

## Aufgabe 1: Scheduling-Algorithmen

(11 Punkte)

Gegeben seien die fünf nicht-periodischen Prozesse ohne I/O-Operationen aus Tabelle 1, die auf einem Einprozessor-System ausgeführt werden sollen. Nehmen Sie für diese Aufgabe an, dass der Prozesswechsel eine Zeiteinheit benötigt und der Prozessor für diese Zeit keine andere Operation ausführen kann.

Prozess	$P_1$	$P_2$	$P_3$	$P_4$	$P_5$
Ankunftszeit ( $t_i$ )	0	4	5	6	8
Bedienzeitforderung ( $b_i$ )	6	5	1	2	8

Tabelle 1: Ankunftszeiten und Bedienzeitforderungen der Prozesse  $P_1$  bis  $P_5$

- Illustrieren Sie die Ausführungsreihenfolge der fünf Prozesse für die nicht-verdrängende Scheduling-Strategie *beste Bediengüte*. Die Bediengüte  $BG$  ist hierbei definiert als  $BG = \frac{\text{Wartezeit} + \text{Bedienzeit}}{\text{Bedienzeit}}$ . Bei gleicher Bediengüte wird der Prozess mit der kleinsten Bedienzeitanforderung ausgewählt.
- Illustrieren Sie die Ausführungsreihenfolge der fünf Prozesse für die Scheduling-Strategie *Round Robin* mit Zeitscheibengröße  $\Delta t = 2$ . Bei nicht vollständig ausgefüllten Zeitscheiben soll der Prozesswechsel vorgezogen werden, wobei die Prozesswechseldauer zu berücksichtigen ist. Neue Prozesse sollen an das Ende des Warteraums eingestellt werden.
- (optional) Illustrieren Sie die Ausführungsreihenfolge der fünf Prozesse wie in Aufgabe 1b) für die Scheduling-Strategie *Round Robin*, jedoch nun mit der Zeitscheibengröße  $\Delta t = 4$ .
- (optional) Die Antwortzeit eines Prozesses sei als die Zeit zwischen seiner Ankunft und seiner Termination definiert. Berechnen Sie für jeden der drei resultierenden Schedules aus den Aufgaben 1a), 1b) und 1c) die durchschnittliche Antwortzeit der beteiligten Prozesse.
- (optional) Die Größe (Dauer) einer Zeitscheibe beim Scheduling-Algorithmus *Round Robin* wird über einen Parameter  $\Delta t$  festgelegt. Nennen Sie jeweils einen Vor- und Nachteil für kurze bzw. lange Zeitscheiben.

Hinweis: Bitte verwenden Sie zum Illustrieren der Ausführungsreihenfolgen in dieser Aufgabe eine Darstellung in Anlehnung an Abbildung 1. In STiNE wird unter Dokumente zur Veranstaltung eine LaTeX-Vorlage zum Download angeboten).

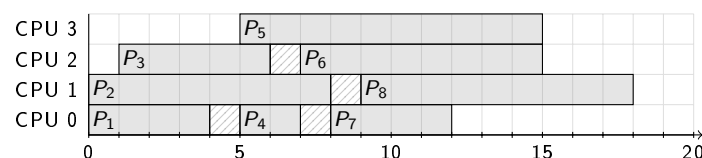
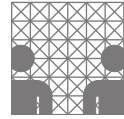


Abbildung 1: Beispiel für die Darstellung von Ausführungsreihenfolgen



## Aufgabe 2: Echtzeit & Mehrprozessor-Scheduling (20 Punkte)

- a) Gegeben seien die drei Aufträge  $A_1$  bis  $A_3$  (ohne I/O-Operationen) aus der folgenden Tabelle, die auf einem Einprozessorsystem periodisch auszuführen sind und dabei stets zum Ende jeder Periode vollständig abgearbeitet sein müssen. Leiten Sie mathematisch her, warum es selbst mit einem idealen Scheduler nicht möglich ist, die Deadlines aller Aufträge einzuhalten.

Auftrag	$A_1$	$A_2$	$A_3$
Periodendauer ( $p_i$ )	4	7	3
Bedienzeitforderung ( $b_i$ )	1	3	1

- b) Gegeben seien die vier Aufträge  $B_1$  bis  $B_4$  (ohne I/O-Operationen) aus der folgenden Tabelle, die auf einem Einprozessorsystem periodisch auszuführen sind und dabei stets zum Ende jeder Periode vollständig abgearbeitet sein müssen. Die Perioden der Aufträge beginnen gemeinsam zum Zeitpunkt  $t = 0$ .

