Semestrální práce z předmětu KIV/ZOS

Tomáš Maršálek, A10B0632P
marsalet@students.zcu.cz

25. prosince 2012

1 Problém spícího holiče

Ve městě sídlí jeden holič, který má pouze jedno křeslo, ve kterém holí zákazníky. Když do jeho holičství přijde zákazník, usadí se v čekárně a bude čekat, dokud ho holič nepřijme do křesla. Čekárna má ovšem omezený počet míst, tedy v případě plné čekárny každý další zákazník musí odejít.

2 Řešení

Program je v jazyce C, pro paralelizaci jsou použita vlákna knihovny **pthread**.

2.1 Správa vláken

Pro vytvoření vlákna je použita funkce **pthread_create**, u které je třeba kontrolovat vrácenou hodnotu. V případě hodnoty různé od nuly se nepodařilo vytvořit vlákno a program je raději ukončen.

Na konci programu se čeká na všechna vytvořená vlákna pomocí **pthread_join**. Bez použití tohoto čekání by program skončil dříve, než by všechna vlákna dokončila svou práci.

2.2 Synchronizace

Celkem je použito sedm semaforů. Čtyři pro samotný korektní běh problému spícího holiče a dva semafory čistě kvůli synchronizaci výpisů holiče a zákazníka.

1. *customer_ready* - zákazníci dávají vědět holiči, že jsou připraveni k ostříhání.
2. *barber_ready* - holič dává vědět zákazníkům, že je volné křeslo.
3. *queue_empty* - celočíselná hodnota, udává počet volných míst v čekárně. Je chráněna binárním semaforem *queue_lock*.
4. *n_lock* - chrání proměnnou n, aby nedošlo k dvojímu zápisu. Proměnná n udává počet zákazníků, kteří buď skončili v čekárně nebo odešli kvůli plné čekárně.
5. *print_lock* - zámek kolem každého výpisu, aby náhodou nedošlo k vypisování ze dvou vláken najednou.
6. *becut_sync* - čistě synchronizační semafor pro vlákna zákazníků, aby holičův výpis proběhl před zákazníkovým.
7. *cut_sync* - stejná funkce, jen pro vlákno holiče.

3 Použití programu

Program spustíme z konzole buďto bez parametrů, kde počet zákazníků bude na výchozí hodnotě 17 a velikost čekárny 4.

```
$ ./sleepingbarber
```

Chceme-li parametry modifikovat, zadáme nejprve počet zákazníků, poté velikost čekárny.

```
$ ./sleepingbarber 8 4
```

4 Ukázka běhu programu

```
Number of customers: 8
Queue size         : 4

START
Customer 0 waits.
Barber cuts hair.
Customer 0 is cut.
Customer 1 waits.
Barber cuts hair.
Customer 1 is cut.
Customer 3 waits.
Barber cuts hair.
Customer 3 is cut.
Customer 2 waits.
Barber cuts hair.
Customer 2 is cut.
Customer 4 waits.
Barber cuts hair.
Customer 4 is cut.
Customer 5 waits.
Barber cuts hair.
Customer 5 is cut.
Customer 6 waits.
Customer 7 waits.
Barber cuts hair.
Customer 6 is cut.
Barber cuts hair.
Customer 7 is cut.
END
```