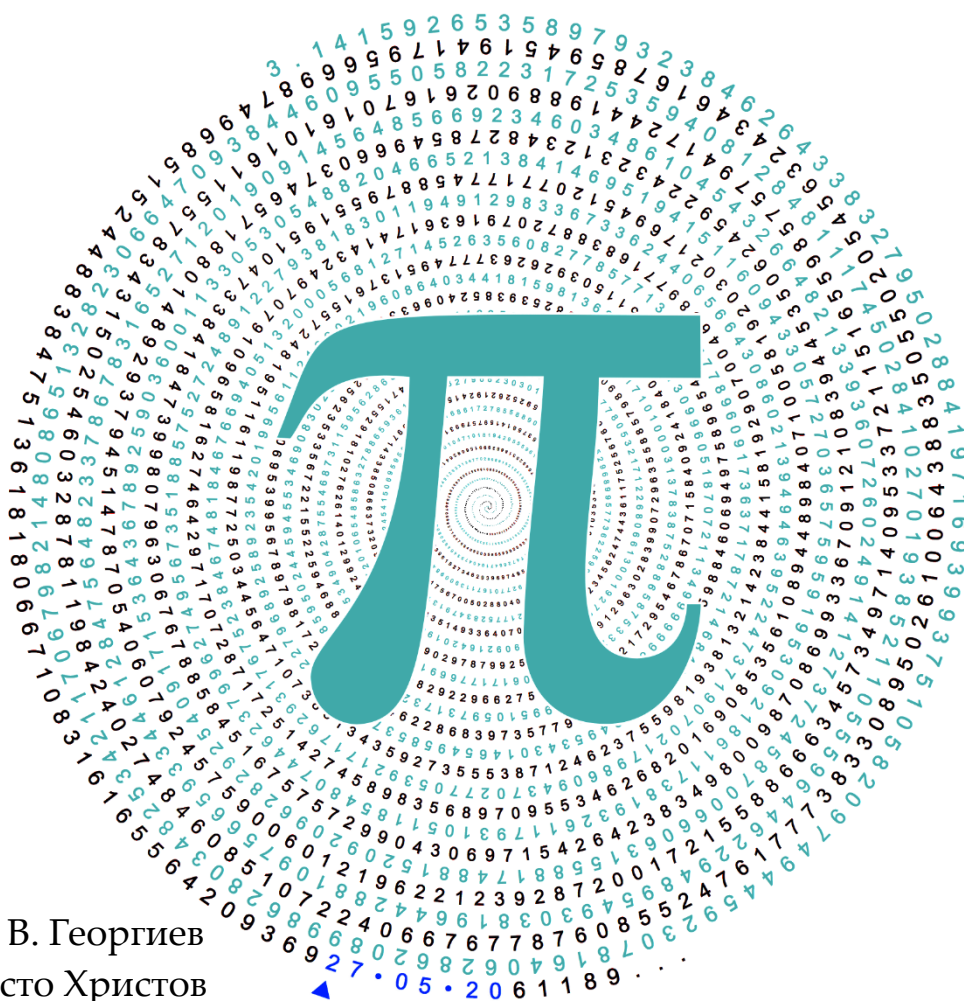




Проект по Системи за Паралелна Обработка

2020

Пресмятане на π чрез реда на братя Чудновски



Лектор: проф. В. Георгиев

Асистент: Христо Христов

Изготвил: Теодор Грозданов, 4. курс КН, 81560

27.05.2020

1. Задача

Използвайки реда на Чудновски да се изчисли с висока точност стойността на числото π .

$$\frac{1}{\pi} = 12 \sum_{k=0}^{\infty} \frac{(-1)^k (6k)! (545140134k + 13591409)}{(3k)! (k!)^3 (640320)^{3k+3/2}}$$

2. Описание на решението

Забелязвам, че чрез един елемент от сумата можем да намерим следващия елемент чрез умножение и събиране, като не е нужно да изчислявам отново функциите факториел и вдигане на висока степен.

