Проект по Системи за паралелна обработка

Пресмятане на числото e

Александър Пирнарев
3 курс, Компютърни науки
поток 1, група 3, 81069

Ръководител: ас. Христо Христов

Проверил:———————
(подпис)

Задача

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:

$$e = \sum \frac{2k+1}{(2k)!}, k = 0, ..., \infty$$

Вашата задача е да 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 10240”;

- 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);
  Примери за подходящи съобщения:
  „Thread-<num> started.“,
  „Thread-<num> stopped.“,
  „Thread-<num> execution time was (millis): <num>“,
  „Threads used in current run: <num>“,
  „Total execution time for current run (millis): <num>“ и т.н.;

- 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”;

Описание на алгоритъма

Алгоритъмът, който изчислява Неперовото число с точност след десетичната запетая, зададена от потребителя, използва подхода на многонишковото програмиране. Използват се нишки, които изчисляват отделни части от реда. Разпределението на тези части е приблизително еднакво, така всяка нишка извършва еднаква работа. Всяка нишка връща резултат, който се натрупва в променлива, която ни дава крайния вид на Неперовото число. Не се изисква от нишките да се изчакват помежду си, така на практика изчисленията са паралелни.
Приложението е написано на Java. Главната програма определя от комадните параметри, или от зададените по подразбиране как ще протече смятането. С помощта на параметрите разделя сумата на подсуми. Всяка нишка изчислява сума със зададени граници. Програмата изчаква

всички нишки да приключат работа, за да изведе резултата, който е сумата на резултатите от нишките. По време на изпълнението на програмата, за всяка нишка се извежда информация – за стартиране, приключване и време, за което е била активна. Всяка нишка първо изчислява (2*началото на сумата + 1) делено на факториела на (2*началото). Следващите елементи на сумата до края й са представени като към числителя се прибавя 2, а знаменателя се умножава с 2i*(2i-1), където i е индекса на текущия елемент от реда. Резултатът от програмата се записва във файл с име 'result.txt', ако не е зададено друго чрез командните параметри. Приложението има и “тих” режим, в който се извежда само времето за изпълнение. За точността е използвана BigDecimal библиотеката на Java, а за изчислението на фактрориела е написана помощна функция.

Ускорение, ефективност, бързодействие

Тестовете са пускани на сървър - cf.rmi.yaht.net. Ползвани са следните формули:

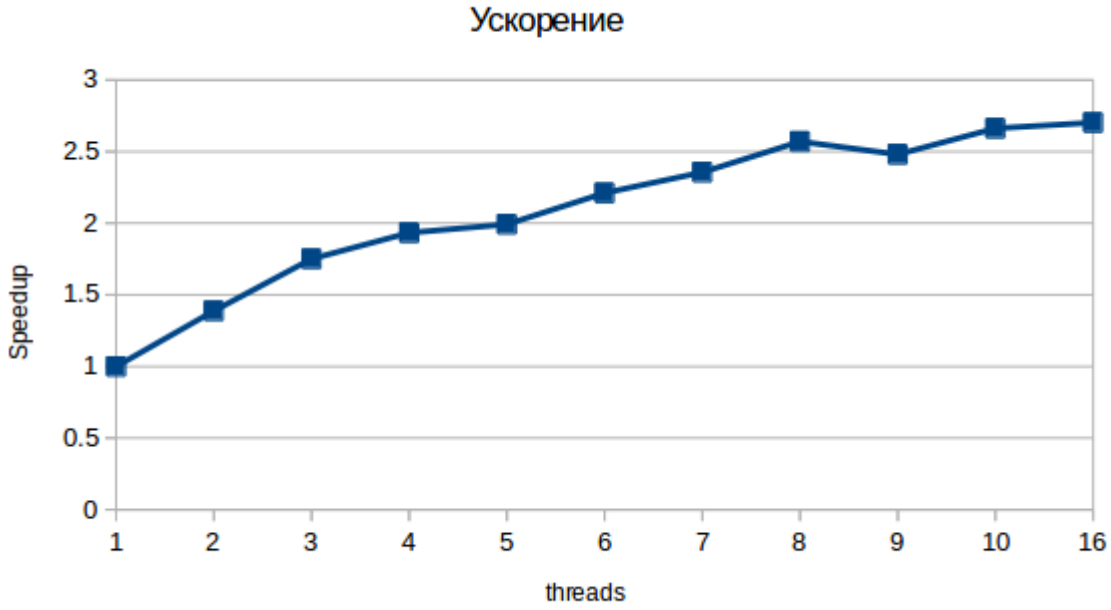
S(p) = T(1)/T(p) – ускорение на програмата при p на брой нишки
E(p) = S(p)/p – ефективност на програмата при p на брой нишки
Tp – време за изпълнение на програмата при p на брой нишки

Направени са тестове с 2 различни стойности на p. Резултатите ни показват, че при по-големи изчисления приложението се възползва по-добре от повечето нишки.

