Ludwig-Maximilians-Universität München
Institut für Informatik
Lehrstuhl für Mobile und Verteilte Systeme
Prof. Dr. Claudia Linnhoff-Popien

mobile and distributed systems group

# Betriebssysteme im Wintersemester 2019/2020
# Übungsblatt 3

**Abgabetermin:** 11.11.2019, 18:00 Uhr

**Besprechung:** Besprechung der T-Aufgaben in den Tutorien vom 04. – 08. November 2019
Besprechung der H-Aufgaben in den Tutorien vom 11. – 15. November 2019

## Aufgabe 12: (T) 5-Zustands-Prozessmodell

(– Pkt.)

a. Geben Sie für jeden der folgenden Zustandsübergänge im 5-Zustands-Prozessmodell an, ob der Übergang zulässig ist und auf welche Weise der Übergang stattfindet oder warum kein solcher Übergang möglich ist:

   (i) Ändern des Prozesszustandes von „Blocked“ zu „Running“.

   (ii) Ändern des Prozesszustandes von „Running“ zu „Blocked“.

   (iii) Ändern des Prozesszustandes von „Ready“ zu „Blocked“.

b. Wieder ausgehend vom 5-Zustands-Prozessmodell:

   (i) Geben Sie für jeden Zustandsübergang ein Beispiel an und beschreiben Sie anhand dieses Beispiels, wie dort der jeweilige Zustandswechsel ausgelöst werden könnte.

   (ii) Wie ändert sich das Modell für ein Zwei-Prozessor-System?

## Aufgabe 13: (T) Prozessstatus unter Linux

(– Pkt.)

Um auf einem Linux-System einen Überblick über die parallel laufenden Prozesse zu erhalten, kann man den Aufruf `ps` nutzen. Machen Sie sich zunächst mit dem `ps`-Befehl vertraut, indem Sie die zugehörige Anleitung (Manual-Seite/Manpage) öffnen. Geben Sie dazu in der Shell den Befehl `man ps` ein.
Beantworten Sie nun folgenden Aufgaben:

a. Aus welchem Verzeichnis stammen die Informationen, die das Kommando `ps` generiert?

b. Was ist das Besondere an diesem Verzeichnis? Erläutern Sie kurz, wie die Inhalte (Verzeichnisse und Dateien) innerhalb dieses Verzeichnisses generiert werden.

c. Welchen Sinn hat dieses Verzeichnis für System- und Nutzerprogramme?

d. Erklären Sie kurz die Bedeutung der folgenden Aufrufparameter (Flags):
(i) `-A` (ii) `-l` (iii) `-p 2475`

e. Nennen Sie vier Elemente/Bestandteile des Prozesskontrollblocks (PCB), die bei einem Aufruf von `ps -l` für jeden gelisteten Prozess angezeigt werden.

f. Welche beiden Prozesse sind bei einer Ausführung von `ps` ohne selektierende Parameter immer in der Ausgabe enthalten? Wovon hängt ab, ob noch weitere Prozesse angezeigt werden?

g. Woraus lässt sich bei Betrachtung der beiden Prozesse aus Teilaufgabe f) ableiten, dass Prozesse unter Linux hierarchisch strukturiert sind?

h. Führen Sie den Befehl `ps -l` mehrmals hintereinander aus. Bei welchen Werten treten Änderungen ein? Wie lässt sich dies erklären? Welche Werte ändern sich nie?

i. Wie heißt unter Linux der Wurzelprozess des Prozessbaums? Wie lautet die Prozess-ID des Wurzelprozesses? Welcher Wert steht bei bei diesem Prozess im Feld der PPID?

j. Terminiert der Vater eines laufenden Prozesses, so wird dieser Prozess zu einem Waisen. Wie wird ein Waisenprozess in die Prozesshierarchie neu eingeordnet, und was für eine PPID erhält er somit?

## Aufgabe 14: (T) Nicht-preemptives Scheduling

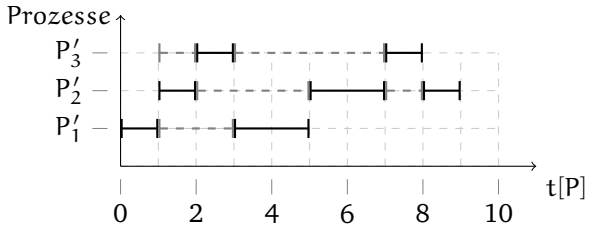
(– Pkt.)