За удобство ще разделя всеки елемент на сумата на три части:


$$a_k = \frac{\overset{a_{k_1}}{(-1)^k} \overset{a_{k_{2,1}}{(6k)! (545140134k + 13591409)}}{\underset{a_{k_{2,2}}{(3k)! (k!)^3 (640320)^{3k+3/2}}}}$$

За да пусна различни нишки ще разбия на равни части сумата и за всяка част ще сметна един елемент, за да изчислявам чрез него следващите. Използвам допирните точки между частите

на сумата, за да тръгне една нишка от този елемент да получава следващите надясно чрез събиране и умножение, а друга да получава следващите наляво чрез изваждане и деление. Така намалявам работата по намиране на елементи директно от формулата на Чудновски.

Пример за 100 елемента на сумата и 4 нишки:

Разбиване на сумата за първите 100 ел. : [0 - 24] [25 - 49] [50 - 74] [75 - 99]
Нишка номер $i = 1, 2, 3, 4$:

 - елементи на сумата, които ще пресметнем от формулата на Чудновски

3. Описание на програмата

Програмата е реализирана на *Java* и използва външната библиотека *Apfloat* за пресмятане на числа с произволна точност. Представява декомпозиране по данни и *master/slave* отношение на *TaskRunner* и *SumRunnable* (преизползвани от упражнението от 8-ма седмица).

NB Боравенето с *Apfloat* обекти с висока точност може да доведе до значително забавяне на изпълнението на програмата. Ще тествам програмата на две различни машини с два вида точност 500 знака и 10 000 знака.

tr

Примерно компилиране и стартиране:

```
[u81560@t5600 proj]$ javac -cp ../libs/apfloat.jar ./*.java
[u81560@t5600 proj]$ java -cp ../libs/apfloat.jar TaskRunner -p 15009 -t 20 -o myfile
Iterations: 15009
Threads: 20
Time: 1.486928791 sec
Error: 1.94217035552245e-496
```

4. Коректност

За да съм сигурен че нишките извършват верни изчисления по намирането на следващ елемент от сумата от вече пресметнат предишен елемент и няма препокриване на индекси, сравнявам резултатите от изпълнение на 1 нишка, на 6 нишки и директно пресмятане реда на Чудновски (от метода *calculateDummy*). Сравненията извършвам по разликата в резултатите им от очаквания резултат, а именно $1/(12*\pi)$, като числото π взимам от *ApfloatMath* с точност 10 000 знака.

```
public static void main(String[] args) {
    //testThreadBenefit();
    verifyCalculations( sumEnd: 31);
}

private static void verifyCalculations(int sumEnd) {
    System.out.println("Sum end index: " + sumEnd);
    String[] arg1 = {Integer.toString(sumEnd), "1"};
    Apfloat a = calculate(arg1);

    String[] arg6 = {Integer.toString(sumEnd), "6"};
    Apfloat b = calculate(arg6);

    Apfloat c = calculateDummy(sumEnd);

    Apfloat realPi = Apfloat.ONE.divide(ApfloatMath.pi(Constants.PRECISION).multiply(new Apfloat( value: 12)));
    System.out.println(ApfloatMath.abs(a.subtract(realPi).precision(20)));
    System.out.println(ApfloatMath.abs(b.subtract(realPi).precision(20)));
    System.out.println(ApfloatMath.abs(c.subtract(realPi).precision(20)));
}
```

Разликата с π намалява с увеличаване на броя на елементите на сумата и разликите са еднакви независимо от броя нишки и вида на формулата (директно или косвено получаване на елементите). Получените резултати са задоволителни:

```
Sum end index: 3
Threads: 1
Time: 0.378571801 sec
Threads: 6
Time: 0.423669299 sec
Dummy
Time: 0.3383163 sec
1.452709576470933672e-44
1.452709576470933672e-44
1.452709576470933672e-44
```

