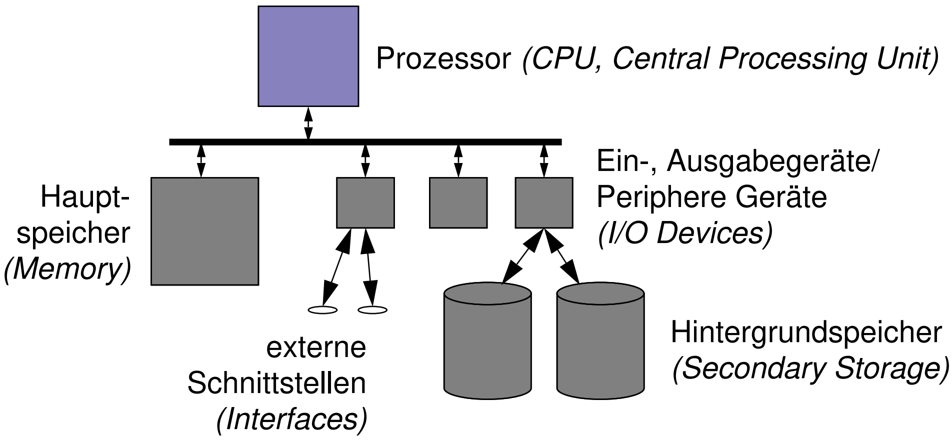
G Verklemmungen

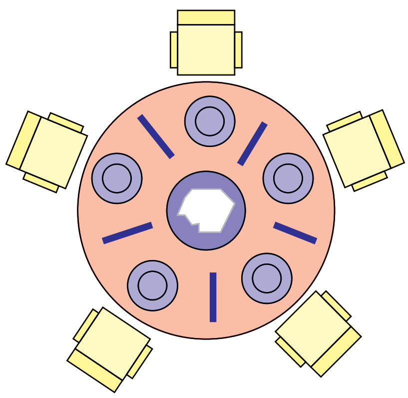
- Einordnung:



  - Verhalten von Aktivitätsträgern / Prozessen

G.1 Motivation

- Beispiel: die fünf Philosophen am runden Tisch

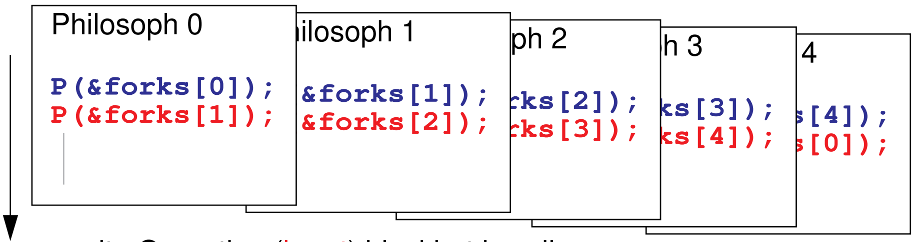


  - Philosophen denken oder essen
    "The life of a philosopher consists of an alternation of thinking and eating." (Dijkstra, 1971)
  - zum Essen benötigen sie zwei Gabeln, die jeweils zwischen zwei benachbarten Philosophen abgelegt sind

- Philosophen können verhungern, wenn sie sich „dumm“ anstellen.

G.1 Motivation (2)

- Problem der Verklemmung *(Deadlock)*
  - alle Philosophen nehmen gleichzeitig die linke Gabel auf und versuchen dann die rechte Gabel aufzunehmen



zweite Operation (in rot) blockiert jeweils

  - System ist **verklemmt:** Philosophen warten alle auf ihre Nachbarn

- Problemkreise:
  - Vermeidung und Verhinderung von Verklemmungen
  - Erkennung und Erholung von Verklemmungen

G.2 Betriebsmittelbelegung

- Betriebsmittel
  - CPU, Drucker, Geräte (Platten, CD-ROM, Floppy, Audio, usw.)
  - nur elektronisch vorhandene Betriebsmittel der Anwendung oder des Betriebssystems, z.B. Gabeln der Philosophen
- Unterscheidung von Typ und Instanz
  - Typ definiert ein Betriebsmittel eindeutig
  - Instanz ist eine Ausprägung des Typs
    (die Anwendung benötigt eine Instanz eines best. Typs, egal welche)
    - **CPU:** Anwendung benötigt eine von mehreren gleichartigen CPUs
    - **Drucker:** Anwendung benötigt einen von mehreren gleichen Druckern (falls Drucker nicht austauschbar und gleichwertig, so handelt es sich um verschiedene Typen)
    - **Gabeln:** jede Gabel ist ein eigener Betriebsmitteltyp

