

---

## Aufgabe 8: Verklemmung

### 8.1 Definitionen

- a) Was genau ist eine Verklemmung (Deadlock) und welche vier Voraussetzungen müssen erfüllt sein, damit es dazu kommen kann? Erläutern Sie diese Voraussetzungen jeweils anhand eines Beispiels.



- b) Welche vier grundsätzlichen Ansätze zur Behandlung von Verklemmungen gibt es?



### 8.2 Betriebsmittelbelegungsgraphen

- a) Gegeben ist ein System bestehend aus vier Prozessen ( $P_1, P_2, P_3, P_4$ ), vier Ressourcen (a, b, c, d), sowie die unten dargestellte Zustandsbeschreibung.

- Prozess  $P_1$  belegt b und verlangt a sowie c
- Prozess  $P_2$  verlangt b, c und d
- Prozess  $P_3$  belegt d und verlangt c
- Prozess  $P_4$  belegt a und verlangt c

Zeichnen Sie einen Betriebsmittelbelegungsgraph und bestimmen Sie, ob der vorliegende Zustand sicher ist. **Begründen Sie Ihre Antwort.**



- b) Wie verhält sich das System, wenn die Ressource c an den Prozess  $P_1$ ,  $P_2$ ,  $P_3$  oder  $P_4$  zugewiesen wird? Erläutern Sie die Zuweisung für jeden Prozess einzeln.

### 8.3 Bankieralgorithmus

Der *Bankieralgorithmus* (engl. *banker's algorithm*) wurde 1965 von E. Dijkstra beschrieben. Im Folgenden wird eine an <sup>(1)</sup> angelehnte Form vorgestellt. Mit dem Algorithmus können Deadlocks verhindert werden, indem überprüft wird, ob die Zuteilung einer Ressource zu einem sicheren oder unsicheren Zustand des Systems führt. Ressourcen werden nur dann zugeteilt, wenn die Zuteilung zu einem sicheren Zustand führt.

Die Sicherheit bzw. Unsicherheit von Zuständen wird dabei wie folgt definiert: Ein Zustand ist *sicher*, wenn es eine Schedulingreihenfolge gibt, die nicht zu einem Deadlock führt, selbst wenn alle Prozesse sofort Ihre maximale Anzahl an Ressourcen anfordern. Ansonsten ist ein Zustand *unsicher*. Dabei ist ein unsicherer Zustand nicht zwangsweise ein Deadlock, allerdings kann das System unter Umständen in einen Deadlock geraten.

Der Bankieralgorithmus erkennt Deadlocks in einer Menge von  $n$  Prozessen  $P_1$  bis  $P_n$ . Die Prozesse konkurrieren um die Zuteilung von Ressourcen aus  $m$  verschiedenen *Ressourcenklassen*. Der Algorithmus arbeitet mit den vier folgenden Datenstrukturen:

- E: Der Ressourcenvektor (*existing resource vector*) der Länge  $m$  gibt die Anzahl von Ressourcen an, die von jeder Klasse insgesamt verfügbar sind.
- A: Der Ressourcenrestvektor (*available resource vector*) der Länge  $m$  enthält für jede Ressource  $i$  die Anzahl der freien Instanzen  $A_i$ .
- C: Die Belegungsmatrix (*current allocation matrix*) der Dimension  $m \times n$  enthält für jeden Prozess eine Zeile  $C_i$ , die die von diesem Prozess belegten Ressourcen angibt.
- R: Die Anforderungsmatrix (*request matrix*) der Dimension  $m \times n$  enthält ebenfalls für jeden Prozess eine Zeile  $R_i$ , die die Ressourcen beschreibt, die der Prozess  $i$  *zusätzlich* benötigt.

Im folgenden Beispiel gilt  $n = 3$  und  $m = 4$ , es konkurrieren also drei Prozesse um Ressourcen aus vier Klassen: Festplatte, Drucker, Scanner und Grafikausgabe. Es sind  $E$ ,  $A$ ,  $C$  und  $R$  gegeben:  $E$  gibt an, dass vier Festplatten, zwei Drucker, drei Grafikausgaben und ein Scanner am System vorhanden sind.  $A$  gibt an, dass zum betrachteten Zeitpunkt zwei Festplatten und ein Drucker verfügbar, d.h. nicht belegt sind.  $C$  gibt an, dass  $P_1$  eine Grafikausgabe,  $P_2$  zwei Festplatten und einen Scanner und  $P_3$  einen Drucker und zwei Grafikausgaben belegt.