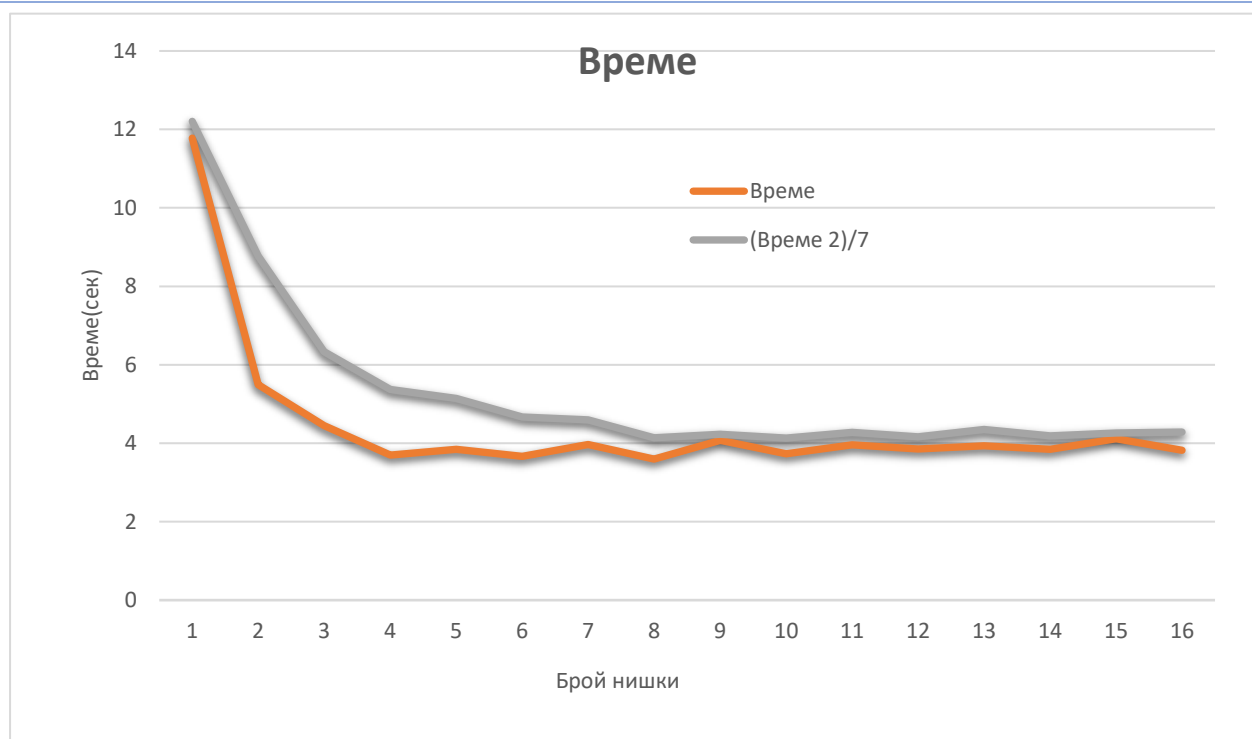
```
Sum end index: 31
Threads: 1
Time: 0.4565696 sec
Threads: 6
Time: 0.480514699 sec
Dummy
Time: 2.731181499 sec
3.9816022704589899102e-442
3.9816022704589899102e-442
3.9816022704589899102e-442
```

