

## Betriebssysteme im Wintersemester 2017/2018

### Übungsblatt 10

**Abgabetermin:** 15.01.2018, 18:00 Uhr

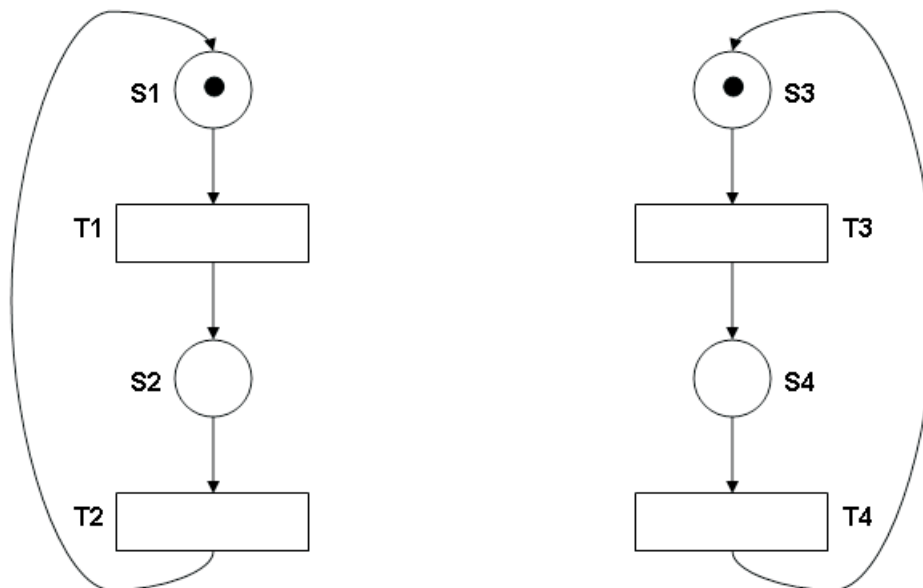
**Besprechung:** Besprechung der T-Aufgaben in den Tutorien vom 08. – 12. Januar 2018  
 Besprechung der H-Aufgaben in den Tutorien vom 15. – 19. Januar 2018

#### Aufgabe 43: (T) Deadlockerkennung

(– Pkt.)

Wenn Prozesse nebenläufig ausgeführt werden, macht das eine geeignete Synchronisation dieser Prozesse erforderlich.

Beispielhaft für **zwei** Prozesse  $P_1$  und  $P_2$  soll dieses Synchronisationsproblem durch ein Petrinetz modelliert werden. Die Stellen können als Zustände interpretiert werden. Dabei entspricht  $S_1$  ( $S_3$ ) dem Zustand "P<sub>1</sub> ( $P_2$ ) rechnet im unkritischen Bereich", und  $S_2$  ( $S_4$ ) dem Zustand "P<sub>1</sub> ( $P_2$ ) rechnet im kritischen Bereich".



- Ergänzen Sie dieses Petrinetz um weitere Stellen, Transitionen und Kanten (sofern jeweils erforderlich), sodass zu jedem Zeitpunkt höchstens ein Prozess in seinem kritischen Bereich rechnen kann. Zeichnen Sie direkt in das obige Petrinetz.
- Geben Sie den Erreichbarkeitsgraphen für Ihr Petrinetz an, und beschriften Sie darin alle Übergänge mit den Namen der jeweiligen Transitionen. Begründen Sie, ob das Petrinetz verklemmt ist oder nicht.

## Aufgabe 44: (T) SJF versus SRPT

(– Pkt.)

In dieser Aufgabe sollen zwei Scheduling-Strategien untersucht werden: die nicht-preemptive Strategie SJF (Shortest Job First) und die preemptive Strategie SRPT (Shortest Remaining Processing Time). Dazu seien die folgenden Prozesse mit ihren Ankunftszeitpunkten und Bedienzeiten (in beliebigen Zeiteinheiten) gegeben.

