

Verklemmungen (Deadlocks)



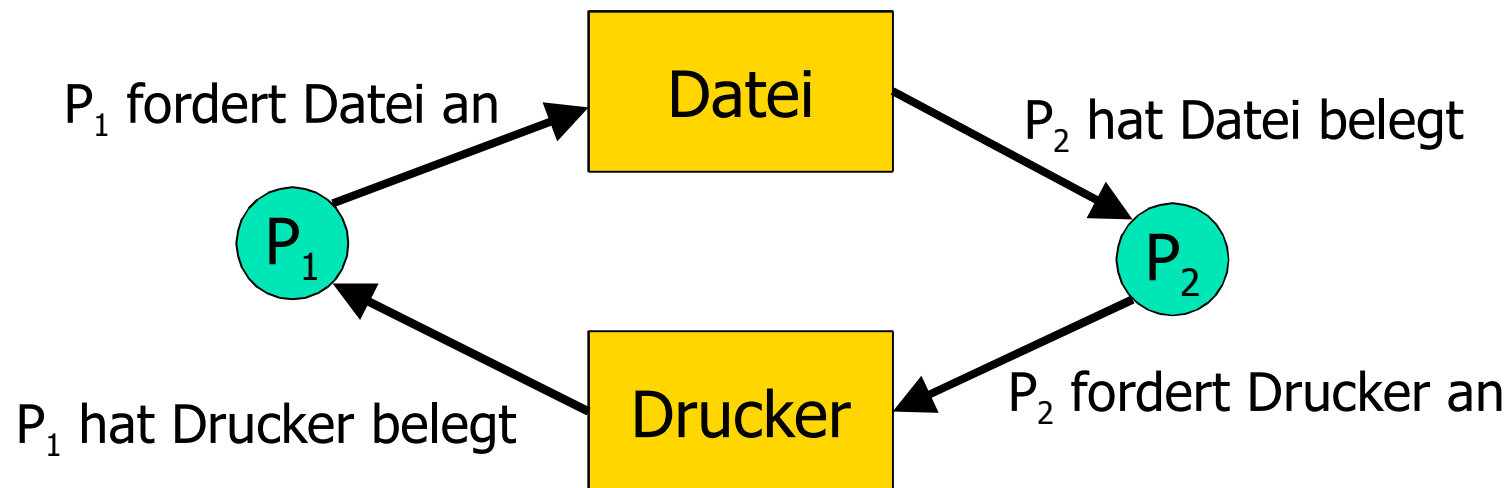
Überblick

- Was sind Verklemmungen?
- Betriebsmittel-Zuordnungsgraphen
- Bedingungen für Verklemmungen
- Umgang mit Verklemmungen
 - Unmöglich machen
 - Vermeiden
 - Erkennen und beseitigen
 - Ignorieren



Verklemmungen (deadlocks)

- Zwei oder mehr Prozesse hindern sich gegenseitig an der Ausführung.



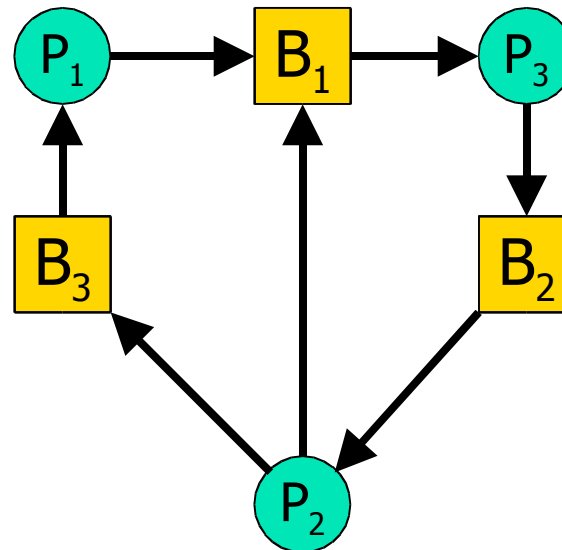
Graphische Modellierung einer Verklemmungssituation