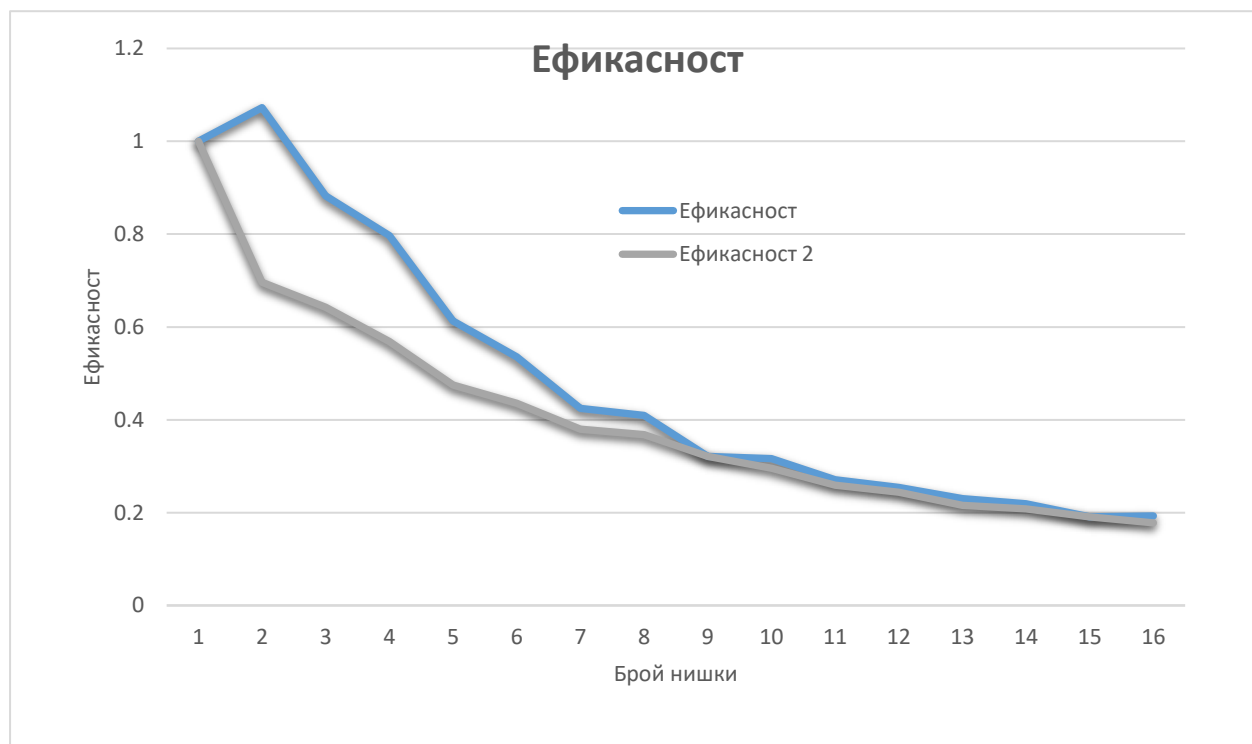
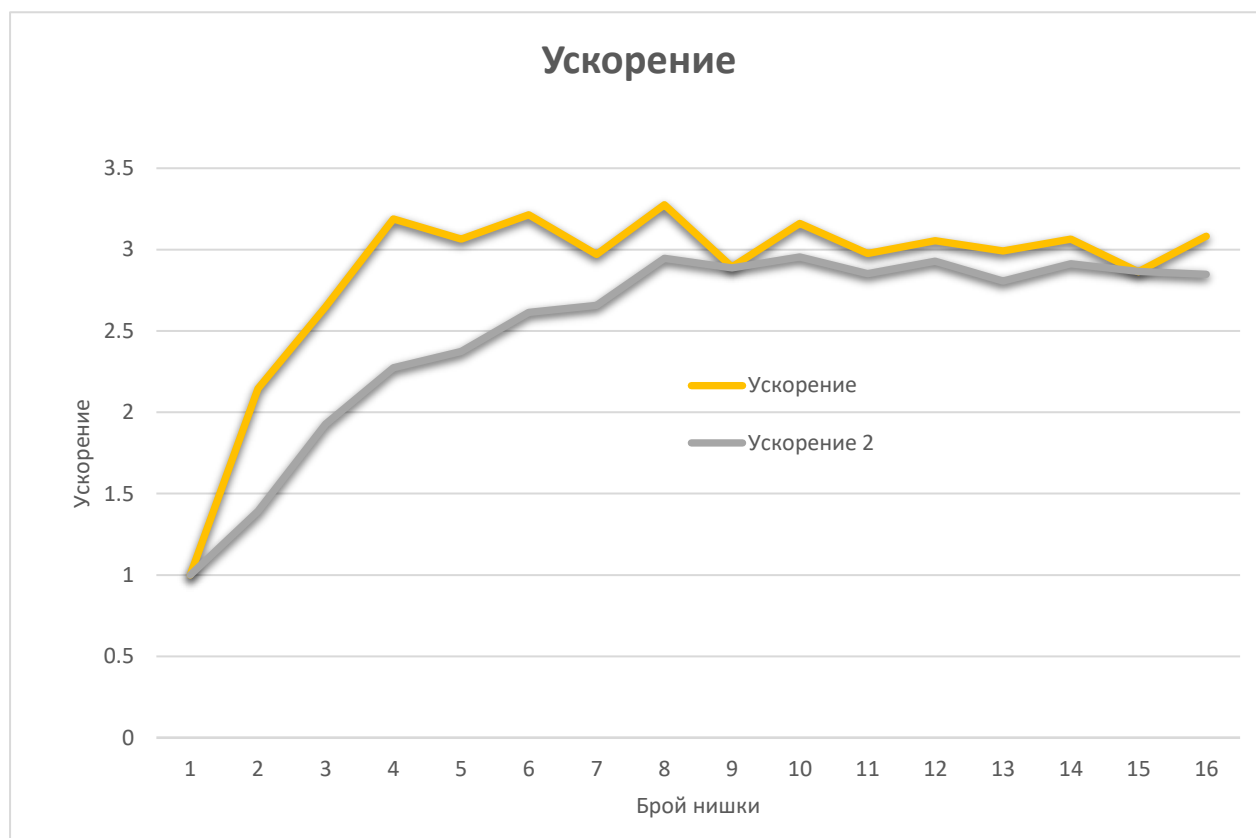
```
Sum end index: 41
Threads: 1
Time: 0.531274899 sec
Threads: 6
Time: 0.5119726 sec
Dummy
Time: 3.755286 sec
5.2928387275807940893e-584
5.2928387275807940893e-584
5.2928387275807940893e-584
```

