# Проект по Системи за Паралелна Обработка

# **Паралелно изчисляване на Неперовото число чрез сходящи редове**

Изготвил:

Антонио Николов, Фн:80867, КН, 4та група, 3ти курс

1. **Условие на проекта**

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



Напишете програма за изчисление на числото *e* използвайки цитирания ред, която използва паралелни процеси (нишки) и осигурява пресмятането на *e* със зададена от потребителя точност. Изискванията към програмата са следните:

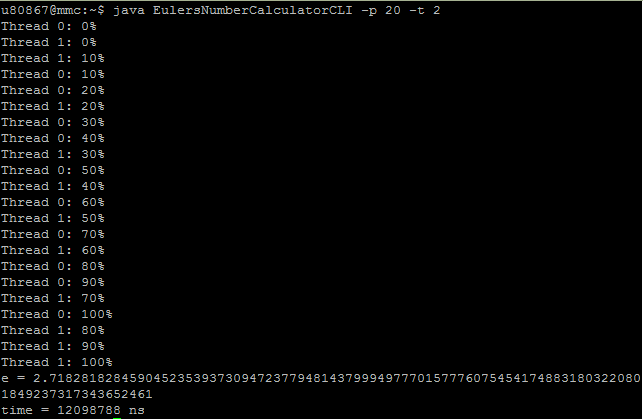
* Команден параметър задава точността на пресмятанията. По Ваше желание, точността се изразява или в брой цифри след десетичната запетая или в брой членове на реда. Командният параметър задаващ точността има вида ­ “­p 10000”;
* Друг команден параметър задава максималния брой нишки (задачи) на които разделяме работата по пресмятането на *e* – например “–t 1” или “–tasks 3”;
* Програмата извежда подходящи съобщения на различните етапи от работата си, както и времето отделено за изчисление и резултата от изчислението (стойността на *e*);
* Записва резултата от работа си (стойността на *e*) във изходен файл, зададен с подходящ параметър, например “­o result.txt”. Ако този параметър е изпуснат, се избира име по подразбиране;
* Да се осигури възможност за „quiet“ режим на работа на програмата, при който се извежда само времето отделено за изчисление на *e*, отново чрез подходящо избран друг команден параметър – например “­q”;

1. **Общо описание на представения алгоритъм**

Приложението за намиране на Неперовото число е осъществено, посредством употребата на езика за програмиране JAVA и срествата, които той предоставя за многонишково програмиране. Проекта е съставен от два класа:

EulersNumber – в който се намира алгоритъма за изчисляване на Неперовото число. По подадената прецизност от потребителя (брой на членове на реда) се образуват n на брой нишки (n е зададено от потребителя) като работата им се разделя равномерно – всяка нишка изчислява равен брой елементи от реда. Като работата на дадена нишка приключи резултата се прибавя към крайното решение. За по-голяма прецизност е използван BigDecimal класа за съхранение на резултатите.

EulersNumberCalculatorCLI – който отговаря за подадените от потребителя аргументи и също в него се извършва целия конзолен интерфейс – извеждане на информацията на екрана (прогреса, крайния резултат и времето за изпълнение). Този клас отговаря и за записването на резултата във файл.



**Фиг.1** Конзолния вид на приложението

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

Приложението е тествано на 16 ядрен мултипроцесор с цел да се оценят ускорението S и ефективността Е на описания алгоритъм, където T(p) е времето необходимо за завършване на работата на алгоритъмa с p на брой нишки то:

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

E(p) = S(p)/p

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Неперовото число – тестови резултати** | | | | | |
| **№** | **прецизност (-p)** | **Брой нишки (-t)** | **Изчислено време T(p) [ms]** | **Изчислено ускорение S(p)** | **Изчислена ефективност E(p)** |
| 1 | 1000 | 1 | 2054 |  |  |
| 2 | 1000 | 2 | 2020 | 1.01 | 0.51 |
| 3 | 1000 | 4 | 1503 | 1.36 | 0.34 |
| 4 | 1000 | 5 | 1381 | 1.48 | 0.30 |
| 5 | 1000 | 8 | 1090 | 1.88 | 0.24 |
| 6 | 1000 | 10 | 982 | 2.09 | 0.21 |
| 7 | 1000 | 20 | 924 | 2.22 | 0.11 |
| 8 | 1000 | 25 | 876 | 2.34 | 0.09 |
| 9 | 1000 | 40 | 1024 | 2.00 | 0.05 |
| 10 | 1000 | 50 | 1075 | 1.91 | 0.03 |