Freie Universität Berlin

Übung 5

Betriebs- und Kommunikationssysteme
Sommersemester 2016

Ausgabe 20.05.2016
Abgabe 03.06.2016

Aufgabe 1

Begriffe

Erklären Sie jeden der folgenden Begriffe mit einem Satz: Scheduling Strategie, Round Robin, HHRN, SPN, SRT und FCFS. Was ist eine Feedback Queue?

Textaufgabe

Sie haben in der Vorlesung nicht-preemptive Scheduling Strategien kennengelernt. In der Vorlesung wurden die Strategien jeweils für eine CPU vorgestellt. Sie sind aber leicht so erweiterbar, dass sie ankommende Prozesse auf mehreren CPUs verteilen können.

Betrachten wir als Beispiel die Kassen eines Supermarktes. Jede Kasse entspricht einer CPU und jeder Kunde, der an die Kasse kommt, stellt einen Prozess dar. Die Zahl der Artikel, die ein Kunde eingekauft hat, entspricht der Zeit, die ein Prozess zur Ausführung benötigt. Wir nehmen dabei an, dass die Zeit zum Registrieren eines Artikels an der Kasse konstant ist und setzen diese mit einer Zeiteinheit an, den eigentlichen Bezahlvorgang vernachlässigen wir.

Da es Kunden im Supermarkt nicht zugemutet werden kann, andere Kunden vorzulassen, kommt üblicherweise die FIFO-Strategie zum Einsatz, d.h. wer zuerst die Kasse erreicht, wird zuerst abkassiert. Da die Kunden eines Supermarktes auf ihren Vorteil bedacht sind, stellen sie sich immer an der Kasse an, an der die Wartezeit am geringsten ist, d.h. an der vor ihnen die wenigsten Artikel registriert werden müssen.

a) Ein Supermarkt habe drei Kassen (K1, K2 und K3). An keiner der Kassen wartet ein Kunde. Nun treffen (fast gleichzeitig) Kunden mit folgenden Artikelanzahlen (in dieser Reihenfolge) ein:

   6, 15, 23, 10, 3, 13, 15, 7, 20, 40, 19, 4, 6, 21 (14 Kunden)

   O.B.d.A. stellt sich der erste Kunde an K1, der zweite an K2, der dritte an K3 an.

   - Dokumentieren Sie die sich ergebenden Warteschlangen an den Kassen K1, K2 und K3.
   - Wie viele Zeiteinheiten muss ein Kunde im Durchschnitt warten, bis er an der Kasse ankommt?

b) Der Leiter des Supermarktes ist mit dieser mittleren Wartezeit für seine Kunden nicht zufrieden und führt eine Kasse ein, an der nur Kunden bedient werden, welche weniger oder gleich 11 Artikel eingekauft haben (K1 wird in diesem Sinn ausgezeichnet). Kunden mit wenigen Artikeln dürfen sich natürlich nach wie vor auch an den anderen Kassen anstellen, sofern sie eine geringere Wartezeit erwarten.

   - Dokumentieren Sie die sich ergebenden Warteschlangen (K1 ist die Kasse für Kunden mit weniger als 12 Artikeln; die ersten Kunden stellen sich an die Kassen K1, K2, K3 - in dieser Reihenfolge).
   - Berechnen Sie die mittlere Wartezeit der Kunden.
   - Was für eine Strategie versucht der Supermarktleiter näherungsweise zu erreichen?
   - Was für einen Nachteil hat die neue Regelung?

Aufgabe 2

In dieser Aufgabe soll eine Prozessverwaltung mit Hilfe einer doppelt verketteten Liste simuliert werden. Führen Sie Fehlerbehandlung durch und erläutern Sie im Quellcode (als Kommentar) warum Sie sich für genau diese Fehlerbehandlung entschieden haben. Die Ausgaben sollen auf `stdout` ausgegeben werden, die Fehler auf `stderr`. Ihr Programm muss mit folgenden Compilerflags ohne Warnungen und / oder Fehler compilieren:

```
$ gcc -std=c11 -o <program.out> -Wall -Wextra -pedantic <program.c>
```

Implementieren

Implementieren Sie die Scheduling Algorithmen Round Robin, First Come First Serve und Shortest Process Next auf einer verketteten Liste von Prozessen.

- `elem roundRobin(elem head,elem current,int tStep);`
- `elem fcfs(elem head,elem current,int tStep);`
- `elem spn(elem head,elem current,int tStep);`

Die einzelnen Elemente `elem` der Prozessliste sind wie folgt definiert:

```
typedef struct ___elem        // Elemente der verketteten Liste
{
    uint64_t cycles_done;    // Anzahl bereits durchlaufener Zyklen
    uint64_t cycles_waited; // Anzahl bereits gewarteter Zyklen
    uint64_t cycles_todo;    // Anzahl der zu verarbeitenden Zyklen
    uint64_t pId;            // Prozess ID

    struct ___elem  *next;   // Doppelt verkettete Liste
    struct ___elem  *prev;
} _elem;

typedef _elem* elem;
```

Wenn nötig können sie weitere Variablen zu der Struktur hinzufügen.

Testen

Testen Sie ihre Implementierung mit folgendem Beispiel.

```
uint64_t processes[4][2]={{0,3},{2,7},{4,1},{6,5}};
```

Die erste Komponente des Arrays repräsentiert den Prozess. Die Einträge der zweiten Komponente repräsentieren Ankunftszeit und Ausführungsdauer. So gibt `processes[1][0]` die Ankunftszeit des zweiten Prozesses aus (im Beispiel 2). Mit `processes[1][1]` erhalten wir seine Ausführungsdauer (im Beispiel 7).