In dieser Aufgabe sollen zwei Scheduling-Strategien untersucht werden: die nicht-preemptive Strategie FCFS (First Come First Served) und die nicht-preemptive Strategie SJF (Shortest Job First). Dazu seien die Prozesse auf der nächsten Seite (nach dem Beispiel) mit ihren Ankunftszeitpunkten und Bedienzeiten (in beliebigen Zeiteinheiten) gegeben.

**Beispiel:** Es seien folgende Ankunfts– und Bedienzeiten für die drei Beispielprozesse $P'_1$, $P'_2$ und $P'_3$ gegeben:

| Prozess | Ankunftszeitpunkt | Bedienzeit |
|---|---|---|
| $P'_1$ | 0 | 3 |
| $P'_2$ | 1 | 4 |
| $P'_3$ | 1 | 2 |

Das folgende Diagramm zeigt ein zufälliges Scheduling der drei Prozesse $P'_1$, $P'_2$ und $P'_3$ und soll die Form der Darstellung veranschaulichen:




– Trifft ein Prozess zum Zeitpunkt $t$ ein, so wird er direkt zum Zeitpunkt $t$ berücksichtigt.

– Wird ein Prozess zum Zeitpunkt $t'$ unterbrochen, so reiht er sich auch zum Zeitpunkt $t'$ wieder in die Warteschlange ein.

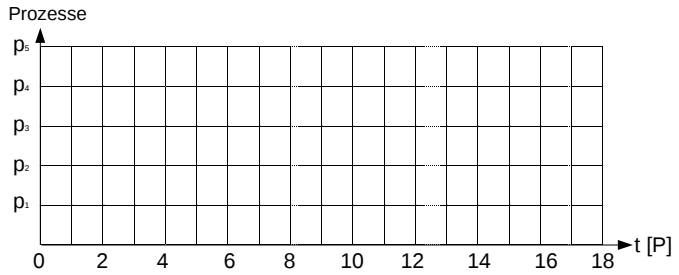
– Sind zwei Prozesse absolut identisch bezüglich ihrer relevanten Werte, so werden die Prozesse nach aufsteigender Prozess-ID in der Warteschlange eingereiht (Prozess $P_i$ vor Prozess $P_{i+1}$, usw.). Diese Annahme gilt sowohl für neu im System eintreffende Prozesse, als auch für den Prozess, dem der Prozessor u.U. gerade entzogen wird!

– Jeder Prozess nutzt sein Zeitquantum stets vollständig aus d.h. kein Prozess gibt den Prozessor freiwillig frei (Ausnahme: bei Prozessende).

| Prozess | Ankunftszeitpunkt | Bedienzeit |
|---|---|---|
| $P_1$ | 0 | 2 |
| $P_2$ | 2 | 4 |
| $P_3$ | 2 | 7 |
| $P_4$ | 4 | 3 |
| $P_5$ | 5 | 2 |

Bearbeiten Sie unter den gegebenen Voraussetzungen nun die folgenden Aufgaben:

a. Verwenden Sie nun die **nicht-preemptive Strategie FCFS** und erstellen Sie entsprechend dem vorherigen Beispiel ein Diagramm, das für die Prozesse $P_1$–$P_5$ angibt, wann welchem Prozess Rechenzeit zugeteilt wird und wann die Prozesse jeweils terminieren. Kennzeichnen Sie zudem für jeden Prozess seine Ankunftszeit. Erstellen Sie Ihre Lösung basierend auf folgender Vorlage und verwenden Sie die oben beispielhaft gezeigte Darstellungsform:




b. Verwenden Sie nun die **nicht-preemptive Strategie SJF** und stellen Sie Ihre Lösung, wie in der vorherigen Teilaufgabe a), dar.

c. Berechnen Sie als Dezimalzahl mit einer Nachkommastelle die mittlere Verweil- und Wartezeit für die zwei Verfahren FCFS und SJF.

d. Welchen Nachteil hat SJF in Bezug auf die Verweildauer von Prozessen?

## Aufgabe 15: (H) Prozesszustände

(10 Pkt.)

a. Nennen Sie die Zustände, welche beim 5–Zustands–Prozessmodell im Vergleich zum 2–Zustands–Prozessmodell hinzukommen und erläutern Sie den Nutzen dieser.

b. Diskutieren Sie, wie sich das Hinzufügen von Prozesszuständen in ein Prozessmodell auf die *Prozessverwaltung* auswirkt. Nennen Sie je einen Vorteil bzw. Nachteil, der sich ergibt, wenn ein Prozessmodell zwischen vielen verschiedenen Prozesszuständen differenziert.

c. Erklären Sie kurz die Begriffe *Scheduling* und *Dispatching*.