1 Belegung

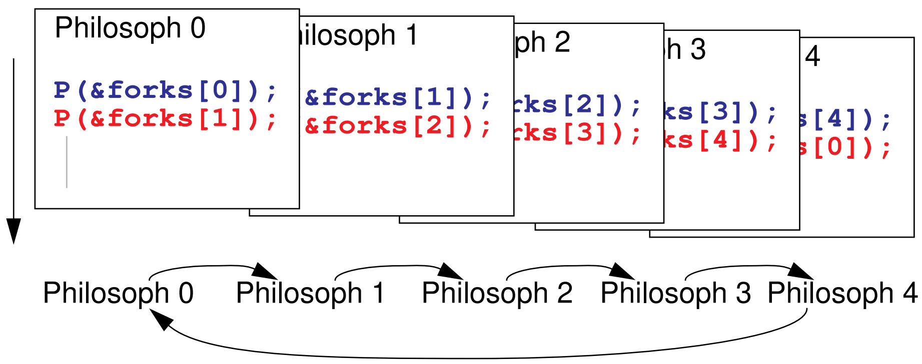
- Belegung erfolgt in drei Schritten
  - Anfordern des Betriebsmittels
    - blockiert evtl. falls Betriebsmittel nur exklusiv benutzt werden kann
    - **Gabel:** nur exklusiv
    - **Bildschirmausgabe:** exklusiv oder nicht-exklusiv
  - Nutzen des Betriebsmittels
    - **Gabel:** Philosoph kann essen
    - **Drucker:** Anwendung kann drucken
  - Freigeben des Betriebsmittels
    - **Gabel:** Philosoph legt Gabel wieder zwischen die Teller

2 Voraussetzungen für Verklemmungen

- Vier notwendige Bedingungen
  - *Exklusive Belegung*
    Mindestens ein Betriebsmitteltyp muss nur exklusiv belegbar sein.
  - *Nachforderungen von Betriebsmittel möglich*
    Es muss einen Prozess geben, der bereits Betriebsmittel hält, und ein neues Betriebsmittel anfordert.
  - *Kein Entzug von Betriebsmitteln möglich*
    Betriebsmittel können nicht zurückgefordert werden bis der Prozess sie wieder freigibt.
  - *Zirkuläres Warten*
    Es gibt einen Ring von Prozessen, in dem jeder auf ein Betriebsmittel wartet, das der Nachfolger im Ring besitzt.

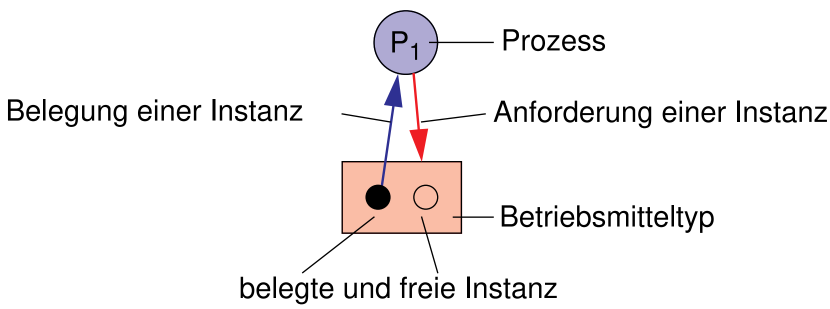
2 Voraussetzungen für Verklemmung (2)

- Beispiel: fünf Philosophen
  - Exklusive Belegung: **ja**
  - Nachforderungen von Betriebsmittel möglich: **ja**
  - Entzug von Betriebsmitteln: **nicht vorgesehen**
  - Zirkuläres Warten: **ja**