precision = 10 000

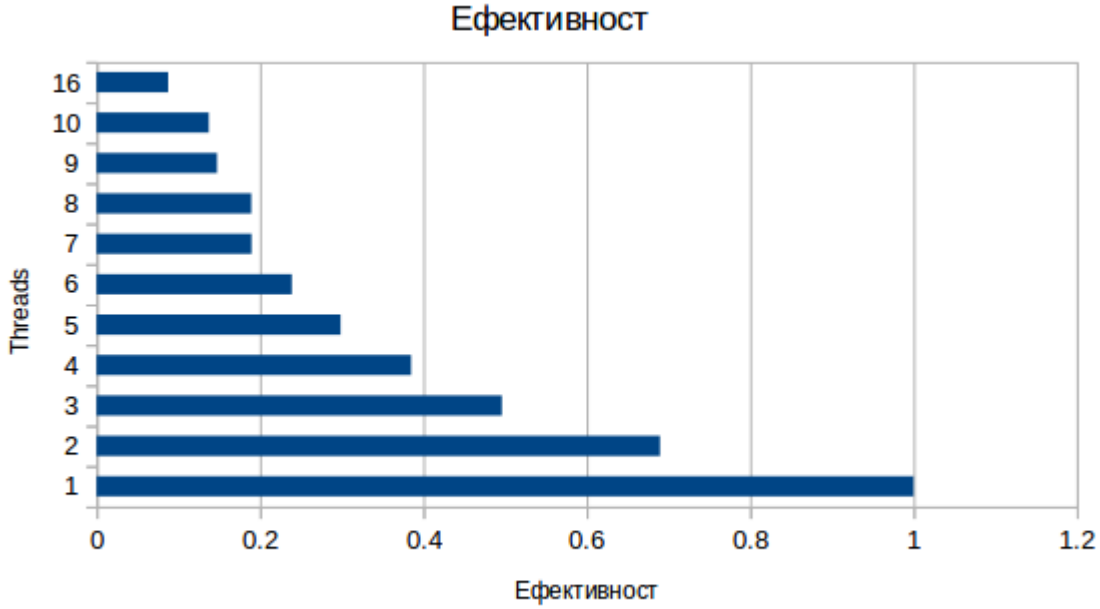
брой нишки	време за изпълнение	ускорение	ефективност
1	3927	1	1
2	2830	1.387632509	0.693816254
3	2243	1.750780205	0.583593402
4	2031	1.933530281	0.48338257
5	1971	1.99238965	0.39847793
6	1776	2.211148649	0.368524775
7	1667	2.355728854	0.336532693
8	1528	2.570026178	0.321253272
9	1583	2.480732786	0.275636976
10	1475	2.662372881	0.266237288
16	1453	2.702684102	0.168917756

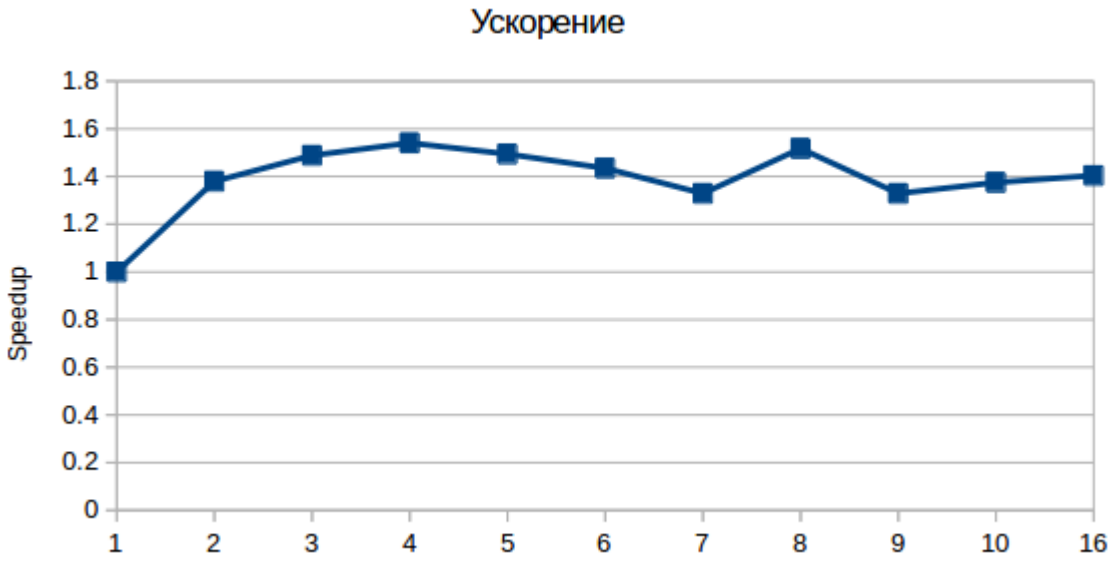

Ефективност
16
10
9
8
7
6
5
4
3
2
1
Threads
0
0.2
0.4
0.6
0.8
1
1.2
Ефективност


Ускорение
3
2.5
2
1.5
1
0.5
0
Speedup
1
2
3
4
5
6
7
8
9
10
16
threads

precision = 4200

брой нишки	време за изпълнение	ускорение	ефективност
1	396	1	1
2	287	1.379790941	0.68989547
3	266	1.488721805	0.496240602
4	257	1.540856031	0.385214008
5	265	1.494339623	0.298867925
6	276	1.434782609	0.239130435
7	298	1.32885906	0.189837009
8	261	1.517241379	0.189655172
9	298	1.32885906	0.147651007
10	288	1.375	0.1375
16	282	1.404255319	0.087765957




Ускорение
Speedup
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
1
2
3
4
5
6
7
8
9
10
16


Време за изпълнение
Threads
16
10
9
8
7
6
5
4
3
2
1
0
50
100
150
200
250
300
350
400
450
Време на изпълнение