## Aufgabe 16: (H) Nicht-preemptives Scheduling

(12 Pkt.)

In dieser Aufgabe sollen zwei Scheduling-Strategien untersucht werden: die nicht-preemptive Strategie FCFS (First Come First Served) und die nicht-preemptive Strategie SJF (Shortest Job First). Dazu seien die folgenden Prozesse mit ihren Ankunftszeitpunkten und Rechenzeiten (in beliebigen Zeiteinheiten) gegeben.

| Prozess | Ankunftszeitpunkt | Rechenzeit |
|---|---|---|
| $P_1$ | 0 | 6 |
| $P_2$ | 1 | 2 |
| $P_3$ | 3 | 1 |
| $P_4$ | 4 | 4 |
| $P_5$ | 2 | 1 |

**Gehen Sie von den gleichen Voraussetzungen und der gleichen Darstellungsform für das Scheduling wie in Aufgabe 14) aus.**

a. Erstellen Sie entsprechend des Beispiels in Aufgabe 14) ein Diagramm für die **nicht-preemptive Strategie FCFS**, das für die Prozesse $P_1$–$P_5$ angibt, wann welchem Prozess Rechenzeit zugeteilt wird und wann die Prozesse jeweils terminieren. Kennzeichnen Sie zudem für jeden Prozess seine Ankunftszeit.

b. Erstellen Sie entsprechend des Beispiels in Aufgabe 14) ein Diagramm für die **nicht-preemptive Strategie SJF**, das für die Prozesse $P_1$–$P_5$ angibt, wann welchem Prozess Rechenzeit zugeteilt wird und wann die Prozesse jeweils terminieren. Kennzeichnen Sie zudem für jeden Prozess seine Ankunftszeit.

c. Berechnen Sie als Dezimalzahl mit einer Nachkommastelle die mittlere Verweil- und Wartezeit für die zwei Verfahren FCFS und SJF.

## Aufgabe 17: (H) Einfachauswahlaufgabe: Prozesse

(5 Pkt.)

Für jede der folgenden Fragen ist eine korrekte Antwort auszuwählen („1 aus n“). Nennen Sie dazu in Ihrer Abgabe explizit die jeweils ausgewählte Antwortnummer ((i), (ii), (iii) oder (iv)). Eine korrekte Antwort ergibt jeweils einen Punkt. Mehrfache Antworten oder eine falsche Antwort werden mit 0 Punkten bewertet.

<table>
<tr><td colspan="4">a) Wie bezeichnet man die Informationen, die den aktuellen Ausführungszustand eines Prozesses genau beschreiben (insb. die CPU-Register-Belegungen und alle Prozess-Status-Informationen)?</td></tr>
<tr><td>(i) (Prozess-)Kontext</td><td>(ii) (Prozess-) Quellcode</td><td>(iii) (Prozess-) Spiegel</td><td>(iv) (Prozess-) Rahmen</td></tr>
<tr><td colspan="4">b) Wie bezeichnet man die sequenzielle, vollständige und unterbrechungsfreie Ausführung von Prozessen?</td></tr>
<tr><td>(i) Broadprogram-ming</td><td>(ii) Uniprogram-ming</td><td>(iii) Multi-programming</td><td>(iv) Multi-processoring</td></tr>
<tr><td colspan="4">c) Welche Aussage bezüglich des Stacks (Stapelspeicher) ist falsch?</td></tr>
<tr><td colspan="4">(i) Er kann zur Übergabe von Parametern bzw. Rückgabewert zwischen Haupt- und Unterprogrammen verwendet werden.<br>(ii) Er besitzt eine <code>PUSH</code> Direktive.<br>(iii) Er arbeitet nach dem FIFO Prinzip.<br>(iv) Er besitzt eine <code>POP</code> Direktive.</td></tr>
<tr><td colspan="4">d) Mit welchem Systemaufruf werden unter Unix/Linux Systemen neue Prozesse erzeugt?</td></tr>
<tr><td>(i) <code>fork</code></td><td>(ii) <code>creat</code></td><td>(iii) <code>execl</code></td><td>(iv) <code>getpid</code></td></tr>
<tr><td colspan="4">e) Wie heißt der Prozess, der einen in Bearbeitung befindlichen Prozess unterbrechen und dem Prozessor einen anderen Prozess zuweisen kann?</td></tr>
<tr><td>(i) Scheduler</td><td>(ii) Swapper</td><td>(iii) Blocker</td><td>(iv) Dispatcher</td></tr>
</table>