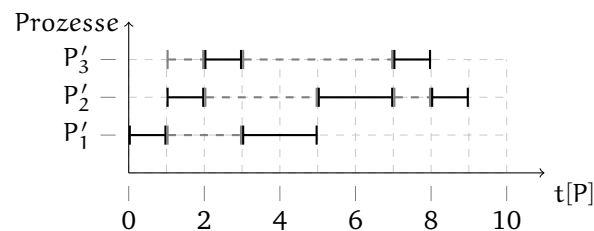
Prozess	Ankunftszeitpunkt	Bedienzeit
P <sub>1</sub>	0	6
P <sub>2</sub>	2	4
P <sub>3</sub>	2	2
P <sub>4</sub>	4	1
P <sub>5</sub>	8	7
P <sub>6</sub>	9	3

- 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).

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

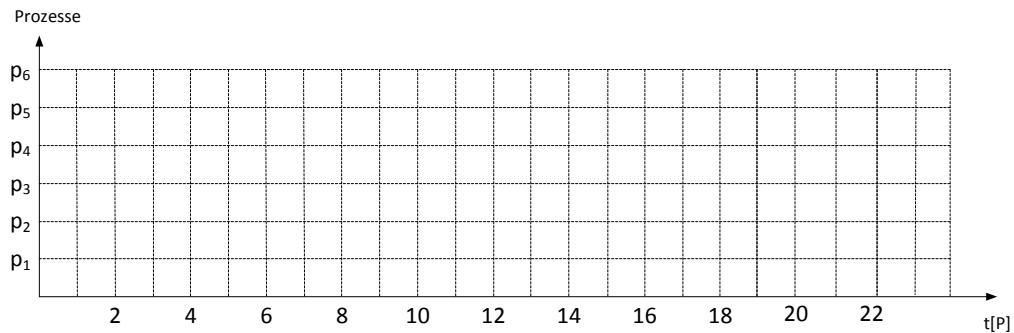
Prozess	Ankunftszeitpunkt	Bedienzeit
P' <sub>1</sub>	0	3
P' <sub>2</sub>	1	4
P' <sub>3</sub>	1	2

Das folgende Diagramm veranschaulicht ein beliebiges Scheduling der drei Prozesse  $P'_1$ ,  $P'_2$  und  $P'_3$ :



Bearbeiten Sie unter den gegebenen Voraussetzungen nun die folgenden Aufgaben:

- Verwenden Sie nun die **nicht-präemptive Strategie SJF** und erstellen Sie entsprechend dem vorherigen Beispiel ein Diagramm, das für die Prozesse  $P_1$ – $P_6$  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:



- b. Verwenden Sie nun die **preemptive Strategie SRPT** 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 SJF und SRPT.
- d. Wodurch lässt sich der Unterschied zwischen SJF und SRPT in Bezug auf die mittlere Wartezeit begründen?

### Aufgabe 45: (H) Petrinetze und Erreichbarkeitsgraphen

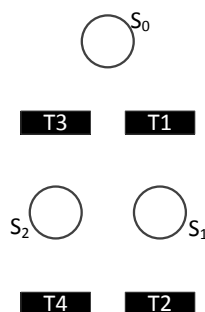
(14 Pkt.)

Gegeben sei ein Rechner mit einer CPU und zwei Prozessen  $P_1$  und  $P_2$ , die mit preemptivem Multitasking auf der CPU ausgeführt werden. Wird zwischen den Prozessen gewechselt, so wird der aktive Prozess suspendiert und das Betriebssystem  $BS$  aktiv. Dieses führt mittels Scheduler und Dispatcher den Wechsel zwischen den Prozessen durch. Der ausgewählte Prozess wird der CPU durch das Dispatchen zum Rechnen zugewiesen. Gehen Sie zunächst von folgenden Bedingungen aus:

- Auf der CPU kann zu jedem Zeitpunkt immer nur genau ein Prozess gleichzeitig rechnen.
- Die Betriebssystemfunktionen sind ebenfalls als eigener Prozess zu beachten.
- Zu Beginn ist das Betriebssystem aktiv.

