

Курсов проект

по

Разпределени софтуерни архитектури

летен семестър, учебна година 2019/2020

**Тема 17**

**Изобразяване на фрактал по формулата**

Ръководители: проф. Васил Цунижев, ас. Христо Христов

Изготвил: Иван Чучулски

Факултетен номер: 62167

Специалност: Софтуерно инженерство

Дата: Проверил :……………………………..

1. **Анализ**
   1. **същност на проблема**

Фракталите са математически обекти, които представляват множество от точки върху комплексната равнина. Те са детайлно изследвани за пръв път в средата на 20-ти век от математика Беноа Манделброт. Фракталите намират приложение в моделирането на структурата на природни обекти и взаимодействията между тях, фрактална графика за компютърни изображения, анализ и компресия на сигнали, метод за създаване на картини и архитектурен дизайн и други.

Целта на проекта е да се създаде програма, която да визуализира фрактал, който е зададен чрез рекурентна формула **.**

Реализирана е програма на Java, която използва асинхронна паралелна обработка на escape-time алгоритъм за пресмятане и визуализиране на множеството на Манделброт по зададената рекурентна формула. Направена е декомпозиция на данните SPMD, като е използвана средна грануларност при формиране на заданията – броят на заданията е равен на броя на редовете на изображението. Имплементирани са два начина за балансиране на изпълнението : статично и динамично с централен процес.

* 1. **преглед на функционалността на приложения, които пресмятат множеството на Манделброт (функционален анализ)**

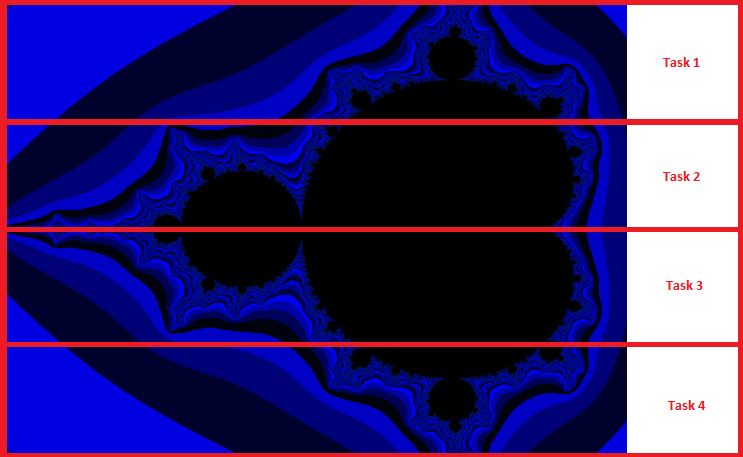
Източник [1] представя математическа дефиниция на множеството на Манделброт и escape-time алгоритъма за пресмятането и оцветяването на множеството. За да определим дали точка от комплексната равнина принадлежи на множеството на Манделброт, трябва да пресметнем границата на редицата, чиито членове се получават при прилагане на зададената рекурентна формула започвайки от точката . Ако границата на редицата от комплексни числа не клони към безкрайност при пресмятане на безбройно много членове на тази редица, то точката принадлежи на множеството. Разбира се, безкрайността не може да бъде точно моделирана в компютърните изчисления, затова идеята на escape-time алгоритъмът е да използваме големината на модула на комплексното число за определяне дали точката принадлежи на множеството. Задаваме число, което обозначава максималния брой итерации, които правим на всяка точка и пресмятаме броя на итерациите, за които модулът на числото остава по-малък от дадена стойност. Отношението на максималния брой итерации и броя итерации за дадена точка можем да използваме за задаване на цвят на точката при генериране на изображението. В [1] е представена реализация на последователна програма, която използва escape-time алгоритъма на езика C#.

Източник [2] и източник [3] представят два начина за имплементиране на паралелна реализация на escape-time алгоритъма на езика C. Първо е направена декомпозиция по данни SPMD и статично балансиране на изпълнението. Създаването и управлението на паралелните процеси става по модела Master-Slaves. Master частта разделя изображението на блокове, които се състоят от реда и колони. След това създава изпълняващите процеси и ги пуска. Първата нишка обработва първия такъв блок, втората работи върху втория и така нататък, докато всички не приключат.

След това източник [2] дава идеята за прилагане на различни схеми за статично балансиране на изпълнението. За статичното e представен принципа “round robin”, т.е. нишките се редуват в разпределянето на задачите. Представени са и два начина за правене на динамично балансиране – централизирано и децентрализирано. Първия е Master-Slaves модел : заданията се разпределят по време на изпълнение от един процес, наречен Master на множеството обработващите процеси наречени Slaves. Между slave процесите няма комуникация и обмен на данни. При втория начин всички задания първоначално се разпределят между обработващите процеси и в хода на изпълнение те могат да дават задания помежду си, с цел да се уеднакви натоварването.

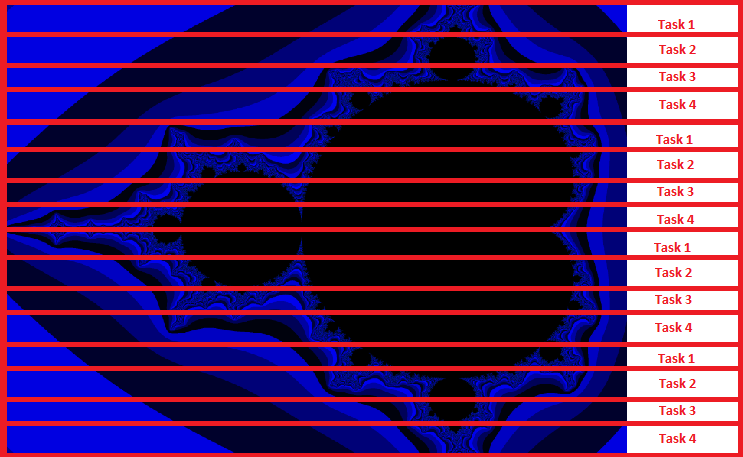
В източник [3] е представено решение, като грануларността е средна и е използвано динамично балансиране с централен процес, т.е. топология звезда. Изображението е разделено на задания, като заданията представляват хоризонталните редове на изображението. Master частта създава масив, която да представлява заданията, след това създава обработващите процеси и ги пуска да работят. Те трябва да достъпват масива със заданията, за да вземат задание, след това го да го обработят и отново да потърсят задание, докато няма повече задания. Master частта изчаква всички Slaves да приключат и след това програма приключва. Обозначено е, че достъпването на масива със заданията трябва да не наруши структурата от данни и следователно обработващите процеси трябва да осигурят синхронизация помежду си, когато взимат задание.

