PPDS - 5. zadanie

Poznámka

Dokumentácia môže riešenie často porovnávať s riešením modifikácie problému divochov #1 (ďalej iba MPD1), ktoré bolo ukázané na seminári a ktoré bolo použité ako základ pri riešení modifikácie problému divochov #2.

Analýza

O aké synchronizačné problémy sa bude jednať?

- ako zabezpečíme že divochom oznámi práve jeden kuchár že je hotové jedlo?
- ako zabezpečíme že divosi nebudú jesť kým sa bude variť?
- ostatné problémy sú rovnaké ako u MPD1

V zadaní sme použili tieto synchronizačné nástroje

- SimpleBarrier
- Mutex
- Semaphore
- Event

Postup

Oproti MPD1 sa musíme vysporiadať s *n* kuchármi. Semafory inicializované na nulu, ktoré zastavovali/spúštali divochov a kuchára môžu byť ponechané, ale my sme na to išli aj cez Eventy.

V našom riešení sme použili Event. Ten, keď nastane, spustí všetkých *n* kuchárov. To, či môžu divosi jesť môžeme nechať implementované ako semafor, ako pri MPD1.

Je potrebné riešiť, aby iba jeden z kuchárov spustil divochov. Po dovarení všetkých kuchárov nasleduje KO riadená mutexom. Kuchári vstupujú po jednom a pozrú sa, či už boli divosi upozornení. Ak nie, daný kuchár naplní hrniec, oznámi divochom, že hrniec je plný, reštartuje udalosť a spolu s ostatnými kuchármi sa dostáva do - opäť - čakajúceho stavu.

Na to, aby divosi nejedli, kým sa varí, stačí taktiež vytvoriť KO. V nej si budú divosi naberať z hrnca a kontrolovať, či je prázdny. Ten divoch ktorý zistí, že je hrniec prázdny, uvedomí kuchárov a pozastaví samého seba (a tým pádom aj ostatných divochov - daný divoch je v KO, takže ostatní čakajú pred KO, takže čakajú spolu s ním, takže čakajú všetci).

Pseudokód

```
barrier := new Barrier
KO := new Mutex
cook := new Event
eat := new Semaphore
pot_full := false
pot_capacity := 5
portions = 0
savages := 4
cooks := 3

for i := 0 to savages do
     t := new Thread
     t.launch(savaging)
endfor

for i := 0 to cooks do
     t := new Thread
     t.launch(cooking)
endfor

func cooking()
     while 1 equal to 1 do
          cook.wait()
          pot_full = false
          barrier.wait(cooks)
          KO.lock()
          if not pot_full
               portions := pot_capacity
               eat.signal()
               pot_full = true
               cook.restart()
          endif
          KO.unlock()
     endwhile
endfunc

func savaging()
     barrier.wait(savages)
     while 1 equal to 1 do
          KO.lock()
          if portions is 0
               cook.signal()
               eat.wait()
          endif
          KO.unlock()
          portions := portions - 1
```

```
    endwhile
endfunc
```

Beh programu

Program spustíme a sledujeme výstup:

```
divoch  0: prisiel som na veceru, uz nas je  1
divoch  1: prisiel som na veceru, uz nas je  2
divoch  2: prisiel som na veceru, uz nas je  3
divoch  3: prisiel som na veceru, uz nas je  4
divoch  0: uz sme vsetci, zaciname vecerat
divoch  0: pocet zostavajucich porcii v hrnci je  0
divoch  0: budim kucharov
kuchar  0: varim
kuchar  1: varim
kuchar  2: varim
kuchar  0: dovarili vsetci kuchari
divoch  0: beriem si porciu
divoch  0: hodujem
divoch  3: pocet zostavajucich porcii v hrnci je  4
divoch  3: beriem si porciu
divoch  3: hodujem
divoch  2: pocet zostavajucich porcii v hrnci je  3
divoch  2: beriem si porciu
divoch  2: hodujem
divoch  1: pocet zostavajucich porcii v hrnci je  2
divoch  1: beriem si porciu
divoch  1: hodujem
divoch  2: pocet zostavajucich porcii v hrnci je  1
divoch  2: beriem si porciu
divoch  2: hodujem
divoch  1: pocet zostavajucich porcii v hrnci je  0
divoch  1: budim kucharov
kuchar  0: varim
kuchar  1: varim
kuchar  2: varim
kuchar  0: dovarili vsetci kuchari
divoch  1: beriem si porciu
divoch  1: hodujem
divoch  0: pocet zostavajucich porcii v hrnci je  4
divoch  0: beriem si porciu
divoch  0: hodujem
divoch  3: pocet zostavajucich porcii v hrnci je  3
```

Ukázaný výstup vznikol po dodaní vstupov:

- hrniec môže mať v sebe maximálne 5 porcií misionára
- kuchári sú traja
- divosi sú štyria

Hneď po spustení programu sa divosi najprv počkajú.
Potom chcú jesť. Nemajú čo. Okamžite budia kuchárov.

Každý kuchár varí a keď dovarí, čaká na bariére na ostatných.
Následne sú porcie doplnené a divosi môžu začať hodovať.

Zakaždým, keď si divoch zoberie porciu, ubudne z ostávajúcich porcií.
Po vyprázdnení hrnca sa proces opakuje ako na začiatku.

Poznámka:
Na fotke to nemôžeme vidieť, ale čím menej kuchárov vložíme do programu, tým dlhšie trvá fáza varenia. Lineárny rast kuchárov spôsobuje logaritmický rast časovej efektivity.