3 Betriebsmittelgraphen

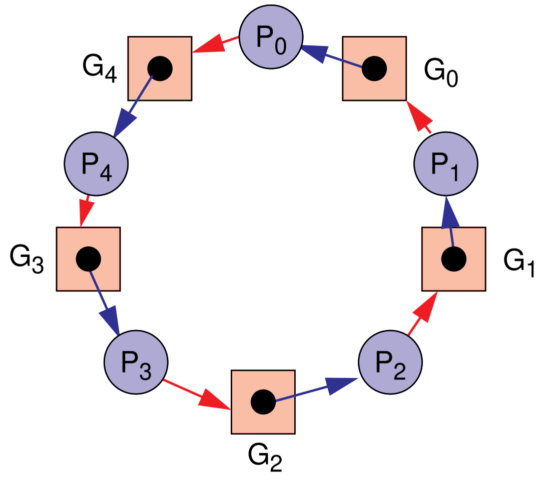
- Veranschaulichung der Belegung und Anforderung durch Graphen (nur exklusive Belegungen)



- Regeln:
  - kein Zyklus im Graph → keine Verklemmung
  - Zyklus im Graph ↛ Verklemmung
  - nur jeweils eine Instanz pro Betriebsmitteltyp und Zyklus → Verklemmung

3 Betriebsmittelgraphen (2)

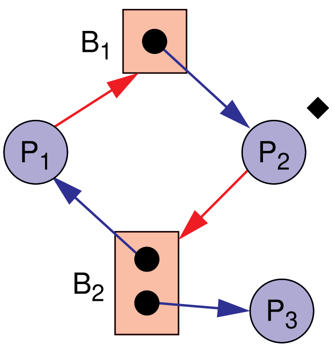
- Beispiel: fünf Philosophen



- ◆ Zyklus und jeder Betriebsmitteltyp hat nur eine Instanz → Verklemmung

3 Betriebsmittelgraphen (3)

- Beispiel mit Zyklus und ohne Verklemmung



- ◆ Prozess 3 kann seine Instanz vom Betriebsmitteltyp B_2 wieder zurückgeben und den Zyklus damit auflösen

G.3 Vermeidung/Vorbeugung von Verklemmungen

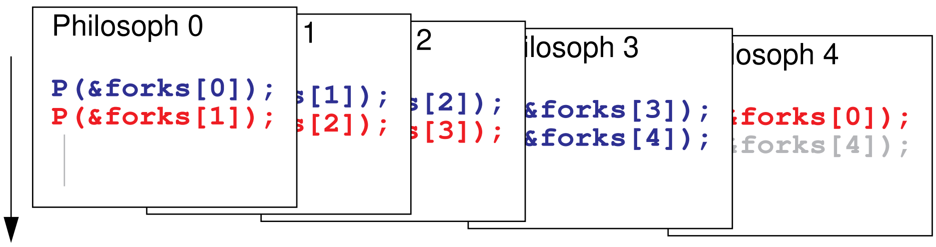
- *Deadlock prevention*
- Ansatz: Vermeidung der notwendigen Bedingungen für Verklemmungen
  - *Exklusive Belegung:*
    oft nicht vermeidbar
  - *Nachforderungen von Betriebsmittel möglich:*
    alle Betriebsmittel müssen auf einmal angefordert werden
    - ungenutzte aber belegte Betriebsmittel vorhanden
    - Aushungerung möglich: ein anderer Prozess hält immer das nötige Betriebsmittel belegt

G.3 Vermeidung von Verklemmungen (2)

- *Kein Entzug von Betriebsmitteln möglich:*
  Entzug von Betriebsmitteln erlauben
  - bei neuer Belegung werden alle gehaltenen Betriebsmittel freigegeben und mit der neuen Anforderung zusammen wieder angefordert
  - während ein Prozess wartet, werden seine bereits belegten Betriebsmittel anderen Prozessen zur Verfügung gestellt
  - möglich für CPU oder Speicher jedoch nicht für Drucker, Bandlaufwerke oder ähnliche
- *Zirkuläres Warten:* Vermeidung von Zyklen
  - Totale Ordnung auf Betriebmitteltypen

G.3 Vermeidung von Verklemmungen (3)