Източник [4] представлява имплементация на първата идея от източник [2] и [3] на езика Java,т.е. Master-Slaves програма, при която е направена SPMD декомпозиция по данните на блокове с големина реда и колони, и е използвано статично балансиране на изпълнението.



Фигура 1. Декомпозиция на данните на блокове и статично балансиране на изпълнението, източник [3]

Също така е представена втора имплементация на Master-Slaves програма, при която е направена SPMD декомпозиция по данните и използвана по-фина грануларност, като са задавани различен брой редове, които да се съдържат в даден блок. Балансирането на изпълнението отново е статично.



Фигура 2. Декомпозиция на данните по редове и статично балансиране на изпълнението, източник [3]

Следната сравнителна таблица показва характеристиките на паралелните решения, разгледани в източниците. Всички те представят асинхронна паралелна имплементация на escape-time алгоритъма по модела Master-Slaves.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Образец | Декомпозиция | Грануларност | Балансиране | Коментар |
| Пример 1 от [2] и [3], Пример 1 от [4] | По данни, SPMD | Едра, блокове, които се състоят от по (височина на изображението)/(брой нишки) реда и (ширина на изображението) колони | Статично, всяка нишка обработва блок, който съответства на номера ѝ. Нагледно показано на Фигура 1. | При използването на едра грануларност и статичното балансиране е възможно да се получи сериозен дисбаланс в товара на различните нишки. Тези, които получат области със много точки от фрактала ще се забавят и от там цялата паралелна част. |
| Пример 2 от [2] | По данни, SPMD | Средна, редовете на изображението са оформени като индивидуални задания. | Динамично с централен процес, звездовидна топология. Заданията са в масив, който се съдържа в централният процес. | Средната грануларност и динамичното разпределение на заданията гарантира равномерно разпределяне на товара между нишките. Остава въпрос за оптимизиране на структурата от данни със заданията и начина по който се прави синхронизацията между обработващите процеси. |
| Пример 2 от [4] | По данни, SPMD | Различна грануларност, от едра към средно-едра. Коефициентът на грануларност е вариран между 8, 16 и 32. Изображението се разделя на блокове, която имат по (коефициент на грануларност \* брой нишки) реда и (ширина на изображението) колони. Броят на блоковете е (коефициент на грануларност \* брой нишки). | Статично разпределение. Нишка с индекс t, обработва блок с индекс b, ако  b % (брой нишки ) == t.  Нагледно показано на Фигура 2. | По-добър вариант от Пример 1 от [4], тъй като по-фината грануларност и „цикличното“ разпределение би „разбило“ секторите, в които има повече точки от множеството на няколко по-малки парчета, които да се обработят от няколко нишки. |

* 1. **нефункционален анализ**
     1. **технологии, използвани в решението**

За реализирането на двете решения е използван езика **Java** версия 8. Стандартната библиотека на езика предлага удобна и проста абстракция на нишка, класа **Thread**, която използваме за имплементиране на изчислителните процеси. От пакета за паралелно програмиране използваме и класа **ConcurrentLinkedQueue**, който служи за опашката от задания. За генерирането на изображението на фрактала е използван класа **BufferedImage**. За оцветяването е използван метод, който превръща модела HSB(Hue, Saturation, Brightness) в RGB(Red, Green, Blue).

За обработването на аргументите от командния ред е използвана библиотеката **Аpache commons-cli-1.4**. С нейна помощ се прави разделянето на опциите и взимането на техните стойности, които се подават при стартиране на програмата. За извършване на изчисленията с комплексни числа е използвана библиотеката **Apache commons-math3-3.5**. От нея са използвани класа, който представя комплексно число **Complex**. В този клас статичните методи за косинус **cos()** и експоненциална функция **exp()**служат за пресмятането на математическите функции, които участват във рекурентната формула на фрактала.

Изборът на Java за език на реализацията дава на програмата доста широка преносимост, тъй като има множество имплементации на JVM(Java Virtual Machine) на повечето съвременни хардуерни и софтуерни платформи. Също така това гарантира безпроблемното използване на паралелна обработка на всяка от тези платформи, като ние можем да използваме API-то предложено от Java, а не специфичното за съответната платформа. Разбира се, това означава допълнително ниво на абстракция, предоставено от Java, което ще доведе до по-ниско ускорение в сравнение с използването на механизмите за паралелна обработка, които например са директно предоставени от ядрото на дадена операционна система.

* + 1. **инфраструктурни изисквания**

За решението е използват софтуерния модел Master-Slaves. Изборът на Java за език за реализиране на решението и използване на предоставените от него абстракции за паралелна обработка може да гарантира сравнително еднаква производителност на различните платформи, които поддържат JVM.

* + 1. **обосновка на избраното решение**

Първоначално беше обмислена реализация на програмата като Пример 1 от източник [2] и Пример 1 от [3], т.е. едра грануларност и статично балансиране. При този метод обаче има няколко проблема като на първо място поради едрата грануларност е възможно някои нишки да получат блок, който се състои главно от точки в множеството, като това означава по-тежки сметки и така тяхното изпълнение ще се забави. Тъй като балансирането е статично, други нишки, които са получили по-лек блок и приключили работата си по-бързо няма как да помогнат на тези, които се бавят.

За решението на тези проблеми първо е използването на по-фина грануларност: средна грануларност, при която заданията няма да бъдат блокове от по няколко реда, а всеки хоризонтален ред от изображението представлява отделно задание.

Също така може да се направи различно статично балансиране, при което нишките няма да обработват последователни редове, а вместо това нишка обработва първо реда, който съответства на нейния индекс, после „прескача“ „брой на нишките“ редове напред, пак обработва един ред и пак „прескача“, докато не се изчерпат редовете.

След това е имплементирана опция на програмата, при която се извършва динамично балансиране при разпределяне на заданията между обработващите процеси. По този начин нишки, на които се паднат по-лесни задания и те успяват да ги обработят бързо, могат да помогнат с обработването на по-интензивни задания, които при статично балансиране биха могли да се паднат на изцяло на една нишка.

1. **Проектиране**
   1. **описание на реализирания алгоритъм**

Имплементиран е асинхронен паралелен алгоритъм, софтуерният модел е **Master**-**Slaves** направена е декомпозиция по данни SPMD. Всеки ред от изображението е отделно задание, например при размер 1920 на 1440 пиксела се обособяват 1440 на брой задания от по 1 пиксел на височина и 1920 пиксела широчина. При бройка на обработващите процеси между 1 и 24, отношението е , което означава средна грануларност. Имплементирани са два начина за балансиране на изпълнението: статично и динамично с централен процес и опашка, от която обработващите процеси взимат заданията.

