СУ „Св. Климент Охридски“



Курсов проект

по

„Системи за Паралелна Обработка“

задачача:

**Пресмятане на *е***

изготвил:

*Венцислав Танев Джукелов*

спец. Компютърни науки, 3-ти курс, ф.н.80915

Ръководител:

ас. Христо Христов

Проверил: ........................

(ас. Христо Христов)

1. *Постановка на задачата*

Едно важно за математиката число е Неперовото число (Ойлеровото число), тоест числото **е**. Използвайки сходящи редове, можем да сметнем стойността на е с произволно висока точност. Един от сравнително бързо сходящите към е редове е:

Вашата задача е да напишете програма за изчисление на числото е използвайки цитирания ред, която използва паралелни процеси (нишки) и осигурява пресмятането на е със зададена от потребителя точност. Изискванията към програмата са следните:

(o) Команден параметър задава точността на пресмятанията. По Ваше желание, точността се изразява или в брой цифри след десетичната запетая или в брой членове на реда. Командният параметър задаващ точността има вида - “-p 10000”;

(о) Друг команден параметър задава максималния брой нишки (задачи) на които разделяме работата по пресмятането на е - например “-t 1” или “-tasks 3”;

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

(o) Записва резултата от работа си (стойността на е) във изходен файл, зададен с подходящ параметър, например “-o result.txt”. Ако този параметър е изпуснат, се избира име по подразбиране;

(o) Да се осигури възможност за „quiet“ режим на работа на програмата, при който се извежда само времето отделено за изчисление на е, отново чрез подходящо избран друг команден параметър - например “-q”;

ЗАБЕЛЕЖКА:

(о) При желание за направата на подходящ графичен потребителски интерфейс (GUI) с помощта на класовете от пакета javax.swing задачата може да се изпълни от двама души; Разработването на графичен интерфейс не отменя изискването Вашата програма да поддържа изредените командни параметри. В този случай към функцията на параметъра параметъра „-q“ се добавя изискването да не пуска графичният интерфейс. Причината за това е, че Вашата програма трябва да позволява отдалечено тестване, а то ще се извършва в terminal.

Уточнения (hints) към задачата:

(о) В условието на задачата се говори за разделянето на работата на две или повече нишки. Работата върху съответната задача, в случаят в който е зададен „-t 1“ (т.е. цялата задача се решава от една нишка) ще служи за еталон, по който да измерваме евентуално ускорение (т.е. това е T1). В кода реализиращ решението на задачата трябва да се предвиди и тази възможност - задачата да бъде решавана от единствена нишка (процес); Пускайки програмата да работи върху задачата с помощта на единствена нишка, ще считаме че използваме серийното решение на задачата; Измервайки времето за работа на програмата при използването на „p“ нишки - намираме Tp и съответно можем да изчислим Sp. Представените на защитата данни за работата на програмата, трябва да отразят и ефективността от работата и, тоест да се изчисли и покаже Ep.

Като обобщение - данните събрани при тестването на програмата Ви, трябва да отразяват Tp, Sp и Ep. Желателно е освен табличен вид, да добавите и графичен вид на Tp, Sp, Ep, в три отделни

графики.

(о) Интересен е въпросът, кога достигаме зададената точност на изчисленията? Тоест кога сме сметнали “е” със зададените от потребителя брой цифри след десетичната точка. Едно добро ограничение за серийната (последователната) програма е разликата между две поредно изчислени стойности на “***е*** ” да е произволно малка.

1. *Описание на реализирания алгоритъм*

Проектът е реализирана на Java с помощта на многонишково програмиране.То спомага за разпределянето на работата на няколко нишки, чрез което се постига по-добро ускорение и ефективност на програмата.

За изчислението са реализирани 2 класа: class eCalculator и class Threads.

1. **Клас Threads:**

Класът Threads се грижи за създаване на нишките и работата им в

програмата. Той наследява класа Thread, който е част от пакета java.lang в Java. В този класа имаме метод *run()*, в който е реализиран програмния код на всяка нишка, т.е. какво да прави нишката, когато я стартираме. При извикването на класа е зададен интервал от кой до кой терм в сумата за изчисление на ***e*** трябвада се пресметне.В метода *run()* се изчисляват от първия до последния терм в този интервал и се прибавя към общата сума. За изчислението ползваме и метода *factorial(),* който пресмята факториела на дадено число. Всеки терм се изчислява чрез предходния и чрез дадената ни формула за изчисление на ***е***.

1. **Клас eCalculation**

Класът eCalculation съдържа алгоритъма за решение на задачата, разпределение работата на нишките и графичен интерфейс. Според желаната от потребителя точност на изчислението и зададения брой нишки, с които то да се извърши, на всяка нишка се заделят равен брой термове за изчисление и сумиране. След като всички нишки приключат работата си, изчислените от тях суми се събират и се проверява точността на получения резултат (неперовото число). Зададената точност на изчисленията достигаме, когато разликата между две поредно изчислени стойности на ‚***е‘***  е произволно малка. Ако точността не е достатъчна, горният механизъм се повтаря отново.

Накрая записваме получената стойност във файл чрез метода *writeToFile().*

За постигане на максимална точност при изчисленията е използвана библиотеката *Apfloat*. Заради големите числови стойности, които се получават при пресмятането на термовете с по-голям номер, се използва класа *BigInteger* за представяне на големи числа.

1. *Тестови замервания и анализ на метричните показатели*

Разработено е приложение на JAVA и е тествано на предоставения сървър d7.yaht.net 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:

Замерванията са направени със следните входни параметри: “-p 10000” знака след десетичната точка и “-t” приемащо стойности от 1 до 12 . Резултатите са следните:

|  |  |  |  |
| --- | --- | --- | --- |
| Брой нишки | Време в ms | Ускорение | Ефективност |
| 1 | 51707 | 1 | 1 |
| 2 | 25047 | 2,06439893 | 1,032199465 |
| 3 | 16970 | 3,046965233 | 1,015655078 |
| 4 | 12731 | 4,061503417 | 1,015375854 |
| 5 | 10826 | 4,776186957 | 0,955237391 |
| 6 | 9590 | 5,391762252 | 0,898627042 |
| 7 | 8786 | 5,885158206 | 0,840736887 |
| 8 | 7981 | 6,47876206 | 0,809845257 |
| 9 | 7506 | 6,888755662 | 0,765417296 |
| 10 | 7632 | 6,775026205 | 0,677502621 |
| 11 | 7117 | 7,265280315 | 0,660480029 |
| 12 | 7075 | 7,308409894 | 0,609034158 |
| 13 | 6972 | 7,416379805 | 0,570490754 |
| 14 | 6595 | 7,840333586 | 0,560023828 |
| 15 | 6908 | 7,485089751 | 0,499005983 |
| 16 | 6921 | 7,471030198 | 0,466939387 |