 Prozess

 Betriebsmittel

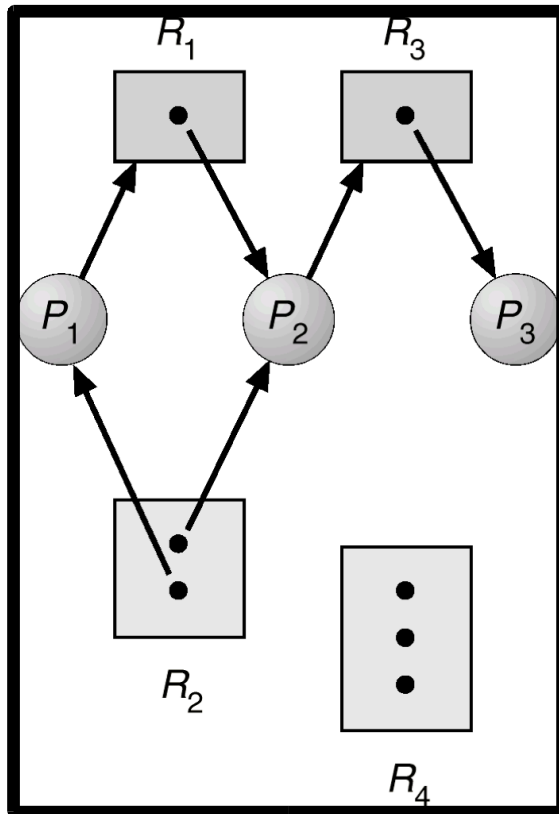
  Prozess fordert Betriebsmittel an

  Prozess belegt Betriebsmittel

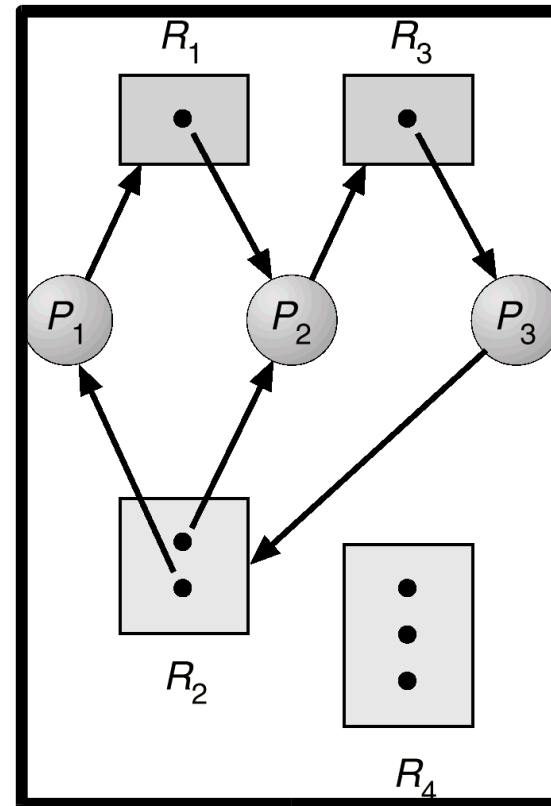


Zyklus im Graph:
Verklemmung!