**Main** класът е в ролята на **Master**. Първоначално се обработват аргументите от командния ред. С тях се задават големината на изображението, крайните точки на комплексната равнина, името на изходния файл, максимален брой итерации на точка, начина на балансиране на изпълнението и дали програмата да работи в „тих режим“, т.е. дали да изписва съобщения за започване и завършване на обработващите процеси или просто времето, за което се е изпълнила програмата. След това се създава обекта от класа **BufferedImage**, който представлява изображението и се запълва с бял цвят. Следва проверка за вида на балансирането, което да бъде използвано.

Общото при двата начина на балансиране е алгоритъмът за изчисляване на точките от множеството на Манделброт и той е базиран на escape-time алгоритъма. При него първо за всеки пиксел от заданието се намира съответстващата му точка в комплексната равнина. За тази комплексна точка в метода **calculateNumberOfSteps()** се изчисляват членовете на редицата, определени от рекурентната формула **.** Ако членовете на тази редица не нарастват към безкрайност, означава че точката принадлежи на фрактала. Когато това е изпълнено, то се извършват максималният брой итерации пъти пресмятания(по подразбиране 500) по формулата и заради това, задания с много точки от множеството изискват най-много изчислителен ресурс. В противен случай, изчисленията завършват при по-малък брой итерации. На база на тази бройка обработващите процеси определят цвета на точката чрез метода **getHSBToRGBColor()** и след това изпълняват метода **drawPixel()**,който записва резултата в изображението. Тъй като обектът на изображението е споделен ресурс във всички нишки, то този метод е **synchronized**. Това означава, че кодът в него е критична секция и само една нишка може да го изпълнява в даден момент от време.

Ако е избрано статично балансиране, то **Main** класът създава обработващите процеси, които са от обекти от класа **StaticSlaveThread**. Те наследяват класа **Thread,** който е от стандартната библиотека на Java и представлява абстракция на нишка. При създаването на всяка нишка **StaticSlaveThread** ѝ се задава индекс и референция към обекта за изображението **BufferedImage**. **Master**-ът ги стартира като извиква метода **start()**, който извиква метода **run()** в **StaticSlaveThread**. В него всяка нишка започва да обработва реда с номер равен на индекса си, след това отива на реда с номер (сегашен номер на ред + брой на нишки) и така отново, докато не надхвърли броя на редовете. Илюстративно това е показано на Фигура 2, но разликата е, че там нишка обработва по няколко реда един след друг, а в нашата програма заданията са по 1 ред. **Master**-ът изчаква всички нишки да приключат работата си, като извиква метода на **Thread,** **join().**

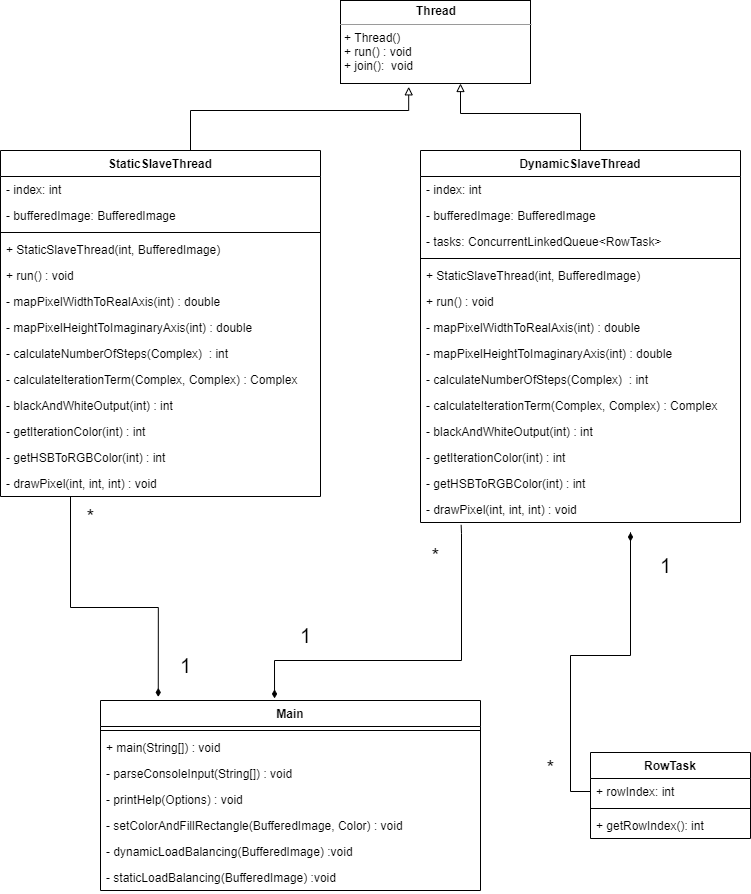
При избрано динамично балансиране **Main** класът първо създава опашката от задания. Заданията са обекти от класа **RowTask**. Този клас е създаден единствено с цел по-ясно дефиниране на абстракцията задание. Той има една член-данна и тя е номерът на реда, който представлява това задание. Самата опашка е обект от класа **ConcurrentLinkedQueue**. След това **Main** класът създава **DynamicSlaveThread** обектите, които са **Slave** процесите и наследяват класа **Thread**. При инициализирането им се задава индекс, референция към обекта за изображението **BufferedImage** и референция към обекта за опашката от задания. **Master**-ът стартира нишките като извиква **start()** метода, който извиква **run()** метода на **DynamicSlaveThread.** В този метод всяка нишка взима поредното задание от опашката със задания, обработва го и записва резултата в изображението и проверява за ново задание в опашката. Ако няма такова, то нишката приключва изпълнението си. **Master**-ът изчаква всички нишки да приключат работата си, като извиква метода им **join().**

Първоначално интерфейсът **Queue** и класът **LinkedList** бяха избрани като структура от данни за опашката от задания. За да се осигури правилното паралелно достъпване на опашката беше използван **synchronized** метод в **DynamicSlaveThread** класа за вземане на задание от опашката. Метод, който е **synchronized** позволява само на една нишка да изпълнява кода му. При увеличаване на броя на нишките, честата синхронизация при вземане на задание забавяше цялата паралелна част на програмата. Затова имплементацията на опашката със задания беше сменена на **ConcurrentLinkedQueue**. Този клас е от пакета **java.util.concurrent**, в който има имплементации на основните структури от данни, подходящи за използване в паралелен алгоритъм, като целостта на данните и правилният резултат от операциите са гарантирани[6].

