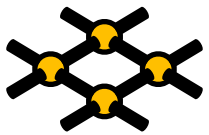
Vláknové programování
část XI

Lukáš Hejmánek, Petr Holub

`{xhejtman,hopet}@ics.muni.cz`



Laboratoř pokročilých síťových technologií

PV192

2012–05–15

Přehled přednášky

Ada

Závěrečný projekt

Atomické a volatilní proměnné

- `pragma Atomic ();`
  - zajišťuje atomické aktualizace proměnných

```
1 Prirozene_Cislo : Natural;
  pragma Atomic (Prirozene_Cislo);
```

- `pragma Atomic_Components ();`
  - zajišťuje atomické aktualizace součástí složeného typu `record`

```
  type Byte is range 0 .. 255;
2 for Byte'Size use 8;
  type Moje_Struktura is
4 record
          B1 : Byte;
6         B2 : Byte;
          B3 : Byte;
8         B4 : Byte;
  end record;
10 pragma Atomic_Components (Moje_Struktura);
```

Atomické a volatilní proměnné

- **`pragma Volatile ();`**
  - upozornění pro kompilátor, že se hodnoty proměnných mohou neočekávaně měnit
  - zejména kompilátor musí zamezit optimalizacím, které by mohly interferovat (zakazuje cachování na čtení i zápis)

```
Buffer_Zarizeni : Integer;
pragma Volatile (Buffer_Zarizeni);
```

- **`pragma Volatile_Components ();`**
  - totéž pro komponenty složeného typu **`record`**

Simpsonův algoritmus

- Kompilátor může mít omezení na maximální délku atomické proměnné
  - pokud je (nesplnitelný) požadavek na větší atomickou proměnnou, musí ho kompilátor odmítnout
- Algoritmus pro větší proměnné: Simpson'90[1]
  - jeden čtenář, jeden zapisovatel
  - dva sloty, každý o dvou bankách
  - do jednoho slotu se zapisuje (round-robin do bank)
  - ze druhého slotu se čte (poslední zapsaná hodnota)
  - atomické nastavování indexů slotů a bank
  - volatilní zápisy do bank/slotů
  - **`http://www.cs.ox.ac.uk/ucs/rushbysimpson.pdf`**

```
generic
   type Data is private;
   Initial_Value : Data;
package Simpsons_Algorithm is
   procedure Write(Item : Data); -- non-blocking
   procedure Read (Item : out Data); -- non-blocking
end Simpsons_Algorithm;
```

[1] H. Simpson, 'Four-Slot Fully Asynchronous Communication Mechanism', IEE Proceedings, 137 (Pt.E.1), 17–30 (January 1990). Implementace z CRTPA.

Simpsonův algoritmus

```
1  package body Simpsons_Algorithm is
      type Slot is (First, Second);
3     Four_Slot : array (Slot, Slot) of Data :=
         (First => (Initial_Value,Initial_Value),
5        Second => (Initial_Value,Initial_Value));
      pragma Volatile(Four_Slot);
7
      Next_Slot : array(Slot) of Slot := (First, First);
9     pragma Volatile(Next_Slot);

11    Latest : Slot := First;
      pragma Atomic(Latest);
13
      Reading : Slot := First;
15    pragma Atomic(Reading);
```

Simpsonův algoritmus

```
   procedure Write(Item : Data) is
2     Pair, Index : Slot;
   begin
4     if Reading = First then
      Pair := Second;
6     else
      Pair := First;
8     end if;
      if Latest = First then
10    Index := Second;
      else
12    Index := First;
      end if;
14    Four_Slot(Pair, Index) := Item;
      Next_Slot(Pair) := Index;
16    Latest := Pair;
   end Write;
```

Simpsonův algoritmus

```
   procedure Read(Item : out Data) is
2     Pair, Index : Slot;
   begin
4     Pair := Latest;
      Reading := Pair;
6     Index := Next_Slot(Pair);
      Item := Four_Slot(Pair, Index);
8  end Read;

10 end Simpsons_Algorithm;
```

