Betriebssysteme

Tutorium 5

Philipp Kirchhofer
philipp.kirchhofer@student.kit.edu
http://www.stud.uni-karlsruhe.de/~uxbtt/

Lehrstuhl Systemarchitektur
Universität Karlsruhe (TH)

25. November 2009

Was machen wir heute?

Tutorien Übungsblatt - Scheduling Basics

Frage 5.1.a

Erkläre die Begriffe „Verweilzeit“, „Wartezeit“ und „Antwortzeit“.

Antwort

Verweilzeit Zeit vom Starten eines Prozesses/Threads bis zu seiner Beendigung

Wartezeit Zeit, die ein Prozess/Thread in der Warteschlange verweilt

Antwortzeit Zeit vom Starten eines Prozesses/Threads bis zur ersten Antwort

Tutorien Übungsblatt - Scheduling Basics

Frage 5.1.b

Wie funktioniert SJF[a] scheduling?
Was ist der Unterschied zwischen präemptivem und nicht-präemptivem SJF?

[a] Shortest Job First

Antwort

SJF wählt immer den Job mit der geringsten verbleibenden Laufzeit als nächsten Job aus. Für die Auswahl kann auch die (geschätzte) Länge des nächsten CPU-Burst herangezogen werden.

Präemptiver SJF scheduler
Trifft bei jedem ankommenden neuen Prozess eine neue Entscheidung: Wenn der neue Prozess eine kürzere Laufzeit als der gegenwärtig laufende Prozess hat so wird der neue Prozess gestartet und der alte in die Warteschlange eingereiht.

Nicht-Präemptiver SJF scheduler
Schedulingentscheidungen werden nur nach Ende des laufenden Jobs durchgeführt.

Frage 5.1.c

Was ist die Idee eines prioritätsbasierten Schedulers?

Antwort

Jeder Prozess bekommt eine Priorität zugewiesen und der Prozess mit der höchsten Priorität wird als nächster Prozess ausgewählt. Bei mehreren Prozessen mit der gleichen Priorität wird ein weiterer Auswahl-Algorithmus ausgeführt (z.B. Round-Robin).

Frage 5.1.c

Welches große Problem entsteht beim Einsatz eines prioritätsbasierten Schedulers?

Antwort

Das Hauptproblem ist das **Verhungern von Prozessen (Starvation)**:
Ein Prozess mit einer bestimmten Priorität wird nie laufen, solange es Prozesse mit einer höheren Priorität gibt.

Frage 5.1.d

Erkläre den Multilevel Feedback Queue Algorithmus.
Welche Arten von Prozessen kommen häufig in den oberen Ebenen, welche Arten in den unteren Ebenen vor?

Antwort

Multilevel Feedback Queue Scheduling verwendet mehrere Warteschlangen mit unterschiedlichen Charakteristiken.

- Prozesse werden anhand diesen Charakteristiken zwischen den Warteschlangen verschoben.
- Prozesse werden immer von der höchsten, belegten Warteschlange genommen
- Prozesse in der höchsten Warteschlange erhalten kleine Zeitscheiben. Sobald sie eine Zeitscheibe komplett aufgebraucht haben werden sie in die nächste tiefere Ebene verschoben

Daraus ergibt sich, dass nur E/A-intensive Prozesse in den höheren Ebenen verbleiben.

Frage 5.1.e

Wie kann das Verhungern von Prozessen mit dem Multilevel Feedback Queue Scheduling verhindert werden?

Antwort

Ähnlich wie bei prioritätsbasierten Schedulern können Prozesse bei Multilevel Feedback Queue Scheduling verhungern.

Eine Lösung dieses Problems ist das Berechnen der Wartezeit von jedem Prozess. Wenn ein Prozess zu lange wartet wird er in die nächsthöhere Warteschlange eingereiht.

Dieses Schema kann auch bei prioritätsbasierten Schedulern eingesetzt werden.

Frage 5.1.f

Beschreibe „Lottery Scheduling“.

Antwort

Bei Lottery Scheduling bekommt jeder Prozess eine bestimmte Anzahl an Tickets. Der Scheduler wählt zufällig eines der vergebenen Tickets aus und lässt den entsprechenden Prozess laufen.