След тази промяна динамичното балансиране показа равни по ускорение резултати на статичното, а при тестване на по-небалансирани участъци от фрактала полученото ускорение беше по-високо. Това се дължи на факта, че при използването на **synchronized** метод в класа **DynamicSlaveThread** за достъпване на опашката синхронизацията е „по-груба“, тъй като целият код на метода **poll()** на опашката е маркиран като критична секция. При използване на **ConcurrentLinkedQueue** класа само отделни части на метода са критични секции, по-точно само промените върху референциите към възлите на опашката, тъй като те са от тип **AtomicReference.** Така синхронизацията е върху по-малък брой инструкции и по-късата критична секция означава по-голяма пропускливост и това води до по-бързо изпълнение дори и при голям брой нишки[5], [6].

**Main** класът след изчакване на обработващите записва **BufferedImage** изображението във файл с формат .png.

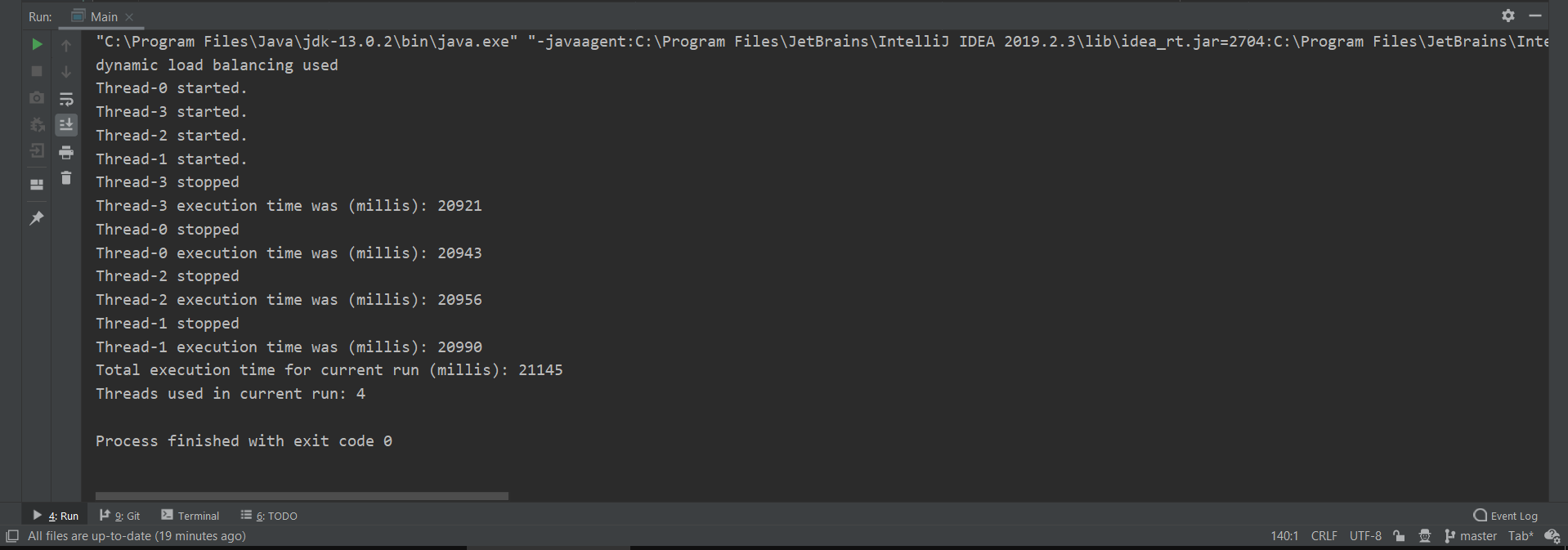
* 1. **диаграма на класовете**



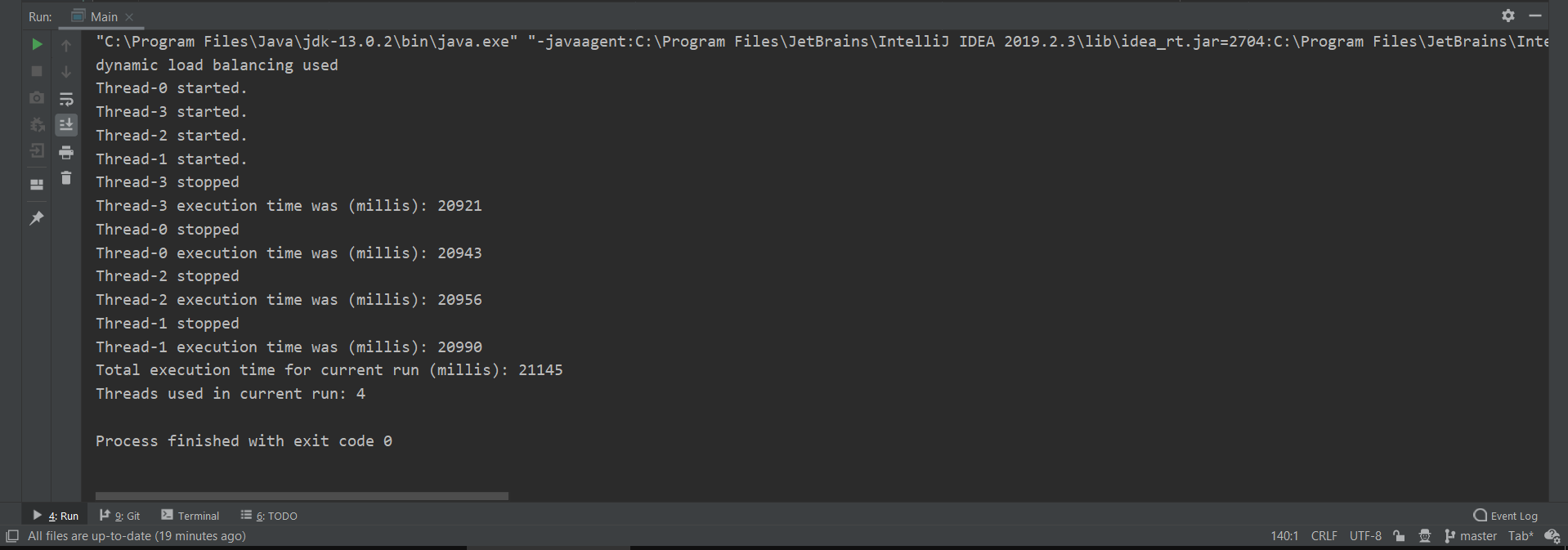
Фигура 3. Клас диаграма

* 1. **интерфейсни екрани**

Екраните са от Run прозореца при изпълнение на програмата в интегрираната среда за разработка Intellij 2020.

