Aufgabe 3.5: Scheduling-Theorie (1 Punkt) (Theorie[1])

Es werden in einem Ein-Prozessor-System Prozesse wie in Abbildung 2 beschrieben gestartet:

Prozess	A	B	C	D	E
Ankunftszeitpunkt	0	2	4	6	9
Dauer	5	3	6	2	7

Abbildung 2: Prozesse eines Systems mit einer CPU und einem Thread

1. a) Erklären Sie den Begriff *Verdrängung*. (0,1 Punkte)

   Verdrängung in der Scheduling Theorie, bezieht sich unabhängig von der Scheduling-strategie, auf das Konzept, dass falls ein ankommender Prozess eine höhere Priorität hat, als der momentan arbeitende Prozess, der momentan arbeitende Prozess unterbrochen, der Arbeitszustand gespeichert und der Prozess wieder zurück in die Warteschlange gesetzt wird. Dies erlaubt dem höher priorisierten ankommende Prozess die Prozessorkapazität zu nutzen und seine Tasks durchzuführen. Der in die Warteschlange platzierte Prozess gilt somit als „verdrängt“ und ist als der Entzug der Betriebsmittel von einem laufenden Prozess vor Ablauf seines Quantums zu verstehen.

   Werner (TU Chemnitz), Betriebssysteme – 4. Foliensatz, Folie 9

2. b) Zwischen welchen zwei Schedulingzielen bildet das HRRN-Verfahren einen Kompromiss? (0,2Punkte)

   Die Schedulingstrategie: Highest Response Rate Next (HRRN), berechnet dynamisch einen sogenannten response ration (rr), anhand dessen die Priorität verschiedener Prozesse gemessen werden. Je höher das rr eines Prozesses ist, desto höher priorisiert ist er.

   HRRN bildet einen Kompromiss zwischen den Schedulingstratgiem Shortest-Job_Next (SJN) und die länger andauernden Prozesse nicht verhungern zu lassen. Wie auch bei SJN, werden kürzere Prozesse bei HRRN bevorzugt, jedoch müssen längere Prozesse nicht längerfristig warten, da Sie durch ihre bereits gewartete Zeit an Priorität dazugewinnen.

   Werner (TU Chemnitz), Betriebssysteme – 4. Foliensatz, Folie 17

3. c) Erklären sie das Prinzip der *Prioritätsvererbung*. Was kann ohne sie passieren? (0,2 Punkte)

   Wenn sich ein niedrig priorisierter Prozess und ein hoch priorisierter Prozess dieselbe Ressource teilen (zb. Quellcode) und dem niedrig priorisierten Prozess wird die CPU,

zugewiesen, dann wird der hoch priorisierte Prozess geblockt, da er keinen Zugang auf die Ressource hat. Ein mittel priorisierter Prozess könnte nun den niedrig Priorisierten Prozess verdrängen, obwohl der hoch priorisierte Prozess auf die Freigabe seiner Ressource wartet. Das führt dazu, dass der hoch priorisierte Prozess „verhungert". Mithilfe der Prioritätsvererbung, würde der niedrig priorisierte Prozess die Priorität des hoch priorisierten Prozesses erben, sodass er den mittel priorisierten Prozess verdrängen kann und seine Ressource an den hoch priorisierten Prozess abgibt.

Kao, Dritter Foliensatz (Scheduling), Folie 21

4. d) Was sind die Unterschiede zwischen *Online-* und *Offline-Scheduling*? Gehen sie dabei auch auf die benötigten Voraussetzung für beide ein. (0,2 Punkte)

   Offline-Scheduling setzt voraus, dass alle Prozesse, auch zukünftig ankommende vor der Laufzeit bereits bekannt sind (vollständige Informationen liegen vor). Dies führt dazu, dass Offline Scheduling einen konkreten Endpunkt und auch immer das optimale Scheduling Ergebnis zurückliefert.

   Im Gegensatz dazu, steht das sogenannte: Online-Schduling, welches nur die aktuellen Prozesse kennt und eine Scheduling Entscheidung anhand von unvollständigen Informationen (keine Kenntnis von möglichen weiteren Prozessen) trifft, dies praktisch zur Laufzeit entscheidet.

   Im Allgemeinen, ist Offline-Scheduling wenn im gegebenen Kontext möglich, immer die beste Option für ein System, da es ein optimales Ergebnis und einen konkreten Endpunkt aller Prozesse erzeugt.

   Kao, Dritter Foliensatz (Scheduling), Folie 7

5. e) Was sind die Unterschiede und Gemeinsamkeiten von *Hard-* und *Soft-real-time-Systems*. Nennen sie jeweils ein Beispiel. (0,3 Punkte)

   Hard- und Soft-real-time-Systems, beziehen sich auf Scheduling mit Sollzeitpunkten, sprich Punkte an denen Prozesse spätestens fertig gelaufen sein sollten.

   Hard-real-time-Systems sind strikte Echtzeitsysteme, bei denen das Verpassen bzw. verletzen des Sollzeitpunktes nicht tolerierbar ist. Das bedeutet, dass ein Hard-real-time-System keinerlei Verspätung des Sollzeitpunktes akzeptiert, da dies möglicherweise katastrophale Folgen auf das Echtzeitsystem haben könnte.

