Fachgebiet Verteilte Systeme
Prof. Dr. Kurt Geihs
FB16 – Elektrotechnik/Informatik
Universität Kassel

VERTEILTE SYSTEME

Betriebssysteme WS 20/21
Übung: Stefan Jakob
jakob@vs.uni-kassel.de
http://www.vs.uni-kassel.de

Übungsblatt 3 – Scheduling-Verfahren

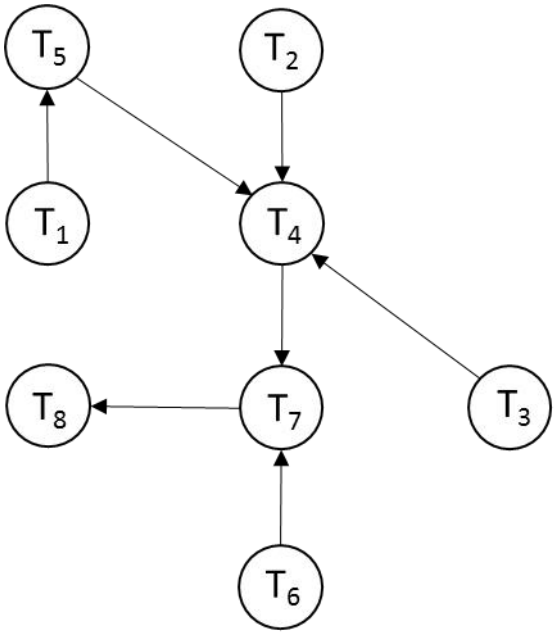
Punktzahl gesamt: 67 Punkte

Abgabe der Lösungen bis spätestens 04.12.20, 10:00 Uhr

Aufgabe 1 – B-Schedule

8 Punkte

a) Gegeben ist der folgende Taskgraph. Es wird angenommen, dass alle Tasks die gleiche Ausführungszeit haben und es keine Unterbrechungen geben kann. Wenden Sie den B-Schedule-Algorithmus hier an und stellen Sie eine mögliche Ausführungsreihenfolge als Gantt-Diagramm für M=3 dar! **8 Punkte**

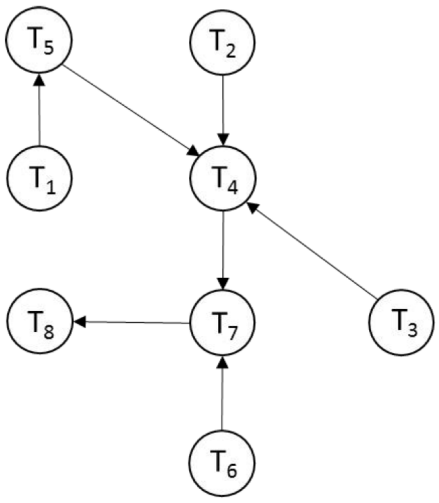


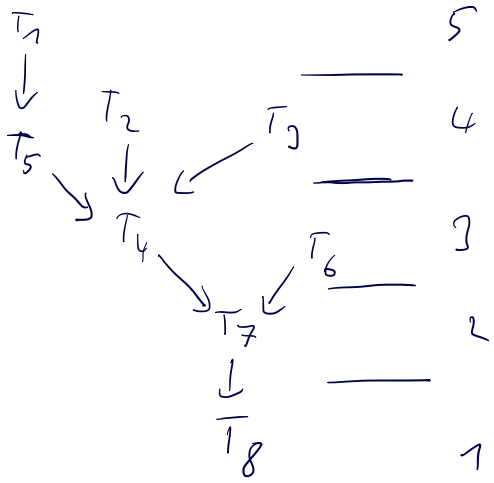
Aufgabe 1 – B-Schedule

8 Punkte

a) Gegeben ist der folgende Taskgraph. Es wird angenommen, dass alle Tasks die gleiche Ausführungszeit haben und es keine Unterbrechungen geben kann. Wenden Sie den B-Schedule-Algorithmus hier an und stellen Sie eine mögliche Ausführungsreihenfolge als Gantt-Diagramm für M=3 dar!

8 Punkte





P_1	T_1	T_5	T_4	T_7	T_8
P_2	T_2	T_6			
P_3	T_3				

a) LIFO – Last In First Out (ohne Unterbrechung)

8 Punkte

F	F	D	D	C	C	C	C	A	E	E	E	E	B	D	D
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

B,F E A,D C

b) SJN – Shortest Job Next (ohne Unterbrechung)

8 Punkte

F	F	A	D	D	B	B	B	C	C	C	C	E	E	E	E
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

c) HPF – Highest Priority First (mit Unterbrechung)

8 Punkte

B	B	A	B	C	E	E	E	F	F	D	D	C	C	C	C
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

d) SRTF – Shortest Remaining Time First (mit Unterbrechung)

