0, 1, 0)$; $A_3 = (2, 1, 0, 0)$

■ Ablauf

- ◆ Auswahl eines Prozesses: Prozeß 3, da $A_3 \leq R$; markiere Prozeß 3
- ◆ Addiere C_3 zu R : neues $R = (2, 2, 2, 0)$
- ◆ Auswahl eines Prozesses: Prozeß 2, da $A_2 \leq R$; markiere Prozeß 2
- ◆ Addiere C_2 zu R : neues $R = (4, 2, 2, 1)$
- ◆ Auswahl eines Prozesses: Prozeß 1, da $A_1 \leq R$; markiere Prozeß 1
- ◆ kein Prozeß mehr markiert: keine Verklemmung

SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

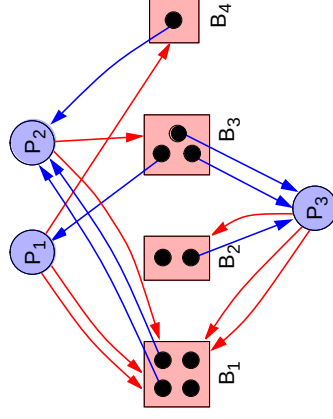
H-Deadline.doc: 1998-01-21 09:27

Reproduzieren oder die Weiterverbreitung dieser Unterlagen ist ohne schriftliche Genehmigung der Universität Erlangen-Nürnberg. Kopieren ist ausdrücklich untersagt.

H.25

3 Graphische Reduktion

■ Betriebsmittelgraph des Beispiels



- ◆ Auswahl eines Prozesses für den Anforderungen erfüllbar: nur P_3 möglich
- ◆ Löschen aller Kanten des Prozesses

SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

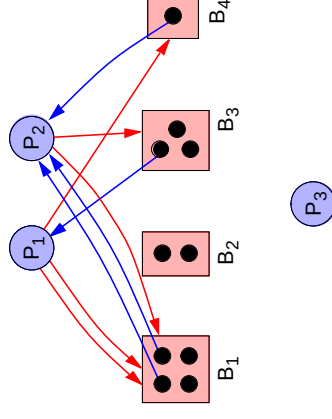
H-Deadline.doc: 1998-01-21 09:27

Reproduzieren oder die Weiterverbreitung dieser Unterlagen ist ohne schriftliche Genehmigung der Universität Erlangen-Nürnberg. Kopieren ist ausdrücklich untersagt.

H.26

3 Graphische Reduktion (2)

■ Betriebsmittelgraph des Beispiels (1. Reduktion)



- ◆ Auswahl eines Prozesses für den Anforderungen erfüllbar: nur P_2 möglich
- ◆ Löschen aller Kanten des Prozesses

SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

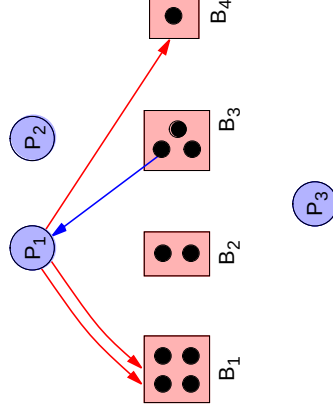
H-Deadline.doc: 1998-01-21 09:27

Reproduzieren oder die Weiterverbreitung dieser Unterlagen ist ohne schriftliche Genehmigung der Universität Erlangen-Nürnberg. Kopieren ist ausdrücklich untersagt.

H.27

3 Graphische Reduktion (3)

■ Betriebsmittelgraph des Beispiels (2. Reduktion)



- ◆ Auswahl eines Prozesses für den Anforderungen erfüllbar: P_1
- ◆ Löschen aller Kanten des Prozesses

SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

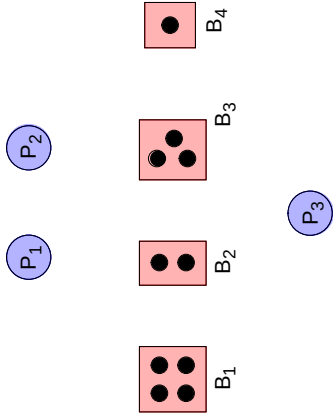
H-Deadline.doc: 1998-01-21 09:27

Reproduzieren oder die Weiterverbreitung dieser Unterlagen ist ohne schriftliche Genehmigung der Universität Erlangen-Nürnberg. Kopieren ist ausdrücklich untersagt.

H.28

3 Graphische Reduktion (4)

- Betriebsmittelgraph des Beispiels (3. Reduktion)



- ◆ es bleiben keine Prozesse mit Anforderungen übrig → keine Verklemmung
- ◆ übrig bleibende Prozesse sind verklemmt und in einem Zyklus

SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

H-Deadline.doc: 1998-01-21 09:27

Reproduzieren oder Abdruck ohne Erlaubnis der Universität Erlangen-Nürnberg ist strafbar. Bei der Zustimmung des Autors.

H.29

4 Einsatz der Verklemmungserkennung

- Wann sollte Erkennung ablaufen?
 - ◆ Erkennung ist aufwendig (Aufwand $O(n^2)$ bei Zyklenerkennung)
 - ◆ Häufigkeit von Verklemmungen eher gering
 - ◆ zu häufig: Verschwendung von Ressourcen zur Erkennung
 - ◆ zu selten: Betriebsmittel werden nicht optimal genutzt, Anzahl der verklemmten Prozesse steigt
- Möglichkeiten:
 - ◆ Erkennung, falls eine Anforderung nicht sofort erfüllt werden kann
 - ◆ periodische Erkennung (z.B. einmal die Stunde)
 - ◆ CPU Auslastung beobachten; falls Auslastung sinkt Erkennung starten

SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

H-Deadline.doc: 1998-01-21 09:27

Reproduzieren oder Abdruck ohne Erlaubnis der Universität Erlangen-Nürnberg ist strafbar. Bei der Zustimmung des Autors.

H.30

5 Erholung von Verklemmungen

- Verklemmung erkannt: Was tun?
 - ◆ Operateur benachrichtigen; manuelle Beseitigung
 - ◆ System erholt sich selbst
- Abbrechen von Prozessen (terminierte Prozesse geben ihre Betriebsmittel wieder frei)
 - ◆ alle verklemmten Prozesse abbrechen (großer Schaden)
 - ◆ einen Prozeß nach dem anderen abbrechen bis Verklemmung behoben (kleiner Schaden aber rechenzeitintensiv)
- ◆ mögliche Schäden:
 - Verlust von berechneter Information
 - Dateninkonsistenzen

SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

H-Deadline.doc: 1998-01-21 09:27

Reproduzieren oder Abdruck ohne Erlaubnis der Universität Erlangen-Nürnberg ist strafbar. Bei der Zustimmung des Autors.

H.31

5 Erholung von Verklemmungen (2)

- Entzug von Betriebsmitteln
 - ◆ Aussuchen eines „Opfer“-Prozesses (Aussuchen nach geringstem entstehendem Schaden)
 - ◆ Entzug der Betriebsmittel und Zurückfahren des „Opfer“-Prozesses (Prozeß wird in einen Zustand zurückgefahren, der unkritisch ist; benötigt Checkpoint oder Transaktionsverarbeitung)
 - ◆ Verhinderung von Aushungerung (es muß verhindert werden, daß immer derselbe Prozeß Opfer wird und damit keinen Fortschritt mehr macht)

SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

H-Deadline.doc: 1998-01-21 09:27

Reproduzieren oder Abdruck ohne Erlaubnis der Universität Erlangen-Nürnberg ist strafbar. Bei der Zustimmung des Autors.

H.32