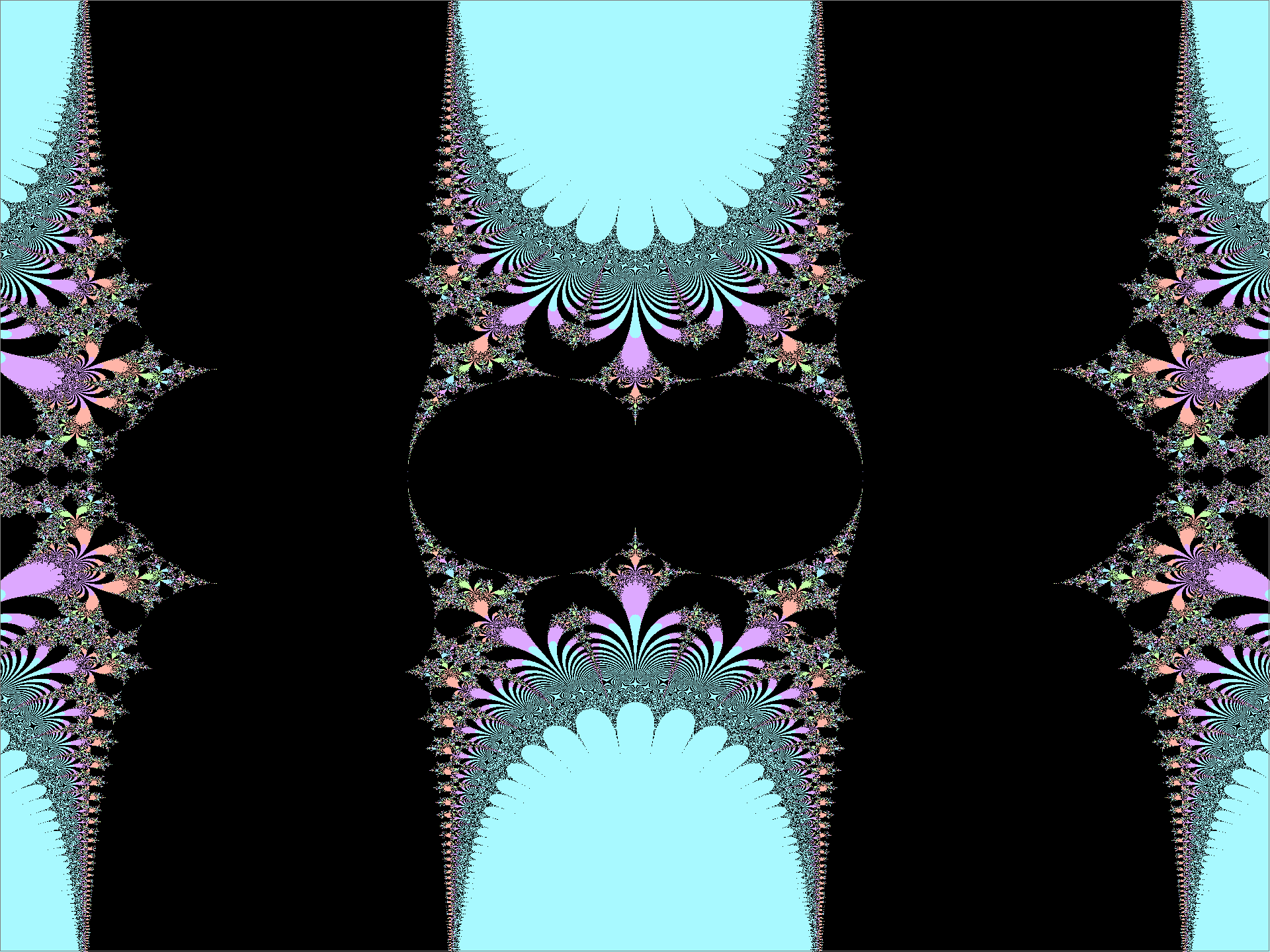


Фигура 4. Изпълнение на програмата с опцията за показване на помощ



