Arbeitsgruppe Autonome Intelligente Systeme

Freiburg, 20. Dezember 2016

Systeme I: Betriebssysteme
Übungsblatt 9

Aufgabe 1 (2 Punkte)

In der Vorlesung wurde die Aussage getroffen, dass ein nach Definition des Bankier-Algorithmus „unsicherer“ Zustand nicht notwendigerweise zu einem Deadlock führt.

Begründen Sie diese Aussage und geben Sie ein Beispiel mit zwei Prozessen an, bei dem ein unsicherer Zustand zu keinem Deadlock führt. Verwenden Sie dafür Pseudocode und spezifizieren Sie die Ausführungsreihenfolge.

Aufgabe 2 (2+2+1 Punkte)

Ein System besitzt mehrere (V) identische Ressourcen (z.B. Drucker) sowie mehrere Prozesse, die jeweils maximal M dieser Ressourcen benötigen. Da die Ressourcen identisch sind, ist es egal, welche Ressource welchem Prozess zugeordnet wird.

a) Wir nehmen an, dass das System drei identische Ressourcen besitzt ($V = 3$). Jeder laufende Prozess braucht maximal zwei dieser Ressourcen ($M = 2$). Ist in diesem Szenario ein Deadlock möglich, falls auf dem System

   1) gleichzeitig 2 solcher Prozesse laufen?
   2) gleichzeitig 3 solcher Prozesse laufen?

   Begründen Sie Ihre Antworten.

b) Das Problem aus a) soll nun verallgemeinert werden auf n Prozesse und V identische Ressourcen, von denen jeder Prozess maximal M benötigt. Gesucht ist eine Bedingung, welche die *Minimal*anforderungen beschreibt. Seien n und M gegeben. Wie groß muss die Anzahl V der Ressourcen mindestens sein, damit garantiert bei keiner Ausführungsreihenfolge ein Deadlock auftritt? Begründen Sie Ihre Antwort.

c) Das System aus a2) mit drei Prozessen (P1, P2, P3) sei in einem Zustand, in dem P1 und P2 jeweils eine Ressource halten und P3 keine. Ist das ein sicherer Zustand?

   Hinweis: Für die Definition siehe die Vorlesungs-Folie Kap. 6, „Bankier-Algorithmus (1)“.

Hinweis: Bei den Aufgaben a) und b) geht es um Ausführungsreihenfolgen, bei denen ein Prozess immer den Zugriff auf eine Ressource erhält, falls er eine anfordert und diese gleichzeitig verfügbar ist. Bei Aufgabe c) ist zu prüfen, ob ein Zustand sicher ist. Einer solchen Prüfung liegt der Gedanke zugrunde, dass Prozesswechsel auch bei verfügbaren Ressourcen durch das Betriebssystem vorgegeben werden können.

Aufgabe 3 (1+2+2,5 Punkte)

Ein Verkäufer einer kleinen Bäckereifiliale kommt zu spät zur Arbeit. Vor dem Geschäft warten schon fünf Kunden $C_1, \ldots, C_5$. Diese sind sich allerdings uneinig darüber, wer zuerst da war. Die Kunden haben jeweils einen Einkaufszettel in der Hand, anhand dessen Länge der Verkäufer die Zeit Z_i abschätzen kann, die der Verkaufsvorgang für Kunde C_i in Anspruch nehmen wird. Es gelte $Z_1 = 5\,\text{min}, Z_2 = 3\,\text{min}, Z_3 = 7\,\text{min}, Z_4 = 6\,\text{min}$ und $Z_5 = 1\,\text{min}$. Alle Kunden möchten das Geschäft möglichst schnell mit allen ihren Einkäufen verlassen. Nehmen Sie an, dass keine neuen Kunden das Geschäft betreten und dass der Verkäufer einen Kunden immer fertig bedient, bevor er mit dem nächsten Kunden anfängt.

a) Um möglichst wenig Kunden zu verärgern, möchte der Verkäufer die durchschnittliche Aufenthaltszeit der Kunden in der Bäckerei minimal halten.

   1. Welche Reihenfolge würden Sie dem Verkäufer empfehlen und warum?
   2. Geben Sie an, nach welcher Zeit alle Kunden bedient sind und die Bäckerei verlassen haben.
   3. Berechnen Sie die durchschnittliche Aufenthaltszeit der Kunden in der Bäckerei.