Auftrag	$B_1$	$B_2$	$B_3$	$B_4$
Periodendauer ( $p_i$ )	7	11	9	4
Bedienzeitforderung ( $b_i$ )	3	1	2	1

Illustrieren Sie für das Zeitintervall  $t \in [0, 25]$  die Ausführungsreihenfolge der Aufträge (d. h. die CPU-Belegung über die Zeit durch die vier periodisch zu bearbeitenden Prozesse) für die Strategien

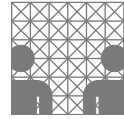
- (optional) Earliest Deadline First,
- Rate Monotonic Scheduling.

Der Aufwand für den Prozesswechsel soll in dieser Aufgabe vernachlässigt werden. Falls zu einem Zeitpunkt zwei oder mehr Aufträge dieselbe Priorität bzw. Deadline haben, soll der Auftrag mit der längsten Wartezeit ausgeführt werden. Falls Deadlines nicht eingehalten werden können, benennen Sie diese Fälle.

- c) Gegeben seien die sieben Prozesse  $P_1$  bis  $P_7$  (ohne I/O-Operationen) aus der folgenden Tabelle, die auf einem Multiprozessorsystem mit vier CPU-Kernen auszuführen sind. Illustrieren Sie die Ausführungsreihenfolge der sieben Prozesse auf den vier CPU-Kernen für die verdrängende Scheduling-Strategie *Timesharing* (auch *Load sharing*) nach Prioritäten. Der Aufwand für den Prozesswechsel ist dabei zu vernachlässigen. Sind mehrere CPU-Kerne frei, wird der mit der niedrigsten Nummer gewählt.

Prozess	$P_1$	$P_2$	$P_3$	$P_4$	$P_5$	$P_6$	$P_7$
Ankunftszeit ( $t_i$ )	0	2	2	3	3	5	9
Bedienzeitforderung ( $b_i$ )	4	6	5	8	8	5	4
Priorität	2	5	4	1	3	4	8

Hinweis: Bitte verwenden Sie zum Illustrieren der Ausführungsreihenfolgen in dieser Aufgabe eine Darstellung in Anlehnung an Abbildung 1.



## Aufgabe 3: Prioritätsinversion

(9 Punkte)

Erlaubt man Prozessen den exklusiven Zugriff auf Ressourcen unter Verwendung von Synchronisationsprimitiven wie zum Beispiel *Semaphoren*, kann es zu unerfreulichen Effekten kommen, sofern keine entsprechenden Maßnahmen getroffen werden. Zwei dieser Effekte sind *Prioritätsinversion* und *Deadlocks*.

Beinahe hätte die *Mars-Pathfinder*-Mission 1996/97 ein unrühmliches Ende gefunden. Informieren Sie sich z. B. im Internet<sup>1</sup> über die Gründe, warum sich der Computer der Mars-Sonde ständig selbst neu starten musste.

- a) Illustrieren Sie (unter Verwendung einer Darstellung in Anlehnung an Abbildung 1) mit Hilfe dieses Hintergrundwissens die Ausführungsreihenfolge der drei periodischen Aufträge  $B$  (Bus-Management),  $Z$  (z. B. Sensor auslesen) und  $M$  (Meteorologie-Auftrag) unter Verwendung einer Scheduling-Strategie nach festen Prioritäten für das Intervall  $[0-170]$ , wobei der Aufwand für den Prozesswechsel in dieser Aufgabe vernachlässigt werden soll. Die Prioritäten seien folgendermaßen definiert:  $Prio(B) > Prio(Z) > Prio(M)$ . Beachten Sie hierbei, dass der  $M$ -Auftrag und der  $B$ -Auftrag ihren Zugriff auf eine gemeinsame Ressource mittels einer Semaphore synchronisieren und die Verdrängung eines Auftrags nur geschieht, wenn der höher priorisierte Auftrag alle erforderlichen Ressourcen erhält.

Auftrag	$M$	$B$	$Z$
Periodendauer $p_i$	115	40	60
Bedienzeit $b_i$	30	10	20

- b) (optional) Erklären Sie anhand ihrer Illustration das Phänomen der *Prioritätsinversion*.
- c) (optional) Wie hat das Software-Team der *Mars-Pathfinder*-Mission das Problem schließlich gelöst? Illustrieren Sie die neue Ausführungsreihenfolge unter Beachtung ihrer Lösung.

<sup>1</sup>Glenn Reeves: What really happened on Mars?

[http://research.microsoft.com/en-us/um/people/mbj/Mars\\_Pathfinder/Authoritative\\_Account.html](http://research.microsoft.com/en-us/um/people/mbj/Mars_Pathfinder/Authoritative_Account.html)