- Anforderungen nur in der Ordnungsreihenfolge erlaubt

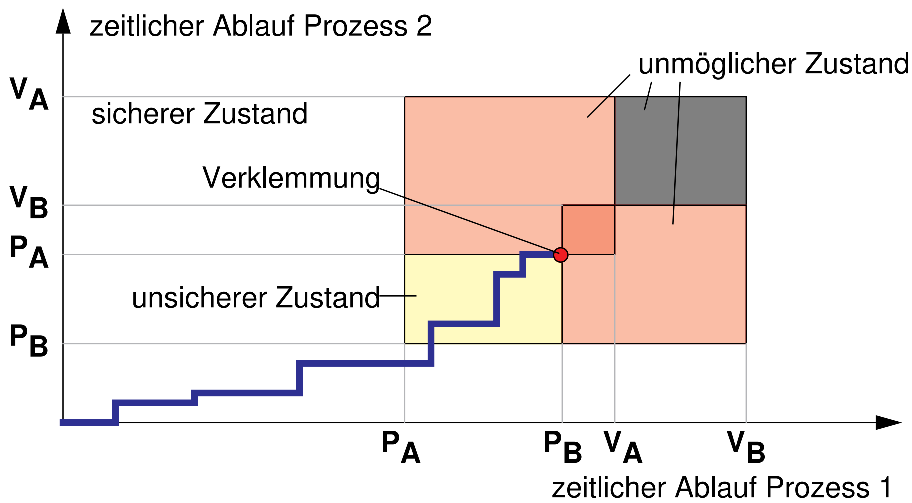


  z.B. Gabeln: geordnet nach Gabelnummer

- Bei neuer Anforderung wird geprüft, ob letzte Anforderung kleiner bzgl. der totalen Ordnung war (Instanzen gleichen Typs müssen gleichzeitig angefordert werden); sonst: Abbruch mit Fehlermeldung
- Philosoph 4 bekäme eine Fehlermeldung, wenn er in der obigen Situation zuerst Gabel 4 und dann Gabel 0 anfordert: Rückgabe und neuer Versuch

G.4 Verhinderung von Verklemmungen

- *Deadlock avoidance*
- Annahme: es ist bekannt, welche Betriebsmittel ein Prozess brauchen wird (hier je zwei binäre Semaphore A und B)
  - Betriebssystem überprüft System auf unsichere Zustände



1 Sichere und unsichere Zustände

- Sicherer Zustand
  - Es gibt eine Sequenz, in der die vorhandenen Prozesse abgearbeitet werden können, so dass ihre Anforderungen immer befriedigt werden können.
  - Sicherer Zustand erlaubt immer eine verklemmungsfreie Abarbeitung

- Unsicherer Zustand
  - Es gibt keine solche Sequenz.
  - Verklemmungszustand ist ein unsicherer Zustand
  - Ein unsicherer Zustände führt zwangsläufig zur Verklemmung, wenn die Prozesse ihre angenommenen Betriebsmittel wirklich anfordern bevor sie von anderen Prozessen wieder freigegeben werden.

1 Sichere und unsichere Zustände (2)

- Beispiel:
  - 12 Magnetbandlaufwerke vorhanden
  - P_0 braucht (bis zu) 10 Laufwerke
  - P_1 braucht (bis zu) 4 Laufwerke
  - P_2 braucht (bis zu) 9 Laufwerke

  - Aktuelle Situation: P_0 hat 5, P_1 hat 2 und P_2 hat 2 Laufwerke
  - Zustand sicher?

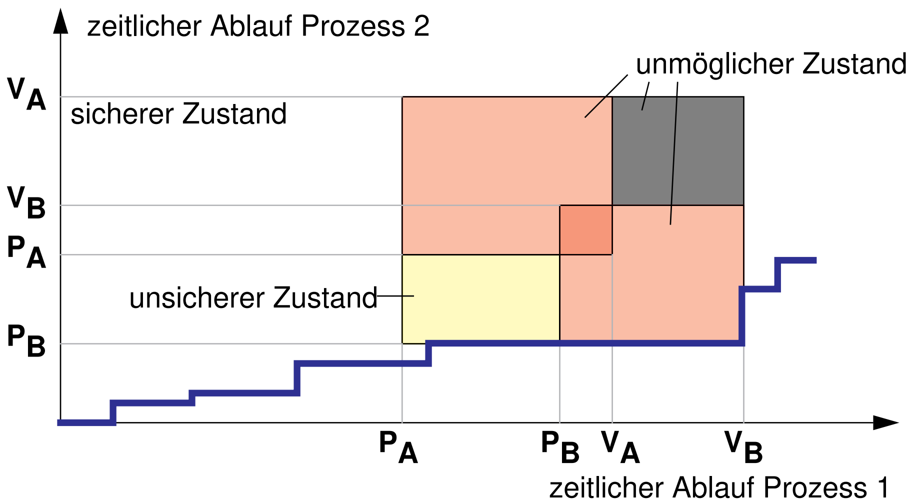
  - Aktuelle Situation: P_0 hat 5, P_1 hat 2 und P_2 hat 3 Laufwerke
  - Zustand sicher?