Tutorien Übungsblatt - Scheduling Basics

Frage 5.1.g

Welche Vorteile hat Lottery Scheduling im Vergleich zu prioritätsbasierten Scheduling Verfahren?

Antwort

- Kein Verhungern von Prozessen
- Einfache Verteilung der Rechenzeit
- Hierarchische Verteilung von Rechenzeit möglich
- Korrekte Abrechnung von Client-Server Anfragen

Tutorien Übungsblatt - Scheduling Policies

Frage 5.2

Die folgenden fünf Prozesse werden in der genannten Reihenfolge direkt nacheinander gestartet. Priorität 10 ist die höchste Priorität.
Berechne die Verweilzeit für alle Prozesse und die durchschnittliche Verweilzeit für jedes der vier folgenden Scheduling Verfahren:

- Round-Robin (Zeitscheibenlänge 1, alphabetische Reihenfolge)
- Prioritätsbasiert
- First Come First Served
- Shortest Job First

Prozess	Rechenzeit	Priorität
A	15	4
B	9	7
C	3	3
D	6	1
E	12	6

Tutorien Übungsblatt - Scheduling Policies

Round-Robin

#	A	B	C	D	E
1	1	2	3	4	5
2	6	7	8	9	10
3	11	12	**13**	14	15
4	16	17		18	19
5	20	21		22	23
6	24	25		**26**	27
7	28	29			30
8	31	32			33
9	34	**35**			36
10	37				38
11	39				40
12	41				**42**
13	43				
14	44				
15	**45**				

$$VZ(A) = 45$$
$$VZ(B) = 35$$
$$VZ(C) = 13$$
$$VZ(D) = 26$$
$$VZ(E) = 42$$

$$T^t_{avg}(RR) = \frac{45 + 35 + 13 + 26 + 42}{5} = \frac{161}{5} = 32 \text{ Minuten und } 12 \text{ Sekunden}$$

Tutorien Übungsblatt - Scheduling Policies

Prioritätsbasiert

Priorität	Prozess	VZ
7	B	**9**
6	E	$9 + 12 = \mathbf{21}$
4	A	$21 + 15 = \mathbf{36}$
3	C	$36 + 3 = \mathbf{39}$
1	D	$39 + 6 = \mathbf{45}$

$$T^t_{avg}(Prio) = \frac{9 + 21 + 36 + 39 + 45}{5} = \frac{150}{5} = 30 \text{ Minuten}$$

First Come First Served

Prozess	VZ
A	**15**
B	15 + 9 = **24**
C	24 + 3 = **27**
D	27 + 6 = **33**
E	33 + 12 = **45**

$$T^t_{avg}(FCFS) = \frac{15 + 24 + 27 + 33 + 45}{5}$$
$$= \frac{144}{5}$$
$$= 28 \text{ Minuten und } 48 \text{ Sekunden}$$

Shortest Job First

Rechenzeit	Prozess	VZ
3	C	**3**
6	D	3 + 6 = **9**
9	B	9 + 9 = **18**
12	E	18 + 12 = **30**
15	A	30 + 15 = **45**

$$T^t_{avg}(FCFS) = \frac{3 + 9 + 18 + 30 + 45}{5}$$
$$= \frac{105}{5}$$
$$= 21 \text{ Minuten}$$

Frage 5.3.a

Welche Zeitscheibenlängen werden üblicherweise verwendet?

Antwort

10 bis 100 Millisekunden

Frage 5.3.b

Was sind die Vor- und Nachteile von kurzen Zeitscheiben im Vergleich zu langen Zeitscheiben?

Antwort

Bei langen Zeitscheiben müssen weniger Kontextwechsel durchgeführt werden. Da ein Kontextwechsel immer einen zusätzlichen Rechenaufwand erfordert kann durch die Verminderung der Anzahl an Kontextwechseln eine Erhöhung des Durchsatzes erzielt werden.

Lange Zeitscheiben verringern allerdings auch die Reaktionsfähigkeit eines Systems: Jeder Prozess läuft länger, damit müssen aber auch Prozesse in der Warteschlange länger warten.

Tutorien Übungsblatt - Scheduling Parameters

Frage 5.3.c

Für welche Arten von Systemen eignen sich kurze/lange Zeitscheiben?