5. Резултати

Изпитания за време за пресмятане на първите 12000 члена извършени на машина с процесор *intel—i5 8265u* с 4 ядра и 8 нишки. Първият тест е с точност на сметките до 500 знака. Вторият с точност 10 000 знака.

Нишки	Време	Ускорение	Ефикасност	Време 2	(Време 2)/7	Ускорение 2	Ефикасност 2
1	11.7773276	1	1	85.4412556	12.20589366	1	1
2	5.490730499	2.144947	1.072473654	61.3925467	8.770363814	1.391720334	0.695860167
3	4.4489005	2.647245	0.88241485	44.3173459	6.331049414	1.927941619	0.642647206
4	3.6943856	3.187899	0.796974712	37.5804888	5.368641257	2.2735536	0.5683884
5	3.8427839	3.064791	0.612958101	35.9847071	5.140672443	2.374376853	0.474875371
6	3.6643851	3.213998	0.535666389	32.6798525	4.668550357	2.614493306	0.435748884
7	3.965468	2.969972	0.424281666	32.1575011	4.593928729	2.656961912	0.379565987
8	3.5955178	3.275558	0.409444768	28.9943687	4.142052671	2.946822415	0.368352802
9	4.0711846	2.89285	0.321427811	29.5822348	4.226033543	2.88826237	0.320918041
10	3.7268116	3.160162	0.316016179	28.9182509	4.1311787	2.954578958	0.295457896
11	3.9583405	2.975319	0.270483589	29.9657027	4.280814671	2.851301585	0.259209235
12	3.8562676	3.054074	0.254506188	29.1663133	4.166616186	2.929449969	0.244120831
13	3.9358543	2.992318	0.230178306	30.4411444	4.348734914	2.806768841	0.215905295
14	3.8431415	3.064505	0.218893238	29.33591	4.190844286	2.912514239	0.208036731
15	4.111335301	2.864599	0.190973277	29.8220002	4.260285743	2.865041078	0.191002739
16	3.820138501	3.082958	0.192684892	29.9991539	4.285593414	2.84812218	0.178007636

