Dokumentace k semestrální práci

Paralelní programování

2018 Bc. Miloslav Moravec

Obsah

[1 Zadání 3](#_Toc529366832)

[2 Popis realizace 4](#_Toc529366833)

[2.1 Implementace 4](#_Toc529366834)

# Zadání

Cílem práce bylo vytvořit simulaci rozjezdu kolony vozidel. Každé vozidlo je reprezentováno vláknem, jehož číslo představuje označení vozidla.

Simulace má za úkol sledovat rozdíly v rychlostech vozidel v daný časový okamžik, kdy vozidla stojící v předu kolony mají zákonitě vyšší rychlost než vozidla na konci kolony.

Aplikace po ukončení simulace zobrazuje tabulku pro zadaný počet vozidel a jejich jednotlivé rychlosti při každé iteraci až do chvíle, kdy nedosáhne maximální rychlosti i poslední vozidlo.

# Popis realizace

K realizaci simulace byla použita soustava direktiv Open MP, která byla využita pro práci s paralelním programováním.

## Implementace

Základním prvkem je kolekce *vector,* která v sobě udržuje kolekci *array*. *Array* je tak velká, jako počet aut, který je nadefinovaný v konstantě *NUMBER\_CARS.*

Na začátku celého programu se nastaví počet vláken na konstantu *NUMBER\_CARS.* Dále se inicializuje *array* na samé nuly, jakožto počáteční rychlosti stojících aut. Rovnou se tato *array* vloží jako první prvek kolekce *vector*.

Následuje *while* cyklus, který probíhá tak dlouho, dokud poslední prvek v kolekci *array* nemá hodnotu 50. Jinými slovy, dokud poslední auto nenabylo 50kilometrové rychlosti. V tomto cyklu se děje následující:

1. Použije se pragma *parallel* pro definování paralelního běhu vláken.
2. Nadefinuje se *for* cyklus s indexem *i* probíhajícím od 0 do konstanty *NUMBER\_CARS.*
3. Cyklus má následující tělo:
   1. Do pomocné proměnné *car*, která je celočíselného typu, se vloží číslo právě používaného vlákna.
   2. Pokud tato proměnná se rovná 0, pak:
      1. Zeptáme se, zda již první auto v kolekci *array* se nerovná 50.
      2. Pokud ano, pak:
         1. Navýšíme prvnímu auto v kolekci *array* jeho hodnotu o 10.
   3. Pokud má jinou hodnotu než 0, nejedná se tedy o první automobil a konáme následující:
      1. Pokud první auto nemá hodnotu 0, pak:
         1. Pokud auto na indexu *car* má jinou hodnotu než jeho předchůdce (*car-1)*, pak:
            1. Provedeme for cyklus od 0 do aktuálního auta a v tomto cyklu, pokud již dané auto nemá rychlost 50, všem autům navýšíme rychlost o 10.
4. Vytvoříme novou pomocnou kolekci *array*, do které vložíme naši dosavadní *array.* To z důvodu zabránění referencím na naši používanou kolekci.
5. Nadefinujeme pragmu pro kritickou sekci, ve které naši pomocnou kolekci *array* vložíme do kolekce *vector*.

Po ukončení *while* cyklu následuje pouze metoda *print()*, ve které vypíšeme naši kolekci *vector*. Výsledkem simulace pro například 5 aut by měla být následující tabulka.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Car 1 | Car 2 | Car 3 | Car 4 | Car 5 |
| 0 | 0 | 0 | 0 | 0 |
| 10 | 0 | 0 | 0 | 0 |
| 20 | 10 | 0 | 0 | 0 |
| 30 | 20 | 10 | 0 | 0 |
| 40 | 30 | 20 | 10 | 0 |
| 50 | 40 | 30 | 20 | 10 |
| 50 | 50 | 40 | 30 | 20 |
| 50 | 50 | 50 | 40 | 30 |
| 50 | 50 | 50 | 50 | 40 |
| 50 | 50 | 50 | 50 | 50 |

Metoda *print()* má kromě výpisu ještě jeden úkol. Práce s vlákny je nevyzpytatelná a často se stávalo, že byly některé řádky ve výsledné tabulce duplicitní. Proto před výpisem kolekce *vector* se provede operace pro odstranění duplicitních prvků s této kolekce.