Betriebssysteme und Sicherheit, WS 2019/20

3. Aufgabenblatt – Verklemmungen

Geplante Bearbeitungszeit: eine Woche

Aufgabe 3.1 Gegeben sei das folgende System paralleler Threads; `s1` bis `s4` bezeichnen dabei binäre Semaphore:

Thread A

```
while (1) {
  s1.down();
  s2.down();
  work();
  s2.up();
  s1.up();
}
```

Thread B

```
while (1) {
  s1.down();
  s3.down();
  work();
  s3.up();
  s1.up();
}
```

Thread C

```
while (1) {
  s4.down();
  s1.down();
  work();
  s1.up();
  s4.up();
}
```

Thread D

```
while (1) {
  s3.down();
  s4.down();
  work();
  s4.up();
  s3.up();
}
```

(a) Wie viele der Threads können sich maximal gleichzeitig im Zustand `work()` befinden?

(b) Was versteht man unter einer Verklemmung (deadlock)? Erläutern Sie mit Bezug auf obiges Beispiel die Bedingungen, die im Zusammenhang mit dem Auftreten von Verklemmungen stehen. Erklären Sie dabei auch den Begriff Betriebsmittel-Zuteilungsgraph (Prozess-Betriebsmittel-Graph).

(c) Ist das System verklemmungsfrei? Verwenden Sie zur Untersuchung der Zyklus-Bedingung einen Betriebsmittel-Zuteilungsgraphen.

(d) Welche Konsequenzen hat jeweils das Vertauschen

- der beiden `down`-Operationen in Thread A,
- der beiden `down`-Operationen in Thread B,
- der beiden `up`-Operationen in Thread B?

(e) Geben Sie einen kurzen Überblick darüber, wie man dem Problem „Verklemmungen“ begegnen kann.

Aufgabe 3.2

(a) Beschreiben Sie den Bank-Algorithmus für Betriebsmittel eines Typs. Welche Bedeutung hat in diesem Zusammenhang der Begriff „sicherer Zustand“?

(b) Betrachten Sie den Bank-Algorithmus in dem folgenden Fall: Vier Prozesse A,...,D bewerben sich um acht Exemplare eines Betriebsmittels. Der Vektor der maximal benötigten Betriebsmittel sei $(6\ 3\ 8\ 4)^{\mathrm{T}}$, der aktuelle Zustand sei $(1\ 2\ 2\ 1)^{\mathrm{T}}$. Untersuchen Sie, inwieweit – jeweils von diesem Zustand ausgehend – die Forderung nach einem Betriebsmittel-Exemplar durch den Prozess A bzw. durch den Prozess C erfüllt werden kann. Diskutieren Sie die Gültigkeit der Aussage: „Der Nachfolgezustand eines sicheren Zustands ist sicher“ und ihrer Umkehrung.

(c) Erläutern Sie Vor- und Nachteile dieser Vorgehensweise.

Aufgabe 3.3 Betrachtet werde ein System mit einem Eingabeprozess, einem Verarbeitungsprozess und einem Ausgabeprozess, die durch zwei Puffer miteinander verbunden sind. Die Prozesse transferieren Daten in Einheiten gleicher Größe. Dabei entnimmt der Verarbeitungsprozess eine Seite aus dem Puffer (gibt diesen Bereich danach frei), speichert den Seiteninhalt in einem lokalen Puffer und füllt dann den Ausgabepuffer. Die Grenze zwischen Eingabepuffer und Ausgabepuffer ist gleitend in Abhängigkeit von der Geschwindigkeit der Prozesse, einzige Beschränkung ist die Gesamtkapazität des Speichers, in dem die Puffer angelegt sind. Jeder der drei Prozesse arbeitet, solange Eingabedaten für ihn vorhanden sind und Speicherplatz in dem Puffer, den er benötigt, zur Verfügung steht; andernfalls wartet er.

(a) Zeigen Sie, dass dieses System verklemmen kann.

(b) Schlagen Sie eine Lösung zur Vermeidung von Verklemmungen vor unter Beibehaltung der gleitenden Grenze zwischen den beiden Puffern.

Aufgabe 3.4 Diskutieren Sie den folgenden Algorithmus unter dem Gesichtspunkt „Maßnahmen gegen Verklemmungen“:

I. Die Menge der Betriebsmittel (BM) wird in m Klassen unterteilt. Jede Klasse besteht aus mindestens soviel BM, dass jeder Prozess einzeln fortschreiten kann.

II. Die von einem Prozess benötigten BM aus einer Klasse müssen auf einmal zugeteilt werden.

III. Wenn ein Prozess BM der Klasse i zugeteilt bekommen hat, kann er nur noch BM aus einer höheren Klasse j ($j > i$) anfordern.

IV. Die BM der höheren Klassen müssen von jedem Prozess vor den BM der niedrigeren Klasse freigegeben werden.

V. Erst wenn ein Prozess alle BM, die er aus einer gegebenen Klasse erhalten hatte, wieder freigegeben hat, kann er neue BM aus dieser Klasse anfordern.

Klausuraufgabe I

Zwei Threads rufen parallel die nebenstehende Funktion auf. Nennen Sie Aufrufe von `f`, die zu einer Verklemmung (*deadlock*) führen können. Jeder Knoten besitzt dabei einen binären Semaphore, der mit `down()` angefordert und `up()` freigegeben wird.

```
1 void f(Knoten *k1, Knoten *k2) {
2    k1->down();
3    k2->down();
4    // Zugriff auf k1 und k2
5    k2->up();
6    k1->up();
7 }
```

Klausuraufgabe II

In einem System stehen insgesamt 7 Bandlaufwerke bereit. Diese werden von verschiedenen laufenden Prozessen (A,B,C,D) für die Langzeitarchivierung von Daten genutzt. Jeder Prozess benötigt dabei ein Bandlaufwerk während seiner gesamten Laufzeit. Je nach zu verarbeitendem Datenaufkommen sind zu verschiedenen Zeitpunkten aber noch zusätzliche Laufwerke nötig. Es spielt dabei keine Rolle welches Laufwerk ein Prozess zugewiesen bekommt; das Schreiben muss allerdings in einem Stück geschehen, da andernfalls Daten verloren gehen. Um dies zu verhindern, wird jedes Bandlaufwerk immer genau einem Prozess exklusiv zur Verwendung zugewiesen bis dieser es ans System zurückgibt.

Die Prozesse benötigen maximal 4 (A), 2 (B), 6 (C) bzw. 5 (D) Bandlaufwerke gleichzeitig. Um eine möglichst hohe Auslastung der verfügbaren Laufwerke zu gewährleisten, soll ein Laufwerk einem Prozess nur dann bereitgestellt werden, wenn und solange er dieses benötigt.

a) Zeigen Sie, dass es im beschriebenen System zu Verklemmungen kommen kann!

b) Um Verklemmungen vorzubeugen wird der Bank-Algorithmus eingesetzt. Aktuell sind den Prozessen 1 (A), 0 (B), 1 (C) und 3 (D) Laufwerke zugeordnet. Nacheinander fordern nun die Prozesse C und D je ein zusätzliches Laufwerk an. Wie reagiert das System auf diese Anforderungen? Wenden Sie zur Beantwortung der Frage den Bank-Algorithmus an!

c) Kann es im Rahmen des Bank-Algorithmus dazu kommen, dass der Vorgänger eines sicheren Zustandes ein unsicherer Zustand ist? Begründen Sie Ihre Antwort!

d) Nennen Sie eine Alternative zur Verwendung des Bank-Algorithmus sowie je einen Vor- und Nachteil dieser Alternative gegenüber dem Bank-Algorithmus!