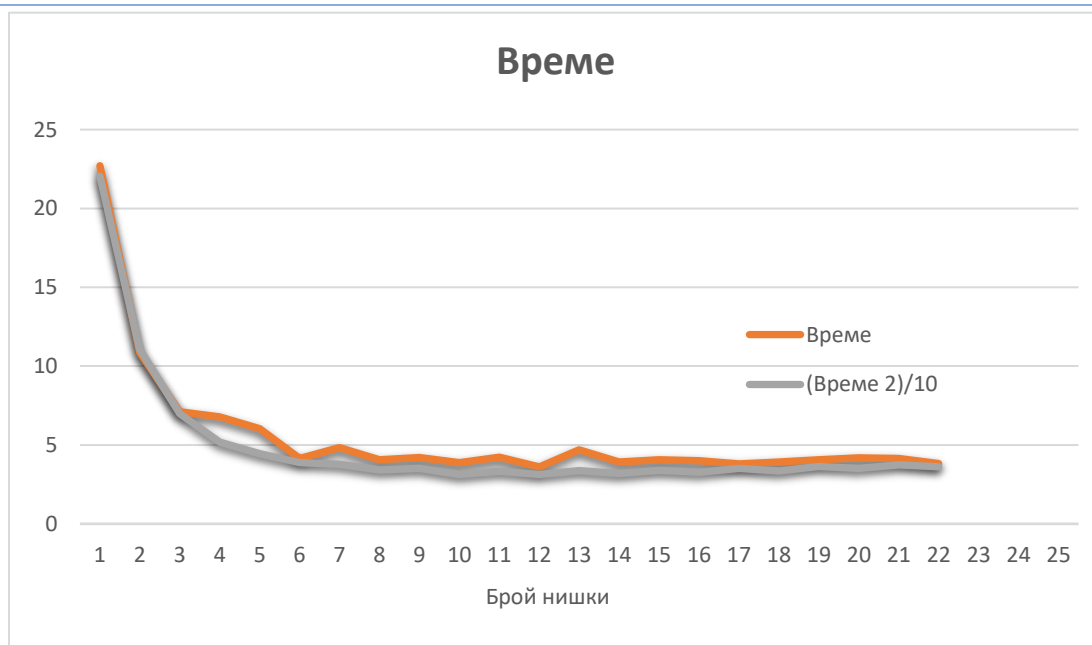


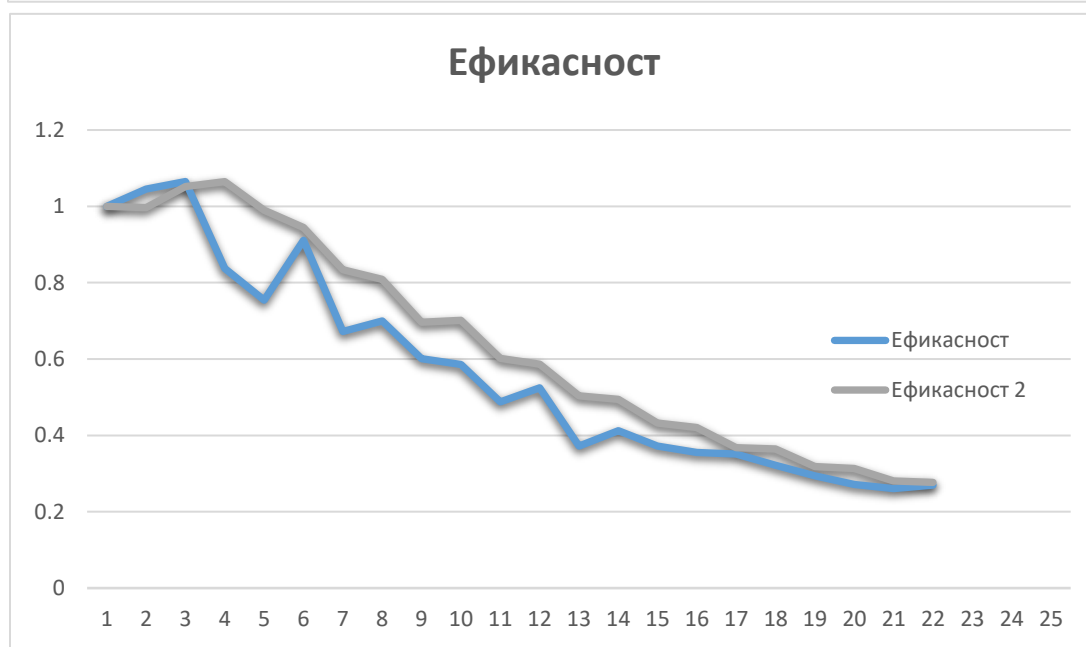


```
[u81560@t5600 proj]$ java -cp ../libs/apfloat.jar TaskRunner -t 22 -p 12000 -apfloatPrecision 3000 test
[u81560@t5600 proj]$ java -cp ../libs/apfloat.jar TaskRunner -t 22 -p 12000 -apfloatPrecision 10000 test
```