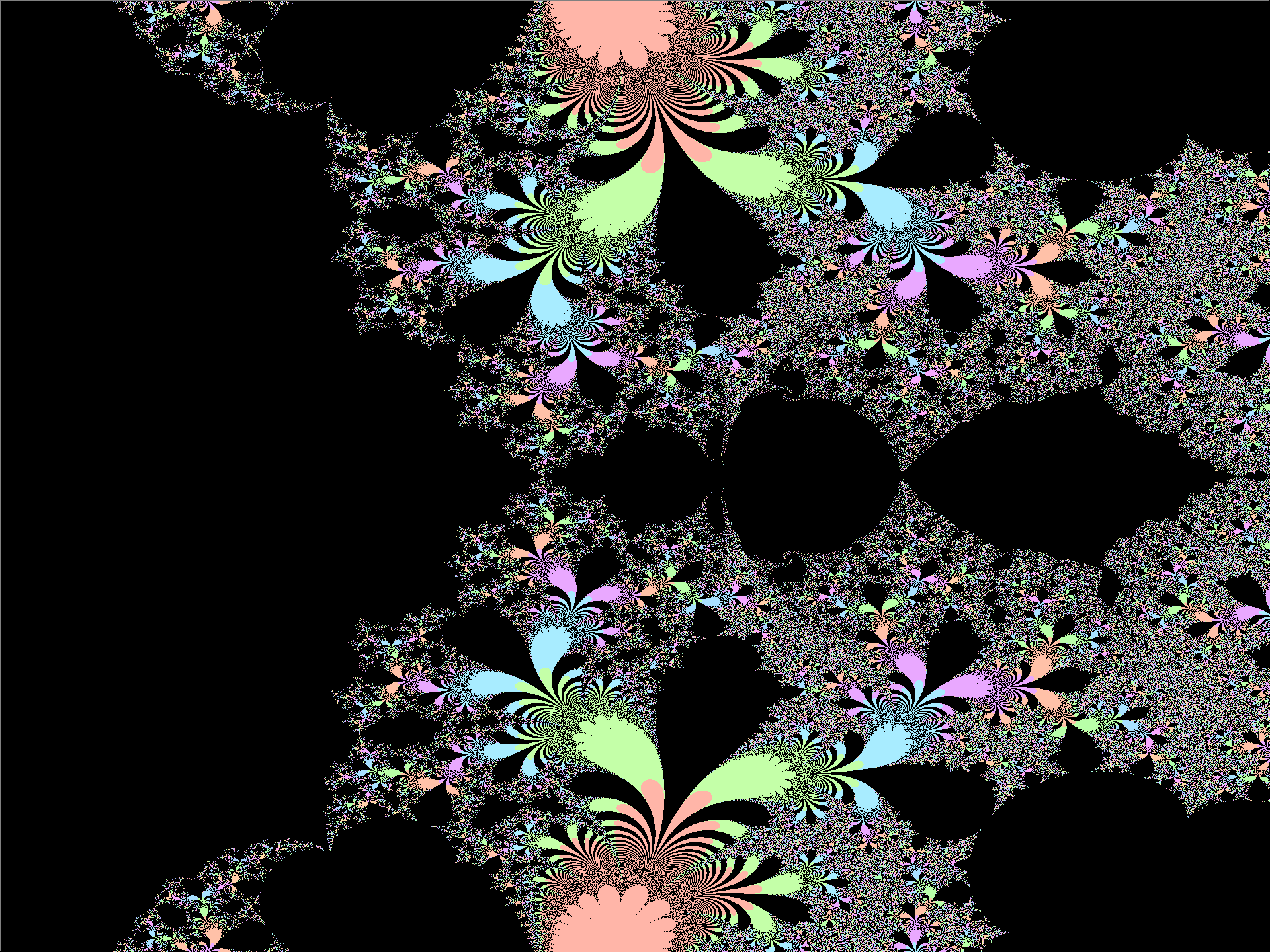
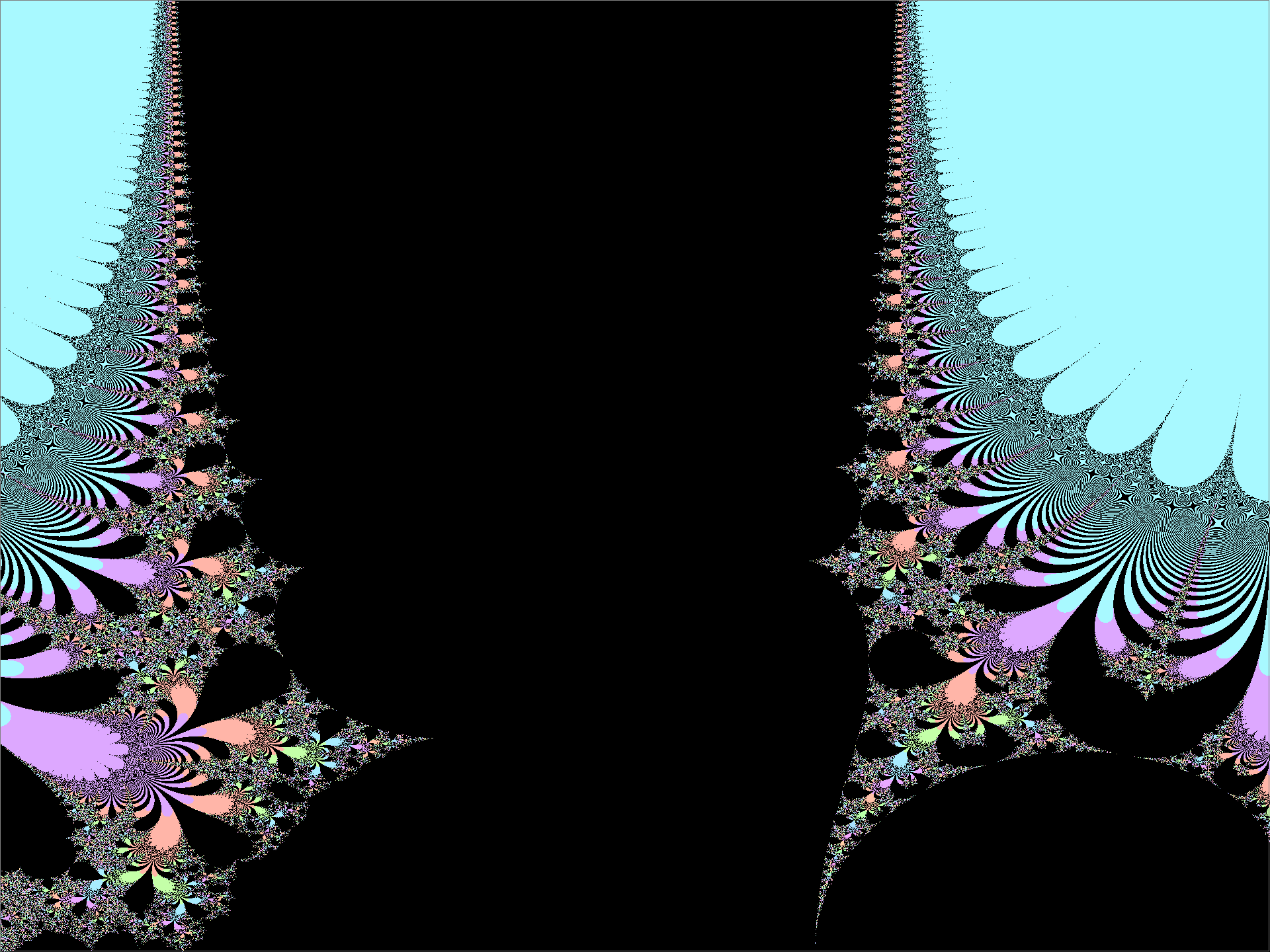
Фигура 5. Изпълнение на програмата с опцията за показване на помощ