$$\mathbf{E} = (4 \quad 2 \quad 3 \quad 1) \quad (1)$$

$$\mathbf{A} = (2 \quad 1 \quad 0 \quad 0) \quad (2)$$

---

<sup>(1)</sup>A. S. Tanenbaum, „Modern Operating Systems“, 3. Edition, Kapitel 6.5

$$\mathbf{C} = \begin{bmatrix} \text{Festplatte} & \text{Drucker} & \text{Grafik} & \text{Scanner} \\ 0 & 0 & 1 & 0 \\ 2 & 0 & 0 & 1 \\ 0 & 1 & 2 & 0 \end{bmatrix} \begin{matrix} P_1 \\ P_2 \\ P_3 \end{matrix} \quad (3)$$

$$\mathbf{R} = \begin{bmatrix} \text{Festplatte} & \text{Drucker} & \text{Grafik} & \text{Scanner} \\ 2 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 2 & 1 & 0 & 0 \end{bmatrix} \begin{matrix} P_1 \\ P_2 \\ P_3 \end{matrix} \quad (4)$$

Der Bankieralgorithmus basiert auf dem Vergleich von Vektoren, dabei gilt:  $A \leq B$  genau dann, wenn jedes Element von  $A$  kleiner oder gleich dem entsprechenden Element von  $B$  ist, d.h. wenn  $A_i \leq B_i$  für alle  $1 \leq i \leq m$ . Die Algorithmus arbeitet in den folgenden drei Schritten:

1. Suche eine Zeile aus  $R$ , deren zugehöriger Prozess noch nicht ausgeführt wurde und dessen Ressourcenbedarf kleiner oder gleich  $A$  ist. Gibt es keine solche Zeile, wird das System in einen Deadlock laufen, da kein Prozess zu Ende laufen kann.
2. Führe den gewählten Prozess aus und addiere seine vorher belegten Ressourcen aus  $C$  zu  $A$ .
3. Wiederhole Schritt 1 und 2 bis entweder alle Prozesse ausgeführt wurden *oder* bis ein Deadlock auftritt.

Die Ausführung des Bankieralgorithmus auf den oben angegebenen Eingaben führt zu folgendem Ablauf:

Wir suchen einen Prozess, dessen Anforderungen erfüllt werden können.  $P_1$  kann nicht zufriedengestellt werden, weil keine Grafikausgabe frei ist.  $P_2$  kann ebenfalls nicht ausgeführt werden, weil es keinen freien Scanner gibt.  $P_3$  dagegen kann ausgeführt werden und gibt nach Beendigung alle Ressourcen wieder frei, sodass nach der ersten Iteration gilt:

$$\mathbf{A} = \begin{pmatrix} 2 & 2 & 2 & 0 \end{pmatrix} \quad (5)$$

Nun kann  $P_2$  ausgeführt werden. Nachdem er die Ressourcen freigegeben hat gilt:

$$\mathbf{A} = \begin{pmatrix} 4 & 2 & 2 & 1 \end{pmatrix} \quad (6)$$

Abschließend kann der letzte verbleibende Prozess  $P_1$  ausgeführt werden. Demnach gibt es keine Deadlock und es gilt:

$$\mathbf{A} = (4 \quad 2 \quad 3 \quad 1) \quad (7)$$

- a) Nach dem letzten Schritt des Bankieralgorithmus im obigen Beispiel ist  $A = E$ . Begründen Sie!

- b) Wir betrachten ein System mit vier Prozessen, die um *je drei* Festplatten, Webcams, CD-Laufwerke und Monitor konkurrieren. Zum Betrachtungszeitpunkt belegt jeder Prozess je Ressource nur ein Gerät.  $P_1$  belegt Festplatte und Monitor;  $P_2$  Webcam, CD-Laufwerk und Monitor;  $P_3$  Webcam und Monitor und  $P_4$

nur eine Webcam. Geben Sie  $n$ ,  $m$ , die Vektoren  $E$  und  $A$  sowie die Matrize  $C$  an.

c) Zusätzlich ist

$$\mathbf{R} = \begin{array}{cccc|l} & \text{Festplatte} & \text{Webcam} & \text{CD-Laufwerk} & \text{Monitor} & \\ \left[ \begin{array}{cccc} 2 & 0 & 1 & 1 \\ 3 & 2 & 1 & 2 \\ 1 & 1 & 2 & 0 \\ 1 & 0 & 1 & 0 \end{array} \right] & P_1 & P_2 & P_3 & P_4 \end{array} \quad (8)$$

Führen Sie auf den in b) und c) gegebenen Datenstrukturen den Bankieralgorithmus aus. Geben Sie dazu für jede Iteration den Wert von  $A$  an. Interpretieren Sie das Ergebnis des Algorithmus!

d) In diesem Fall ist

$$\mathbf{R} = \begin{bmatrix} \text{Festplatte} & \text{Webcam} & \text{CD-Laufwerk} & \text{Monitor} \\ 2 & 0 & 1 & 1 \\ 3 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \begin{matrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{matrix} \quad (9)$$

Führen Sie auf den in b) und d) gegebenen Datenstrukturen den Bankieralgorithmus aus. Geben Sie dazu für jede Iteration den Wert von  $A$  an. Interpretieren Sie das Ergebnis des Algorithmus!

e) Der Bankieralgorithmus wird von modernen Betriebssystemen *nicht* zur Vermeidung von Deadlocks benutzt. Nennen Sie zwei Gründe dafür.

Besprechung der Lösung am 11.12.18 in der großen Übung.