Um die Kunden schneller bedienen zu können, holt der Verkäufer den Azubi der Bäckerei hinzu, sodass sich die beiden Angestellten nun die Kunden untereinander aufteilen können. Der Azubi ist jedoch der Meinung, dass sie die Kunden so aufteilen sollten, dass die benötigte Zeit zum Abarbeiten aller Kunden minimal wird, damit die beiden Verkäufer möglichst bald Pause machen können.

b) Nehmen Sie an, dass sich die beiden Verkäufer für die Strategie entscheiden, jeweils den noch nicht bedienten Kunden mit dem kürzesten Einkaufszettel als nächstes an die Reihe zu nehmen, wobei der Azubi stets beginnt.

   1. Erstellen Sie einen Zeitplan in Form einer Skizze oder Tabelle, aus der hervorgeht, welcher Verkäufer wann welchen Kunden bedient.
   2. Geben Sie an, nach welcher Zeit alle Kunden bedient sind und die Bäckerei verlassen haben.
   3. Berechnen Sie die durchschnittliche Aufenthaltszeit der Kunden in der Bäckerei.

c) Nehmen Sie nun an, dass sich die Verkäufer stattdessen für die Strategie entscheiden, jeweils den Kunden mit dem *längsten* statt dem kürzesten Einkaufszettel als nächstes bedienen und wiederholen Sie die drei Schritte aus b).

   Welche der beiden Strategien ist aus Sicht des Azubis demnach besser und welche aus Sicht der Kunden?

Aufgabe 4 (2 Punkte)

In der Vorlesung wurde gezeigt, dass die Scheduling-Strategie „Shortest Job First“ (SJF) optimal (im Sinne von minimaler mittlerer Durchlaufzeit) ist, wenn alle Prozesse gleichzeitig verfügbar sind. Dies gilt im Allgemeinen nicht mehr, wenn nicht alle Prozesse gleichzeitig verfügbar werden.

Zeigen Sie dies, indem Sie ein Gegenbeispiel konstruieren. Definieren Sie dazu eine Menge von Prozessen mit Lauf- und Ankunftzeiten und geben Sie einen nicht-präemptiven Ausführungsplan an, dessen mittlere Durchlaufzeit geringer ist als die mittlere Durchlaufzeit des mittels SJF erzeugten Planes. Berechnen Sie für beide Ausführungsreihenfolgen die mittlere Durchlaufzeit.

Hinweis: Die Ankunftzeit ist der Zeitpunkt, ab dem ein Prozess verfügbar ist. Die Durchlaufzeit ist die Differenz zwischen dem Zeitpunkt der Beendigung eines Prozesses und seiner Ankunftzeit.

Aufgabe 5 (1,5+0,5 Punkte)

Laden Sie von `http://ais.informatik.uni-freiburg.de/teaching/ws16/systems1/exercises/starte_prozesse.sh` ein Shell-Skript herunter, setzen Sie das Ausführungsrecht und führen Sie das Skript aus. Das Skript startet zwei Prozesse des Programms `dd`, die viel Rechenzeit verbrauchen ohne etwas Sinnvolles zu tun. Das Skript gibt außerdem die Prozess-IDs (PID) der beiden gestarteten Prozesse aus. Mit `Strg+C` können Sie das Skript und die beiden Prozesse beenden.

Beobachten Sie die beiden Prozesse in einem anderen Terminal mit dem Befehl `top`. Über eine passende Kommandozeilenoption können Sie angeben, welche Prozesse angezeigt werden sollen. Details dazu, zur Bedeutung der angezeigten Werte und zur Steuerung des Programms über Tastenkombinationen finden Sie in den `manpages` von `uptime` und `top` sowie in der eingebauten Hilfe des Programms, die Sie über die Taste `h` öffnen können.

a) Der CPU-Anteil der beiden durch das Skript gestarteten Prozesse sollte anfangs etwa gleich hoch sein. Verwenden Sie den Befehl `renice`, um zu erreichen, dass einer der beiden Prozesse deutlich mehr CPU-Anteil zugewiesen bekommt als der andere Prozess. Beachten Sie, dass Sie als normaler Benutzer ohne root-Rechte nicht beliebige Werte setzen können. Bitte beschreiben Sie für die Abgabe in Stichworten,

   - welche Befehle Sie mit welchen Parametern ausgeführt haben,
   - was die Parameter bedeuten, welche Werte dafür zulässig sind und was sie bewirken,
   - welchen Effekt die Ausführung der Befehle auf die CPU-Anteile der Prozesse haben.

b) Begründen Sie, warum die *nice*-Werte der Prozesse keinen Einfluss auf den Wert des Load Average haben.