* 1. **фрактални изображения**

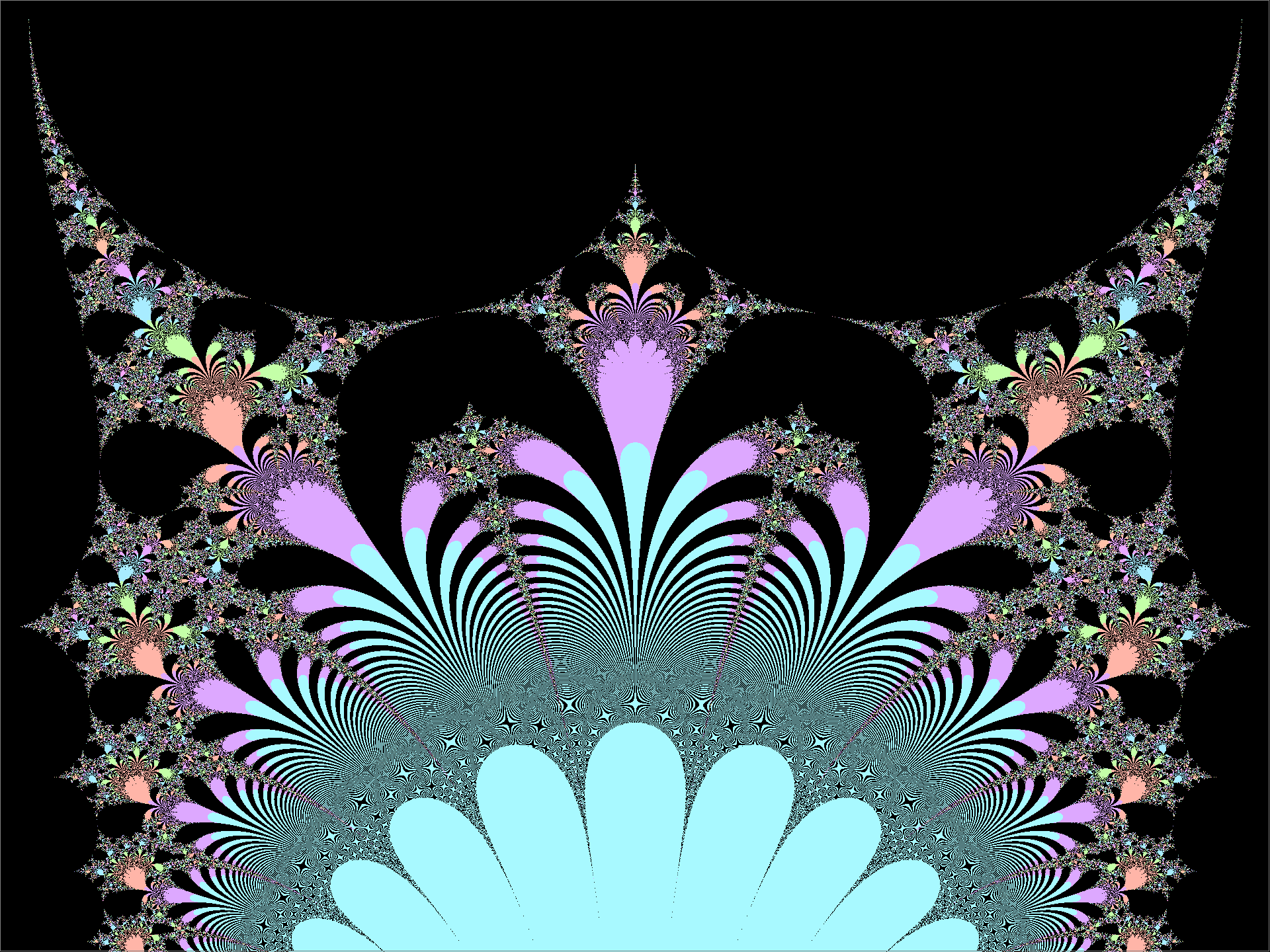


Фигура 6. фрактал, област -2.0:2.0:-2.0:2.0, размер 1920x1440

Фигура 7. област -2.0:2.0:-2.0:2.0, размер 1920x1440



Фигура 8. област -1.5:2.0:-0.25:0.25, размер 1920x1440



Фигура 9. област -0.75:0.75:0.0:1.25, размер 1920x1440

* 1. **изисквания към платформата**

Програмата може да се изпълнява на операционни платформи, които имат инсталирана JDK(Java Development Kit) от версия 8 нагоре и поддръжка на паралелна обработка. За визуализиране на изображенията, които са резултата от програмата е нужен екран с минимална резолюция 1280x960.

1. **Тестов план и тестови резултати**
   1. **характеристика на тестовата платформа**

Решението беше тествано на машина, която е достъпна по SSH на адреса t5600.rmi.yaht.net. Нейните характеристики са

* 2 процесора Intel Xeon E5 2660, 8 ядра, 16 нишки, 2.2 GHz, 32 kB L1 кеш
* 62,7 GB RAM
* Операционна система : Cent OS 7
* версия на JDK : OpenJDK 1.8.0\_252
  1. **Параметри на тестовия план**

Програмата беше изпълнена с паралелизъм, **p** = 1, 2, 4 …, 20, 24 нишки

Варирани бяха следните параметри на програмата :

* големина на изображението(сложност на заданието) : тествани бяха размерите 640x480, 1280x960, 1920x1440, 2560x1920. Регионът в комплескната равнина беше от -2.0 до 2.0 по x и y.
* регион в комплекната равнина, върху който да се визуализира фрактала (различна балансираност на заданията) : бяха тествани 3 различни области. Размерът на изображението при тези тестове беше 1920x1440.
* максимален брой итерации на точка (сложност на заданието) : стойностите на броя итерации са 350, 550, 850, 1150. Размерът на изображението при тези тестове беше 1920x1440 и областта в комплескната равнина беше от -2.0 до 2.0 по x и y.

Всички горни параметри бяха варирани при статичното и при динамичното балансиране на изпълнението. Програмата беше пускана с –q опцията, която само изписва общото време на програмата, без други подробности. Също така беше премахнат кода, който записва във файл резултата. Записването във файл отнемаше между 0,2 и 2 секунди в зависимост от натоварването на машината и това внасяше големи разлики в резултатите.

В тестовия план резултатите за време са измерени в милисекунди. Направени са по 5 изпълнения за всеки параметър и е използвано най-бързото време. С него са пресметнати

* ускорението **Sp**
* ефективността **Ep** : процента от общото време за паралелна обработка, през което ресурсите на процесора се използват ефективно
* цената **Cp** : оценка за горната граница на броя операции, които биха могли да се извършат за дадено **Tp** време на паралелния алгоритъм
  1. **Тестване на програмата със статично балансиране на изпълнението**
     1. **параметър големина на изображение**
        1. **размер 640x480**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| imageSize : 640x480 | | | | | | | | | |
| ***p*** | ***Тр*(1)** | ***Тр*(2)** | ***Тр*(3)** | ***Тр*(4)** | ***Тр*(5)** | ***Тр* = min()** | ***Sр* = *T*1*/Tp*** | ***Eр* = *Sp/p*** | ***Cp = p\*Tp*** |
| **1** | 32029 | 29935 | 30328 | 29604 | 30809 | 29604 | 1 | 1 | 29604 |
| **2** | 15671 | 15476 | 15674 | 15274 | 15565 | 15274 | 1,938195627 | 0,96909781 | 30548 |
| **4** | 8202 | 8174 | 8147 | 8297 | 7985 | 7985 | 3,707451472 | 0,92686287 | 31940 |
| **6** | 5815 | 5810 | 5700 | 5670 | 5721 | 5670 | 5,221164021 | 0,870194 | 34020 |
| **8** | 4515 | 4517 | 4718 | 4528 | 4395 | 4395 | 6,735836177 | 0,84197952 | 35160 |
| **10** | 3649 | 3705 | 3724 | 3704 | 3695 | 3649 | 8,112907646 | 0,81129076 | 36490 |
| **12** | 3202 | 3302 | 3159 | 3296 | 3170 | 3159 | 9,371320038 | 0,78094334 | 37908 |
| **14** | 2797 | 2867 | 2848 | 2904 | 2943 | 2797 | 10,58419735 | 0,7560141 | 39158 |
| **16** | 2721 | 2796 | 2797 | 2686 | 2716 | 2686 | 11,02159345 | 0,68884959 | 42976 |
| **18** | 2633 | 2574 | 2649 | 2726 | 2652 | 2574 | 11,5011655 | 0,63895364 | 46332 |
| **20** | 2560 | 2553 | 2475 | 2535 | 2610 | 2475 | 11,96121212 | 0,59806061 | 49500 |
| **22** | 2388 | 2364 | 2414 | 2340 | 2447 | 2340 | 12,65128205 | 0,57505828 | 51480 |
| **24** | 2334 | 2264 | 2363 | 2358 | 2248 | 2248 | 13,16903915 | 0,54870996 | 53952 |