1 Sichere und unsichere Zustände (3)

- Verhinderung von Verklemmungen
  - Verhinderung von unsicheren Zuständen
  - Anforderungen blockieren, falls sie in einen unsicheren Zustand führen würden
- Beispiel von Folie H.16:
  - Zustand: P_0 hat 5, P_1 hat 2 und P_2 hat 2 Laufwerke
  - P_2 fordert ein zusätzliches Laufwerk an
  - Belegung würde in unsicheren Zustand führen: P_2 muss warten
- ▲ Verhinderung von unsicheren Zuständen schränkt Nutzung von Betriebsmitteln ein
  - verhindert aber Verklemmungen

1 Sichere und unsichere Zustände (4)

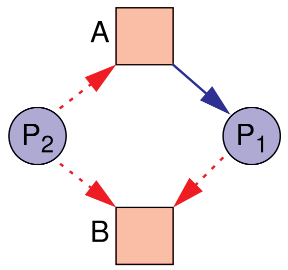
- Beispiel von Folie H.14:



  - Prozess 2 darf $\mathbf{P_B}$ nicht durchführen und muss warten

2 Betriebsmittelgraph

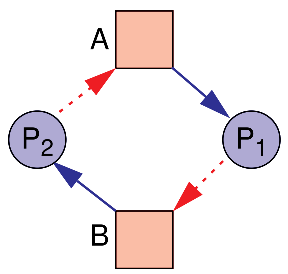
- Annahme: eine Instanz pro Betriebsmitteltyp
  - Einsatz von Betriebsmittelgraphen zur Erkennung unsicherer Zustände



  - zusätzliche Kanten zur Darstellung möglicher Anforderungen (Ansprüche, *Claims*)
  - Anspruchskanten werden gestrichelt dargestellt und bei Anforderung in Anforderungskanten umgewandelt
  - Anforderung und Belegung von B durch P_2 führt in einen unsicheren Zustand (siehe Beispiel von Folie H.14)

2 Betriebsmittelgraph (2)

- Erkennung des unsicheren Zustands an Zyklen im erweiterten Betriebsmittelgraph
  - Anforderung und Belegung von B durch P_2 führt zu:



  - Zyklenerkennung hat einen Aufwand von $O(n^2)$

▲ Betriebsmittelgraph nicht anwendbar bei mehreren Instanzen eines Betriebsmitteltyps

3 Banker's Algorithm

- Erkennung unsicherer Zustände bei mehreren Instanzen pro Betriebsmitteltyp
- Annahmen:
  - m Betriebsmitteltypen; Typ i verfügt über b_i Instanzen
  - n Prozesse
- Definitionen
  - B ist der Vektor $(b_1, b_2, \ldots b_m)$ der vorhandenen Instanzen
  - R ist der Vektor $(r_1, r_2, \ldots r_m)$ der noch verfügbaren Restinstanzen
  - C_j sind die Vektoren $(c_{j,1}, c_{j,2}, \ldots c_{j,m})$ der aktuellen Belegung durch den Prozess j
- Es gilt: $\sum_{j=1}^{n} c_{j,i} + r_i = b_i$ für alle $1 \leq i \leq m$

3 Banker's Algorithm (2)

- Weitere Definitionen
  - M_j sind die Vektoren $(m_{j,1}, m_{j,2}, \ldots m_{j,m})$ der bekannten maximalen Belegung der Betriebsmittel 1 bis m 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
    z.B. $(1, 2, 3) \leq (2, 2, 4)$