Je nach System (Anzahl der CPU-Kerne usw.) können Sie die Effekte besser beobachten, wenn Sie die Anzahl der zu startenden Prozesse erhöhen, z.B. mit `./starte_prozesse.sh 8` für acht zu startende Prozesse.

Aufgabe 6 (0,5+0,5+0,5+0,5+0,5 Punkte)

Beschreiben Sie in eigenen Worten, was die Ausführung nachfolgender Befehle bewirkt. Verwenden Sie `man`, um sich zu informieren.

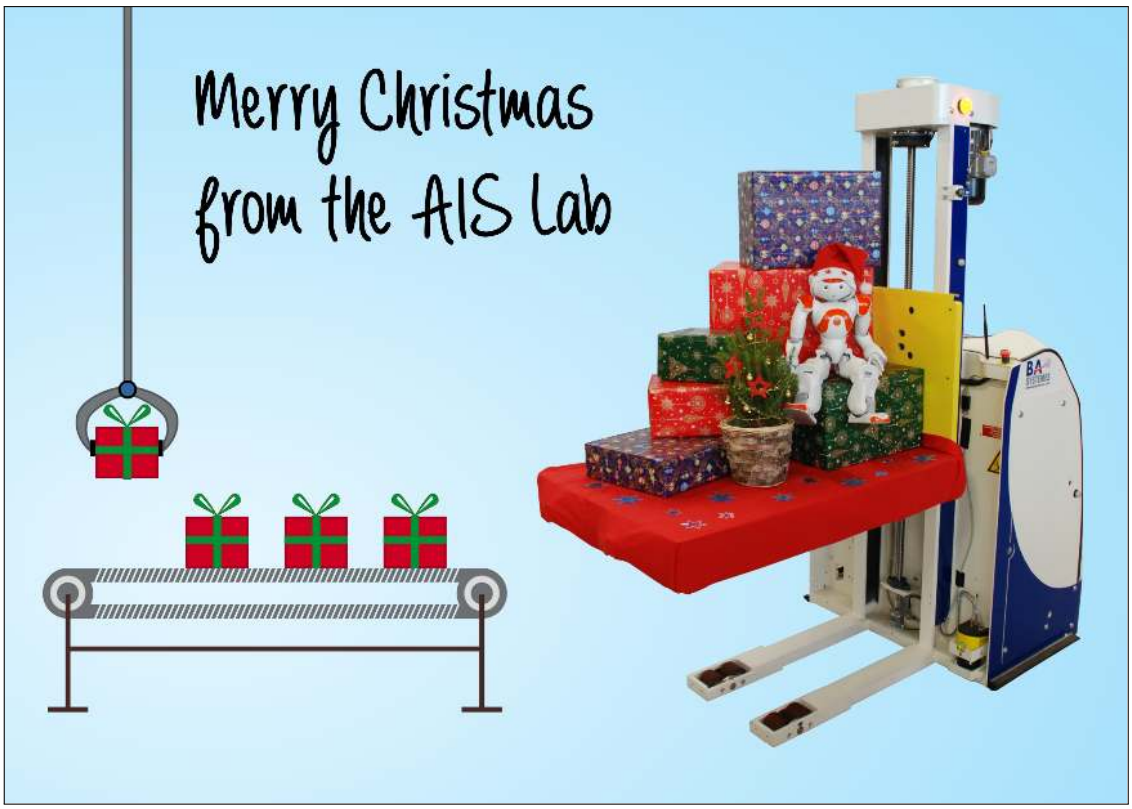
a) `wc -l /etc/passwd`

b) `grep 'Florian' /etc/passwd`

c) `cut -f1 -d ':' /etc/passwd`

d) `find . -type f`

e) `du -s .`



Abgabe: Als PDF-Datei im Ilias bis zum 09. Januar 2017, 23:59 Uhr