8 Punkte

F	F	A	D	D	B	D	B	C	C	C	C	E	E	E	E
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

a) $r = \frac{(7+16+4+2+12+2)}{6} = \frac{43}{6}$

$w = \frac{43}{6} - \frac{16}{6} = \frac{27}{6} = 4$

b) $r = \frac{(1+8+8+3+15+2)}{6} = \frac{37}{6}$

$w = \frac{37}{6} - \frac{16}{6} = \frac{21}{6}$

c) $r = \frac{1+4+12+10+7+10}{6} = \frac{44}{6}$

$w = \frac{44}{6} - \frac{16}{6} = \frac{28}{6}$

d) $r = \frac{1+8+8+3+15+2}{6} = \frac{37}{6}$

$w = \frac{32}{6} - \frac{16}{6} = \frac{21}{6}$

Aufgabe 2 – Scheduling-Strategien

45 Punkte

Es steht ein Prozessor zur Verfügung. Sollte zu irgendeinem Zeitpunkt mehr als ein Prozess gemäß dem entsprechenden Scheduling-Algorithmus zur Ausführung in Frage kommen, dann gewinnt der Prozess mit dem nach alphabetischer Ordnung kleineren Namen (z.B. siegt A über B)(bei LIFO andersherum). Bitte beachten Sie, dass viele Scheduling-Algorithmen zunächst nach Eintrittszeitpunkt sortieren. Nur bei gleichem Eintritt würde somit eine alphabetische Sortierung in Frage kommen.

Prozessname T_i	Eintrittszeit $t_0(T_i)$	Rechenzeit $t(T_i)$	Priorität $P(T_i)$
A	2	1	1
B	0	3	1
C	4	4	3
D	2	2	2
E	1	4	1
F	0	2	1

Σ 16

Vervollständigen Sie die Gantt-Diagramme, wie sie durch die folgenden Scheduling-Strategien entstehen würden und berechnen Sie die fehlenden Zeiten der untenstehenden Tabelle:

a) LIFO – Last In First Out (ohne Unterbrechung) **8 Punkte**

F	F	D	D	C	C	C	C	A	E	E	E	E	B	D	D
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

B,F E 4,D C

b) SJN – Shortest Job Next (ohne Unterbrechung) **8 Punkte**

F	F	A	D	D	B	B	B	C	C	C	C	E	E	E	E
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

c) HPF – Highest Priority First (mit Unterbrechung) **8 Punkte**

B	B	A	B	C	C	C	C	F	F	D	D	C	C	C	C
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

d) SRTF – Shortest Remaining Time First (mit Unterbrechung) **8 Punkte**

F	F	A	D	D	B	D	B	C	C	C	C	E	E	E	E
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

e) RR 2 – Round Robin mit Zeitquantum 2 — 8 Punkte

B	B	F	F	E	E	B	A	D	D	C	C	E	E	C	C
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
B	B	F	F	E	E	B	A	D	D	C	C	E	E	C	C
F	F	E	E	B	B	A	D	C	C	E	E	C	C		
	E	B	B	A	A	D	C	E	E						
		A	A	D	D	C	E								
		D	D	C	C	E									

D, F E A, D C

$r = \frac{6+2+12+8+13+4}{6} = \frac{45}{6}$

	LIFO	SJN	RR 2	HPF	SRTF
unterbrechend / präemptiv	Nein	Nein	Ja	Ja	Ja
r	$\frac{43}{6}$	$\frac{27}{6}$	$\frac{45}{6}$	$\frac{44}{6}$	$\frac{22}{6}$
r_{max}	16	15	13	12	15
w	$\frac{27}{6}$	$\frac{21}{6}$	$\frac{29}{6}$	$\frac{28}{6}$	$\frac{21}{6}$
w_{max}	13	11	9	8	11
Fairness	nein	nein	ja	nein	nein
Priorisierung	nein*	implizit	nein*	explizit	implizit

* einfache Warteschlange

f) Welche Scheduling-Strategie aus Teilaufgabe a) bis e) hat die kürzeste mittlere Verweilzeit? Begründen Sie! — 5 Punkte

Aufgabe 3 – Scheduling in C++ implementieren

14 Punkte

Schreiben Sie in C++ einen Scheduler, der nach der FIFO-Strategie einkommende Tasks abarbeitet.

Implementieren Sie hierzu zunächst die **enqueueTasks**-Methode in der **main.cpp**, welche die **Tasks** gemäß der Tabelle aus Aufgabe 2 (Eintrittszeitpunkte und Rechenzeiten mittels `sleep` in Sekunden umsetzen) in die **taskQueue** einfügt.

Anschließend implementieren Sie die **schedule**-Methode der **Scheduler.cpp**. Normalerweise ist die Anzahl der Tasks in einem System nicht bekannt und kann zur Laufzeit nicht vorhergesagt werden. Damit das von Ihnen entwickelte Scheduling terminiert, wird anstatt der üblichen while(true) eine for-Schleife verwendet, die die maximale Anzahl der Tasks enthält. Innerhalb dieser Schleife soll zunächst in einer weiteren Schleife geprüft werden, ob Tasks vorhanden sind. Ist dies der Fall, soll der erste Task aus der **taskQueue** entnommen und abgearbeitet werden. **14 Punkte**