Betriebsmittel-Zuordnungsgraph mit mehreren Exemplaren pro Ressource

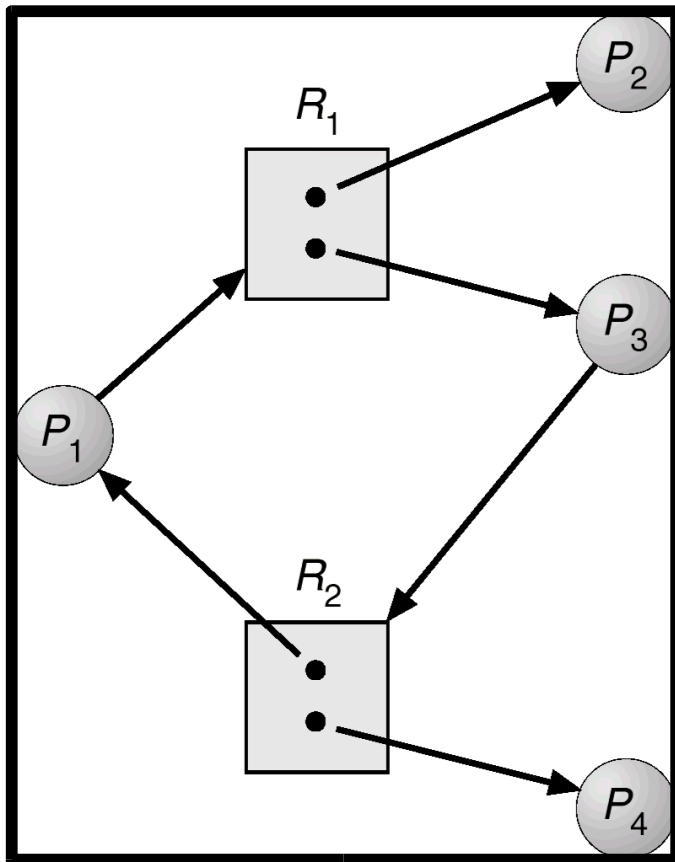


Keine Verklemmung



Verklemmung

Zyklischer Betriebsmittel-Zuordnungsgraph ohne Verklemmung



Nur ein Exemplar pro
Betriebsmittel:
Zyklus \square Verklemmung

Mehrere Exemplare pro
Betriebsmittel:
Zyklus \square Verklemmung

Zyklus, aber keine Verklemmung!



Bedingungen für Verklemmungen

- Vier **notwendige** und **hinreichende** Bedingungen für das Auftreten von Verklemmungen (Coffman, 1971):
 - **Wechselseitiger Ausschluss** der Betriebsmittelnutzung (mutual exclusion)
 - **Zusätzliche** Betriebsmittelanforderungen möglich (hold and wait)
 - Keine **vorzeitige** Rückgabe (no preemption)
 - **Zirkuläres** Warten (circular wait)



Verklemmungen und wie man damit umgeht...

- Verklemmungen unmöglich machen
- Verklemmungen vermeiden
- Verklemmungen erkennen und beseitigen
- Verklemmungen ignorieren



Verklemmungen unmöglich machen (I)

- Wechselseitigen Ausschluss verhindern
 - Z. B. Einrichten eines Druckerdaemonen
 - Probleme: nicht für alle Betriebsmittel geeignet, nur Verlagerung auf andere Betriebsmittel
- Zusätzliche Betriebsmittelanforderungen verbieten
 - Z. B. Anforderung aller benötigten Betriebsmittel zu Prozessbeginn
 - Probleme: Unnötig lange Belegung der Betriebsmittel, schlechte Betriebsmittelauslastung



Verklemmungen unmöglich machen (II)