Изпитания за време за пресмятане на първите 12000 члена извършени на машина на адрес t5600.rmi.yaht.net. Първият тест е с точност на сметките до 3000 знака. Вторият с точност 10 000 знака.

Нишки	Време	Ускорение	Ефикасност	Време 2	(Време 2)/10	Ускорение 2	Ефикасност 2
1	22.70793	1	1	220.3973	22.0397344	1	1
2	10.86497	2.090014	1.045007	110.5837	11.05837314	1.993035876	0.996517938
3	7.107427	3.194958	1.064986	69.83245	6.983244624	3.156087977	1.052029326
4	6.77808	3.350201	0.83755	51.74786	5.17478561	4.259062318	1.06476558
5	6.023116	3.77013	0.754026	44.53262	4.453261556	4.949121924	0.989824385
6	4.15559	5.464429	0.910738	38.88728	3.888727571	5.667595376	0.944599229
7	4.828304	4.703086	0.671869	37.76763	3.77676312	5.835614704	0.833659243
8	4.056742	5.597578	0.699697	34.05491	3.405490567	6.471823652	0.808977956
9	4.198215	5.408949	0.600994	35.16375	3.516374872	6.267743117	0.696415902
10	3.877344	5.856568	0.585657	31.41042	3.14104227	7.01669462	0.701669462
11	4.236248	5.360387	0.487308	33.29761	3.329760709	6.619014496	0.601728591
12	3.609561	6.29105	0.524254	31.29826	3.129826472	7.041839092	0.586819924
13	4.693343	4.838327	0.372179	33.67735	3.36773453	6.544379969	0.503413844
14	3.931605	5.77574	0.412553	31.8609	3.186090093	6.917486247	0.494106161
15	4.071592	5.577162	0.371811	33.98131	3.398130619	6.485840855	0.43238939
16	3.998498	5.679115	0.354945	32.74233	3.274233426	6.73126547	0.420704092
17	3.801524	5.973375	0.351375	35.25574	3.525574424	6.251388213	0.367728718
18	3.924545	5.78613	0.321452	33.56295	3.356294725	6.566686244	0.364815902
19	4.067734	5.582452	0.293813	36.41436	3.641436483	6.052483546	0.318551766
20	4.18277	5.428921	0.271446	35.15337	3.515336627	6.269594276	0.313479714
21	4.148766	5.473418	0.260639	37.41441	3.741441311	5.890706968	0.280509856
22	3.831238	5.927047	0.269411	36.18135	3.618135387	6.09146205	0.276884639





6. Забележки

В момента известна част от работата е оставена в *Master* класа – *TaskRunner*. В него се извършва директното пресмятане на членове на сумата поради това, че тази стойност е се предава на две нишки, за да се спести двойното пресмятане на функции като факториел и високи степени. Възможно е да се изнесе и тази дейност в отделна нишка, която да пресмята този елемент на сумата и да пуска две нишки с тази стойност (*master/slave* с три слоя).