3 Banker's Algorithm (3)

- Algorithmus
  1. alle Prozesse sind zunächst unmarkiert
  2. wähle einen nicht markierten Prozess j, so dass $M_j - C_j \leq R$
     (Prozess ist ohne Verklemmung ausführbar, selbst wenn er alles anfordert, was er je brauchen wird)
  3. falls ein solcher Prozess j existiert, addiere C_j zu R, markiere Prozess j und beginne wieder bei Punkt (2)
     (Bei Terminierung wird der Prozess alle Betriebsmittel freigeben)
  4. falls ein solcher Prozess nicht existiert, terminiere Algorithmus

  - Sind alle Prozesse markiert, ist das System in einem sicheren Zustand.

4 Beispiel

- Beispiel:
  - 12 Magnetbandlaufwerke vorhanden
  - P_0 braucht (bis zu) 10 Laufwerke
  - P_1 braucht (bis zu) 4 Laufwerke
  - P_2 braucht (bis zu) 9 Laufwerke
  - Aktuelle Situation: P_0 hat 5, P_1 hat 2 und P_2 hat 3 Laufwerke
- Belegung der Datenstrukturen
  - $m = 1$
  - $n = 3$
  - $B = (12)$
  - $R = (2)$
  - $C_0 = (5)$, $C_1 = (2)$, $C_2 = (3)$
  - $M_0 = (10)$, $M_1 = (4)$, $M_2 = (9)$

4 Beispiel (2)

- Anwendung des Banker's Algorithm
  - wähle einen nicht markierten Prozess j, so dass $M_j - C_j \leq R$
    - ➔ P_1
  - $R := R + C_1$ ➔ $R = (4)$
  - wähle einen nicht markierten Prozess j, so dass $M_j - C_j \leq R$
    - ➔ kein geeigneter Prozess vorhanden
  - Zustand ist unsicher

G.5 Erkennung von Verklemmungen

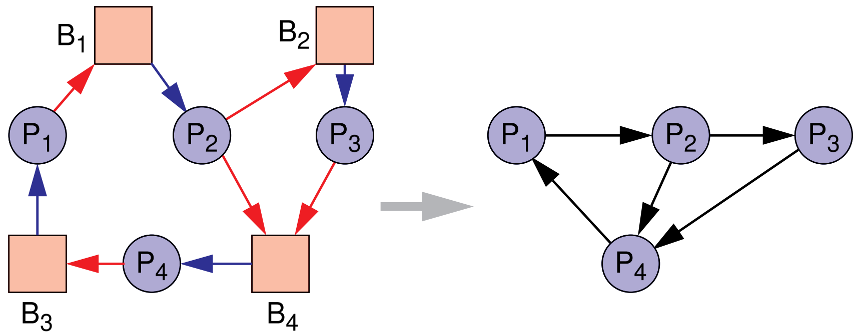
- Systeme ohne Mechanismen zur Vermeidung oder Verhinderung von Verklemmungen
  - Verklemmungen können auftreten
  - Verklemmung sollte als solche erkannt werden
  - Auflösung der Verklemmung sollte eingeleitet werden (Algorithmus nötig)

1 Wartegraphen

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

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 gibt



Betriebsmittelgraph Wartegraph

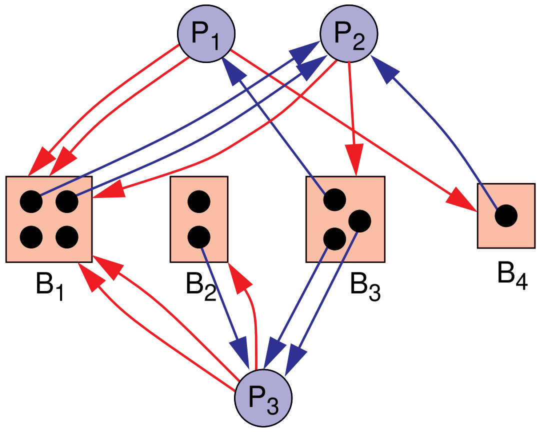
1 Wartegraphen (3)

- Erkennung von Verklemmungen
  - Wartegraph enthält Zyklen: System ist verklemmt