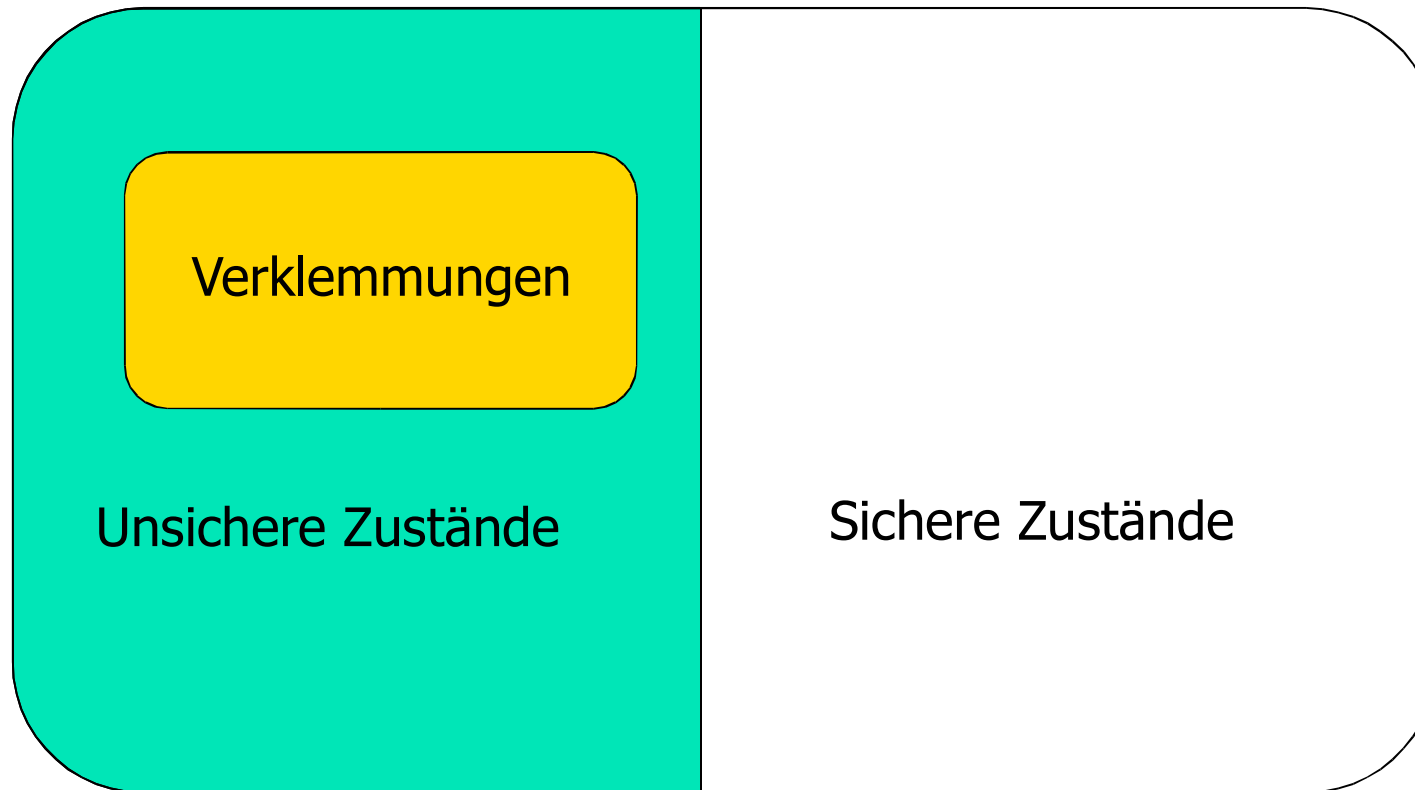
- Vorzeitige Betriebsmittelrückgabe **erzwingbar** machen
 - Z.B. Entzug nach einer bestimmten Zeit
 - Probleme: muss auf Programmebene berücksichtigt werden, bereits geleistete Arbeitsleistung geht verloren
- Zirkularität **unterbinden**
 - Z. B. lineare oder hierarchische Ordnung der Betriebsmittel, Anforderungen dann nur gemäß dieser Ordnung
 - Probleme: keine allgemein brauchbare Ordnung angebar, deshalb oft schlechte Auslastung

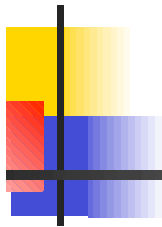


Verklemmungen vermeiden

- Betrachtung der Betriebsmittelanforderungen als gleichzeitig auftretende **Maximalforderungen**
- Unterscheidung von
 - **sicheren** Zuständen (Verklemmung nicht möglich)
 - **unsicheren** Zuständen (Verklemmung nicht zwingend, bei ungünstiger Anforderungsreihenfolge aber möglich)
- Weitere Prozesse werden nur gestartet, wenn **kein unsicherer** Zustand entsteht (Bankers-Algorithmus)

Sichere Zustände, unsichere Zustände, Verklemmungen





Einige Definitionen

E_r : Zahl der **existierenden** Exemplare des Betriebsmittels r

B_{kr} : Zahl der Exemplare des Betriebsmittels r , die Prozess k **bereits belegt** hat

