Проект

по Системи за Паралелна Обработка

Паралелен aлгоритъм за изчисляване на числото π

Изготвил:

Ивайло Николаев Арабаджиев

Ръководител:

ас. Христо Христов

Проверил: ........................

(ас. Христо Христов)

1. ***Условие на задачата.***

Числото (стойността на) π може да бъде изчислена по различни начини. Разглеждаме следния метод за приближено пресмятане на π:

1) Имаме окръжност, вписана в квадрат

2) Генерираме по произволен начин точки в квадрата

3) Определяме броя на точките, които се намират в окръжността

4) Нека **k** е число равно на броя на точките в окръжността, разделен на броя на точките в квадрата

5) Тогава π **~ 4 \* k**

Бележка: Колкото повече точки генерираме, толкова по-добра ще бъде точността, с която пресмятаме π.

По-долу е представен сериен псевдокод реализиращ този метод:

**=== cut ===**

**npoints = 10000**

**circle\_count = 0**

**do j = 1,npoints**

**generate 2 random numbers between 0 and 1**

**xcoordinate = random1**

**ycoordinate = random2**

**if (xcoordinate, ycoordinate) inside circle**

**then circle\_count = circle\_count + 1**

**end do**

**π = 4.0\*circle\_count/npoints**

**=== cut ===**

Задачата е да се напише програма за изчисление на числото π по описания метод, която използва паралелни процеси (нишки). Изискванията към програмата са следните:

(o) Размерността на квадрата се задава в точки (пиксели) от подходящо избран команден параметър – например **“-s 10240”**. Точките в квадрата, генерираме произволно с помощта на **Math.random()** или класа **java.util.Random**.

(о) Друг команден параметър задава максималния брой нишки (задачи), на които разделяме работата по пресмятането на числото π – например **“–t 1”** или **“–tasks 3”**.

(о) Програмата извежда подходящи съобщения на различните етапи от работата си, времето отделено за изчисление, както и резултата от изчислението (стойността на π).

(o) Да се осигури възможност за „quiet“ режим на работа на програмата, при който се извежда само времето отделено за изчисление на π (и самото число), отново чрез подходящо избран друг команден параметър – например **“-q”**.

1. ***Описание на реализирания алгоритъм.***

За паралелното решаване на задачата е избран подходът на многонишковото програмиране. Той предоставя възможност за оптимално намаляване на използвания ресурс памет, тъй като позволява отделните процеси, извършващи изчисленията, да използват едно и също споделено копие на входните данни и да нанасят резултатите върху друг споделен участък памет, който да се ползва без синхронизация, тъй като отделните подпроцеси се нуждаят от различни негови участъци.

Квадратът, в който ще се генерират точките се разделя на вертикални ивици с еднаква големина. Броя им се определя от броя на нишките, които сме задали като входен параметър. Всяка нишка поема работата по генериране на случайни точки в съответната ивица и определя броя на точките, които се намират в кръга, вписан в квадрата. След това получения резултат се записва на отделна позиция в общ масив. Главната нишка изчаква завършването на създадените нишки, след което елементите на масива се сумират и получаваме броя на генерираните точки, които попадат в кръга. След това, използвайки формулата “π = 4.0\*circle\_count/npoints”, получаваме стойността на числото π.

1. ***Тестови замервания и анализ на метричните показатели***

Разработено е приложение на JAVA и е тествано на 8 ядрен мултипроцесор с цел да се оценят ускорението S (забързване, speed-up) и ефективността Е (efficiency) на описания алгоритъм, където ако T(p) е времето необходимо за завършване на работата на алгоритъм с p на брой нишките, то:

S(p) = T(1)/T(p)

E(p) = S(p)/p

Замерванията са направени със следните входни параметри: “-s 10000” за размерността на квадрата и “-p 10000000” за броя на точките, които да се генерират. Резултатите са следните:

По-долу замерванията са направени със следните входни параметри:

“-s 100000” и “-p 100000000”.