▲ Betriebsmittelgraph nicht für Systeme geeignet, die mehrere Instanzen pro Betriebsmitteltyp zulassen

2 Erkennung durch graphische Reduktion

- Betriebsmittelgraph des Beispiels

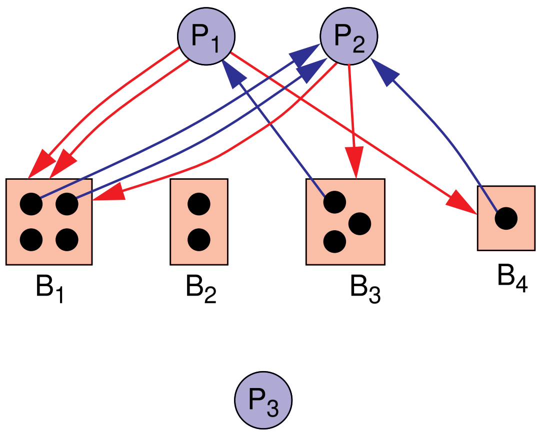


◆ Auswahl eines Prozesses für den Anforderungen erfüllbar: nur P_3 möglich

◆ Löschen aller Kanten des Prozesses

2 Erkennung durch graphische Reduktion (2)

- Betriebsmittelgraph des Beispiels (1. Reduktion)

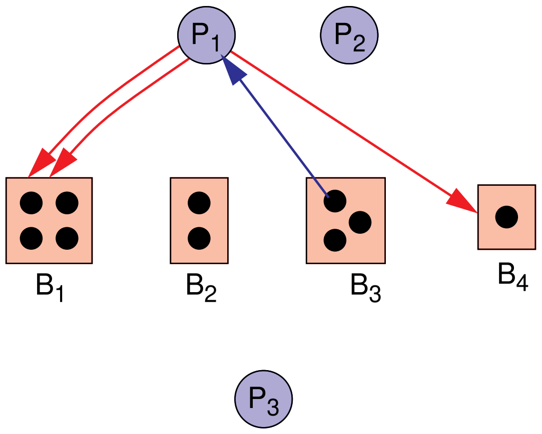


◆ Auswahl eines Prozesses für den Anforderungen erfüllbar: nur P_2 möglich

◆ Löschen aller Kanten des Prozesses

2 Erkennung durch graphische Reduktion (3)

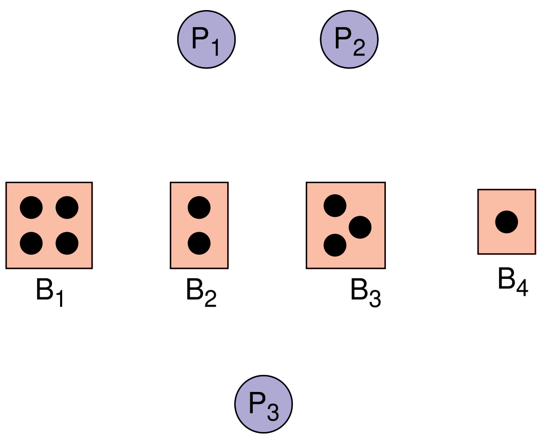
- Betriebsmittelgraph des Beispiels (2. Reduktion)



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

2 Erkennung durch 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

3 Erkennung durch Reduktionsverfahren

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

3 Erkennung durch Reduktionsverfahren (2)

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

3 Erkennung durch Reduktionsverfahren (3)

- 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 unmarkiert: keine Verklemmung

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

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

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)

G.6 Kombination der Verfahren

- Einsatz verschiedener Verfahren für verschiedene Betriebsmittel
  - Interne Betriebsmittel:
    Vermeiden von Verklemmungen durch totale Ordnung der Betriebsmittel (z.B. IBM Mainframe-Systeme)
  - Hauptspeicher:
    Vermeiden von Verklemmungen durch Entzug des Speichers (z.B. durch Swap-Out)
  - Betriebsmittel eines Jobs:
    Angabe der benötigten Betriebsmittel beim Starten; Einsatz der Verhinderungsstrategie durch Feststellen unsicherer Zustände
  - Hintergrundspeicher (Swap-Space):
    Vorausbelegung des Hintergrundspeichers