   Im Gegensatz dazu, stehen die Soft-real-time-Systems, oder auch schwache Echtzeitsysteme genannt. Diese Systeme tolerieren eine Verspätung des eigentlichen Sollzeitpunktes, jedoch könnte diese Verspätung im späteren Verlauf des Systems zu Qualitätsverlusten führen.

Im Allgemeinen lässt sich somit sagen, dass bei beiden System (Hard und Soft) eine Verspätung des Sollzeitpunktes nicht wünschenswert ist, jedoch bei Soft-Systems toleriert wird, mit Abzug eines möglichen Qualitätsverlustes. Bei Hard-Systems jedoch nicht tolerierbar ist und mögliche katastrophale Auswirkungen daraus resultieren könnten.

Beispiel für ein Hard real-time System: Airbag => Verspätung hier lebensgefährlich
Beispiel für ein soft real-time System: Videoanruf => Verspätung hier lediglich Qualitätsverlust

Kao, Folien Scheduling Seite 32-34
Hermann Kopetz: Real Time Systems. Design Principles for Distributed Embedded Applications. Kluwer Academic Publishers, Boston MA u. a. 1997

Aufgabe 3.6: Scheduling-Handsimulation (1 Punkt) (Theorie^2)

Es werden in einem Ein-Prozessor-System Prozesse wie in Abbildung 2 beschrieben gestartet:

Es werden in einem Ein-Prozessor-System Prozesse wie in Abbildung 2 beschrieben gestartet:

Prozess	A	B	C	D	E
Ankunftszeitpunkt	0	2	4	6	9
Dauer	5	3	6	2	7

Abbildung 2: Prozesse eines Systems mit einer CPU und einem Thread

a) Simulieren Sie folgende Scheduling-Verfahren für die Prozesse aus Abbildung 2:

- LCFS-PR,
- HRRN,
- MLF mit $\tau_i = 2^i$ (i=0,1,...).

Geben Sie für *jeden* Zeitpunkt den Inhalt der Warteschlange und den Prozess auf der CPU an.
Die Lösung soll in Form der dargestellten Tabelle abgegeben werden, wobei anzumerken ist, dass für Multilevel-Feedback mehrere Warteschlangen benötigt werden:

Zeit	0	1	2	3	…	26	27
CPU	A	…	…	…	…	…	…
Warteschlange		…	…	…	…	…	…
		…	…	…	…	…	…

b) Berechnen Sie für *jedes* der in a) verwendete Verfahren

- die Warte- und Antwortzeit *jedes* Prozesses sowie
- die mittlere Warte- und Antwortzeit des gesamten Systems

a)

LCFS-PR	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
CPU	A	A	B	B	C	C	D	D	A	E	E	E	E	E	E	E	B	C	C	C	C	A	A
Queue(FIFO)			A	A	A B	A B	A B C	A B C	B C	B C A	B C A	B C A	B C A	B C A	B C A	B C A	C A	A	A	A	A		

HRRN	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
CPU	A	A	A	A	A	B	B	B	D	D	C	C	C	C	C	C	E	E	E	E	E	E	E
Queue(HRRN)			B	B	B C	**B (2)** C(1,17)	C D	C D	C(1,67) **D(2)**	C E	**C(2)** E(1,14)	E	E	E	E	E	**E(2)**						

Der RR steht jeweils in den Klammern. Fett gedruckte Prozesse sind nicht mehr Teil der Warteschlange.

MLF	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
CPU	A	A	A	B	C	B	B	D	C	C	E	D	E	E	A	A	C	C	C	E	E	E	E
t=1	**A**		B	**B**	**C**		D	**D**		E	**E**												
t=2		**A**	**A**		B	**B** C	**B** C	C	**C** D	**C** D	D	**D** E	**E**	**E**										
t=4				A	A	A	A	A	A	A	A C	A C	A C	A C	**A** C E	**A** C E	**C** E	**C** E	**C** E	**E**	**E**	**E**	**E**

b)

LCFS-PR
A:
Wartezeit=18
Antwortzeit=23
B:
Wartezeit=12
Antwortzeit=15

C:
Wartezeit=11
Antwortzeit=17
D:
Wartezeit=0
Antwortzeit=2
E:
Wartezeit=0
Antwortzeit=7

Durchschnittliche Wartezeit= 8,2
Durchschnittliche Antwortzeit= 12,8

HRRN:
A:
Wartezeit=0
Antwortzeit=5
B:
Wartezeit=3
Antwortzeit=6
C:
Wartezeit=6
Antwortzeit=12
D:
Wartezeit=2
Antwortzeit=4
E:
Wartezeit=7
Antwortzeit=14

Durchschnittliche Wartezeit= 3,6
Durchschnittliche Antwortzeit= 8,2

MLF:
A:
Wartezeit=1
Antwortzeit=16
B:
Wartezeit=2
Antwortzeit=5

C:
Wartezeit=9
Antwortzeit=16
D:
Wartezeit=4
Antwortzeit=6
E:

Wartezeit=7
Antwortzeit=7

Durchschnittliche Wartezeit=4,6
Durchschnittliche Antwortzeit=10