Bearbeiten Sie dazu die folgenden Teilaufgaben:

- a. Im Folgenden soll das beschriebene System mit einem Petrinetz modelliert werden. Die Markierung der Stellen kann dabei beschreiben, welcher der Prozesse aktuell der CPU zugeordnet ist. Ergänzen Sie nun das nachfolgend angedeutete Petrinetz um eine minimale Anzahl an weiteren Marken, Stellen, Transitionen und Kanten (insofern erforderlich), so dass die oben genannten Bedingungen erfüllt sind. Erklären Sie für alle Stellen und Transitionen jeweils kurz deren Bedeutung.



- b. Erweitern Sie nun das in Aufgabenteil a) angedeutete Petrinetz um eine minimale Anzahl an weiteren Marken, Stellen, Transitionen und Kanten (insofern erforderlich), so dass das Verhalten von Round-Robin-Scheduling modelliert wird. Hierbei soll  $P_1$  der erste Prozess sein, welcher der CPU vom Betriebssystem zum Rechnen zugewiesen wird. Geben Sie allen neu eingefügten Transitionen oder Stellen einen eindeutigen Bezeichner und erklären Sie kurz deren Bedeutung.
- c. Geben Sie den Erreichbarkeitsgraphen zu dem Petrinetz aus Teilaufgabe b) an. Geben Sie hierbei auch an, wie in den Markierungen des Erreichbarkeitsgraphen die Stellen angeordnet sind. Beschriften Sie alle Übergänge zwischen Markierungen mit der Bezeichnung der Transition, die hierfür feuern muss.
- d. Kann einer der beiden Prozesse  $P_1$  oder  $P_2$  verhungern? Begründen Sie ihre Aussage mithilfe des Erreichbarkeitsgraphen zu dem Petrinetz aus Teilaufgabe b).
- e. Begründen Sie mithilfe des Erreichbarkeitsgraphen zu dem Petrinetz aus Teilaufgabe b), ob das System frei von Verklemmungen bzw. teilweisen Verklemmungen ist.

### Aufgabe 46: (H) Einfachauswahlaufgabe: Prozesskoordination

(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.

a) Welche Art von Prozessen macht die Synchronisation der Prozesse untereinander erforderlich?			
(i) unabhängige Prozesse	(ii) nebenläufige Prozesse	(iii) sequenzielle Prozesse	(iv) parallele aber unabhängige Prozesse
b) Wie bezeichnet man einen Speicher, der in Form eines eindimensionalen Arrays unter Verwendung der Modulo-Funktion realisiert wird?			
(i) Sparpuffer	(ii) Linearpuffer	(iii) Wechsellpuffer	(iv) Ringpuffer
c) Wie bezeichnet man die Phase, in der sich zu einem Zeitpunkt nur ein Prozess befinden darf, da sich sonst z.B. inkonsistente Zustände bei gemeinsam genutzten Datenstrukturen ergeben?			
(i) einfacher Bereich	(ii) schwieriger Bereich	(iii) kritischer Bereich	(iv) unkritischer Bereich
d) Wie berechnet sich allgemein die Verweildauer?			
(i) Verweildauer = Bedienzeit + Wartezeit	(ii) Verweildauer = Beendigungszeit + Wartezeit	(iii) Verweildauer = Startzeit + Wartezeit	(iv) Verweildauer = Ankunftszeit + Wartezeit
e) Wie bezeichnet man den nicht-preemptiven Scheduling-Algorithmus, bei welchem jeweils der Auftrag ausgewählt wird, bei dem die kürzeste Abarbeitungszeit erwartet wird?			
(i) First Come First Served	(ii) Shortest Remaining Processing Time	(iii) Shortest Job First	(iv) Round Robin