Z_{kr} : Zahl der Exemplare des Betriebsmittels r , die Prozess k **insgesamt zusätzlich belegen** will

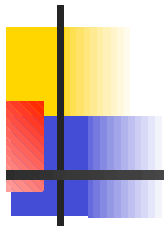
F_r : Zahl der **noch freien** Exemplare des Betriebsmittels s . Es gilt $F_r = E_r - \sum_k B_{kr}$

vermeiden (Banker's-Algorithm)



1. Alle Prozesse entmarkieren
2. Suche unmarkierten Prozess k bei dem für alle Betriebsmittel gilt: $F_r \leq Z_{kr}$
3. Falls es einen solchen Prozess gibt, markiere ihn und setze $F_r = F_r + B_{kr}$ für alle r .
4. Gibt es keinen solchen Prozess, halte an, ansonsten durchlaufe erneut Schritt 2 und 3

Genau dann wenn alle Prozesse markiert werden können, handelt es sich um einen sicheren Zustand.



Sichere Betriebsmittelanforderung

Sei A_{kr} die Anzahl der Betriebsmittel r , die Prozess k gerade anfordert.

2. Ist $A_{kr} > C_{kr}$ für wenigstens ein r
-> Fehler: Zu viele Ressourcen angefordert!
3. Ist $A_{kr} \leq F_r$ für alle r , gehe zu Schritt 3. Anderenfalls muss k warten, da nicht genug Betriebsmittel vorhanden sind.
4. Überprüfe, ob der Zustand, in den man gelangte, wenn man die Anforderung von k gewähren würde, ein sicherer Zustand ist. Erfülle die Anforderung nur in diesem Fall.



Beispiel

Anzahl der verfügbaren Betriebsmittel A=5, B=3, C=4, D=3

	A	B	C	D
P ₁	2	0	1	0
P ₂	0	1	0	2
P ₃	1	0	2	0
P ₄	1	1	0	0
P ₅	0	1	0	1

	A	B	C	D
P ₁	1	0	2	1
P ₂	1	0	1	0
P ₃	1	1	0	2
P ₄	4	0	2	1
P ₅	0	2	4	0

bestehende Belegung max. zusätzliche Anforderungen
 Anforderungen: a) P₂ fordert ein C b) P₃ fordert ein A



Verklemmungsvermeidung: Probleme

- I. A. Zahl der maximal benötigten Betriebsmittel **unbekannt**
- Ständig **wechselnde** Zahl von Prozessen
- Zahl der verfügbaren Betriebsmittel ebenfalls **veränderlich**
- Algorithmus ist **laufzeit-** und **speicherintensiv**



Verklemmungen erkennen

- Analyse bei **verdächtigen** Symptomen:
 - viele Prozesse **warten** und der Prozessor ist **unbeschäftigt**
 - **mindestens zwei Prozesse** warten zu lange auf Betriebsmittel
- Bei Verdacht start eines Erkennungsalgorithmus
 - Z. B. **Zyklen-Erkennung** im Betriebsmittelgraphen



Verklebungen beseitigen

- Prozesse **abbrechen**
- Prozesse **zurücksetzen**
- Betriebsmittel **entziehen**

Probleme:

- Prozess-/Betriebsmittelauswahl
- Verlust bereits geleisteter Arbeit
- Mögliche Inkonsistenzen
- U. U. manueller Mehraufwand erforderlich



Kriterien bei der Auswahl des abzubrechenden Prozesses

- Priorität des Prozesses
- Anzahl abzubrechender Prozesse
- Bisherige Laufzeit
- Noch verbleibende Laufzeit
- Belegte Betriebsmittel
- Noch fehlende Betriebsmittel
- Art des Prozesses (interaktiver Prozess oder Hintergrundprozess?)



Verklemmungen ignorieren

- Erkennung von Verklemmungen **aufwendig**
- Beseitigung von Verklemmungen **nicht unproblematisch**
- Vermeidung bzw. Unmöglichmachen von Verklemmungen u. U. **wenig effizient**
- Verklemmungen sind in der Regel nicht das **dringlichste** Problem