Protected Types – Monitory

- Implementace monitorů
  - funkce – nemohou měnit data
    procedury – mohou měnit data
    entry – strážený vstup, mohou měnit data
  - efektivní paralelizace: podobne `ReadWriteLock`u v Javě
    - funkce mohou přistupovat paralelně
    - procedury a entries musí pracovat exkluzivně

```
1 protected type Muj_Typ is
     procedure Nastav_hodnotu (n : Integer);
3    procedure Odnastav_hodnotu;
     function Zjisti_hodnotu return Integer;
5    entry Pockej_na_nastaveni (n : Integer);
  private
7    Hodnota : Integer;
     Nastaveno : Boolean := False;
9 end Muj_Typ;
```

Protected Types – monitory

```
10 protected body Muj_Typ is
      procedure Nastav_hodnotu (n : Integer) is
12    begin
         Hodnota := n;
14       Nastaveno := True;
      end Nastav_hodnotu;
16
      procedure Odnastav_hodnotu is
18    begin
         Nastaveno := False;
20    end Odnastav_hodnotu;

22    function Zjisti_hodnotu return Integer is
      begin
24       return Hodnota;
      end Zjisti_hodnotu;
26
      entry Pockej_na_nastaveni
28    when Nastaveno is
      begin
30       null;
      end Pockej_na_nastaveni;
32 end Muj_Typ;
```

Guarded Entries

- chránění dle privátního stavu

```
protected type Chraneno_Stavem is
   entry Vstup;
private
   I : Integer;
end Chraneno_Stavem;

protected body Chraneno_Stavem is
   entry Vstup when I > 0 is
   begin
      null;
   end Vstup;
end Chraneno_Stavem;
```

  - používat pouze privátní proměnné
  - např. implementace mutexů a semaforů

Guarded Entries

- Chránění dle atributů

```
1  protected type Chraneno_Stavem is
       entry Vstup;
3  private
       I : Integer;
5  end Chraneno_Stavem;

7  protected body Chraneno_Stavem is
       entry Vstup when Vstup'Count > 4 is
9      begin
           null;
11     end Vstup;
   end Chraneno_Stavem;
```

  - možnost použití atributů
  - atribut `E'Count` vrátí počet zablokovaných vláken na vstupu do `entry E`
  - např. implementace bariér
  - funguje bezpečně pouze u chráněných objektů, nikoli tasků

Závěrečný projekt

- Smyslem je navrhnout a implementovat paralelismus do existujícího rozsáhlejšího technického kódu.
- Odevzdávání:
  - minimálně 3 dny před zkouškou
  - odevzdat implementaci
  - odevzdat 1-2 strany dlouhou zprávu o řešení projektu

Závěrečný projekt: Java – lehčí varianta

Vláknová paralelizace raytraceru pro Longeron Challenge

- **`http://www.topcoder.com/contest/problem/ISS/manual.html`**
- **`http://www.topcoder.com/contest/problem/ISS/v9/ISSVis.java`**
- pro testování možno použít model Tomáše Brüknera **`iss_sol.zip`**, který je v učebních materiálech v ISu

Závěrečný projekt: Java – těžká varianta

Vláknová paralelizace řešiče systémů s omezujícími podmínkami: Choco

- **`http://www.emn.fr/z-info/choco-solver/`**
- existuje distribuovaná verze DisCHOCO
  `http://dischoco.sourceforge.net/`
- **`http://www.4c.ucc.ie/~mwahbi/files/dcr11-dischoco2.pdf`**
- **`http://liawww.epfl.ch/Publications/Archive/DCR07Proceedings.pdf`**

Závěrečný projekt: C/C++

Vláknová paralelizace nástroje pro interpolaci snímků pomocí detekce pohybu: yuvmotionfps

- **`http://jcornet.free.fr/linux/yuvmotionfps.html`**