Systeme

- Mischung aus interaktiven und rechenintensiven Prozessen (Desktopsystem)
- Nur rechenintensive Prozesse (Server)

Antwort

Mit kurzen Zeitscheiben kann die Illusion einer parallelen Ausführung von mehreren Programmen erzielt werden. Gleichzeitig wird die Antwortzeit von Anwendungen reduziert.

⇒ Kurze Zeitscheiben eignen sich gut für Systeme mit einer Mischung von interaktiven und rechenintensiven Prozessen

---

Mit langen Zeitscheiben kann der Systemoverhead durch Vermeidung von Kontextwechseln verkleinert werden. Dies ist besonders wichtig bei Servern.

⇒ Lange Zeitscheiben eignen sich gut für Systeme mit rechenintensiven Prozessen

Tutorien Übungsblatt - Scheduling Parameters

Frage 5.3.d

Was ist der „tickless“ Modus in Linux?

Antwort

Im normalen Modus werden automatisch regelmäßig Interrupts (z.B. Timer-Interrupt) ausgelöst.

Im „tickless“ Modus werden die sonst regelmäßigen Interrupts nur bei Bedarf ausgelöst. Im Extremfall, bei leerlaufendem Prozessor, werden überhaupt keine Interrupts mehr ausgelöst. Durch den Wegfall von zu bearbeitenden Interrupts kann der Prozessor in tiefere Schlafzustände gehen und dadurch Energie sparen.

Tutorien Übungsblatt - Scheduling in SMP Systems

Frage 5.4.a

Welche besonderen Aspekte treten nur bei Scheduling mit mehreren Prozessoren auf?

Antwort

- Verschiedene Threads eines Prozesses auf einem Prozessor
  ⇒ Optimale Cache und TLB Ausnutzung
  ⇒ Verringerung der Parallelisierung
- Kommunizierende Prozesse parallel auf mehreren Prozessoren
  ⇒ Verringerung der Kommunikationslatenz
  ⇒ Cache Ping-Pong erhöht Busauslastung

Adaptive Verfahren wechseln dynamisch zwischen den Vorgehensweisen.

Tutorien Übungsblatt - Scheduling in SMP Systems

Frage 5.4.b

Beim Entwurf eines Multi-Prozessor Schedulers kann zwischen einer zentralen Warteschlange und mehreren Warteschlagen (für jeden Prozessor eine) gewählt werden. Welche Vor- und Nachteile entstehen durch eine zentrale Warteschlange?

Antwort

- ⊖ Bei einer zentralen Warteschlangen müssen Zugriffe auf die Warteschlange zwischen den verschiedenen Prozessoren synchronisiert werden. Dadurch wird wird die Skalierbarkeit eingeschränkt.
- ⊖ Weiterhin führen Veränderungen der Warteschlange zu einer Invalidierung von Cache Zeilen, die Cache Miss Rate wird erheblich erhöht.

⇒ Dies führt insgesamt gesehen zu einer schlechten Geschwindigkeit bei einer großen Anzahl an Prozessoren.

Tutorien Übungsblatt - Scheduling in SMP Systems

Antwort

⊕ Durch eine zentrale Warteschlange kann die Verteilung der Last zwischen den Prozessoren leicht durchgeführt werden. Jeder Prozessor nimmt einfach den ersten Prozess vom Anfang der Warteschlange.

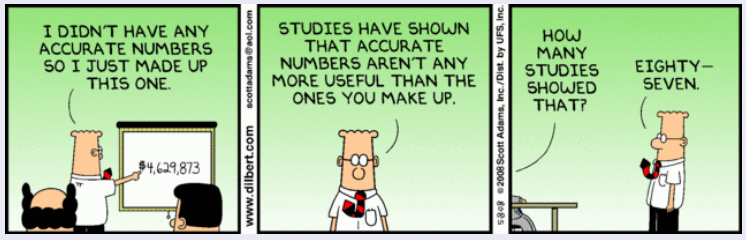
⊖ Durch dieses Verfahren wandern allerdings die Prozesse oft von einem Prozessor zu einem anderen, dadurch wird die Effizienz von Caches weiter herabgesetzt.

Um dieses Problem zu vermeiden bilden die meisten Scheduler eine Prozessoraffinität für Prozesse.

Fragen & Kommentare

Fragen & Kommentare?

Ende



Dilbert