* + - 1. **размер 1280x960**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| imageSize : 1280x960 | | | | | | | | | |
| ***p*** | ***Тр*(1)** | ***Тр*(2)** | ***Тр*(3)** | ***Тр*(4)** | ***Тр*(5)** | ***Тр* = min()** | ***Sр* = *T*1*/Tp*** | ***Eр* = *Sp/p*** | ***Cp = p\*Tp*** |
| **1** | 118394 | 119423 | 118042 | 117476 | 118270 | 117476 | 1 | 1 | 117476 |
| **2** | 60327 | 61114 | 62219 | 60267 | 60418 | 60267 | 1,94925913 | 0,97462957 | 120534 |
| **4** | 30992 | 30994 | 30933 | 31332 | 30929 | 30929 | 3,798247599 | 0,9495619 | 123716 |
| **6** | 21201 | 21254 | 20928 | 21194 | 21075 | 20928 | 5,613340979 | 0,93555683 | 125568 |
| **8** | 16379 | 16426 | 16279 | 16374 | 16241 | 16241 | 7,233298442 | 0,90416231 | 129928 |
| **10** | 13490 | 13259 | 13330 | 13420 | 13222 | 13222 | 8,884888822 | 0,88848888 | 132220 |
| **12** | 11249 | 11297 | 11379 | 11241 | 11278 | 11241 | 10,45067165 | 0,8708893 | 134892 |
| **14** | 9944 | 9815 | 9964 | 9823 | 9838 | 9815 | 11,969027 | 0,8549305 | 137410 |
| **16** | 9484 | 9386 | 9527 | 9188 | 9231 | 9188 | 12,78580758 | 0,79911297 | 147008 |
| **18** | 9327 | 9080 | 9026 | 9027 | 9493 | 9026 | 13,01528916 | 0,72307162 | 162468 |
| **20** | 8975 | 8636 | 8875 | 8928 | 8776 | 8636 | 13,60305697 | 0,68015285 | 172720 |
| **22** | 8420 | 8527 | 8468 | 8649 | 8403 | 8403 | 13,98024515 | 0,63546569 | 184866 |
| **24** | 8093 | 8175 | 8099 | 8088 | 8143 | 8088 | 14,52472799 | 0,605197 | 194112 |

* + - 1. **размер 1920x1440**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| imageSize : 1920x1440 | | | | | | | | | |
| ***p*** | ***Тр*(1)** | ***Тр*(2)** | ***Тр*(3)** | ***Тр*(4)** | ***Тр*(5)** | ***Тр* = min()** | ***Sр* = *T*1*/Tp*** | ***Eр* = *Sp/p*** | ***Cp = p\*Tp*** |
| **1** | 270503 | 267837 | 266953 | 272230 | 266672 | 266672 | 1 | 1 | 266672 |
| **2** | 136167 | 135468 | 136895 | 136299 | 137804 | 135468 | 1,968523932 | 0,98426197 | 270936 |
| **4** | 68958 | 68925 | 68836 | 68977 | 68872 | 68836 | 3,874019408 | 0,96850485 | 275344 |
| **6** | 46524 | 46640 | 46722 | 46892 | 46938 | 46524 | 5,731923308 | 0,95532055 | 279144 |
| **8** | 35856 | 36056 | 35675 | 36099 | 36122 | 35675 | 7,475038542 | 0,93437982 | 285400 |
| **10** | 29512 | 29046 | 29228 | 29419 | 29310 | 29046 | 9,181023205 | 0,91810232 | 290460 |
| **12** | 24637 | 24727 | 24723 | 24852 | 24654 | 24637 | 10,82404514 | 0,90200376 | 295644 |
| **14** | 21451 | 21690 | 21357 | 21605 | 21400 | 21357 | 12,4863979 | 0,89188556 | 298998 |
| **16** | 20340 | 20338 | 20134 | 20374 | 21226 | 20134 | 13,24485944 | 0,82780372 | 322144 |
| **18** | 19547 | 19372 | 19289 | 19234 | 19707 | 19234 | 13,86461474 | 0,77025637 | 346212 |
| **20** | 18675 | 18562 | 18951 | 18800 | 18671 | 18562 | 14,36655533 | 0,71832777 | 371240 |
| **22** | 17990 | 18222 | 18042 | 18049 | 18074 | 17990 | 14,8233463 | 0,67378847 | 395780 |
| **24** | 17378 | 17353 | 17632 | 17595 | 17679 | 17353 | 15,36748689 | 0,64031195 | 416472 |

* + - 1. **размер 2560x1920**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| imageSize : 2560x1920 | | | | | | | | | |
| ***p*** | ***Тр*(1)** | ***Тр*(2)** | ***Тр*(3)** | ***Тр*(4)** | ***Тр*(5)** | ***Тр* = min()** | ***Sр* = *T*1*/Tp*** | ***Eр* = *Sp/p*** | ***Cp = p\*Tp*** |
| **1** | 480502 | 476758 | 478973 | 480365 | 477306 | 476758 | 1 | 1 | 476758 |
| **2** | 241646 | 242215 | 243966 | 243332 | 244710 | 241646 | 1,972960446 | 0,98648022 | 483292 |
| **4** | 124424 | 124146 | 124311 | 123556 | 123869 | 123556 | 3,858638998 | 0,96465975 | 494224 |
| **6** | 83792 | 83800 | 83748 | 83745 | 83784 | 83745 | 5,692972715 | 0,94882879 | 502470 |
| **8** | 68479 | 64162 | 63910 | 64012 | 64447 | 63910 | 7,459834142 | 0,93247927 | 511280 |
| **10** | 52089 | 52013 | 55860 | 53644 | 52147 | 52013 | 9,166131544 | 0,91661315 | 520130 |
| **12** | 47385 | 49961 | 47261 | 44210 | 43970 | 43970 | 10,84280191 | 0,90356683 | 527640 |
| **14** | 38164 | 38092 | 38064 | 44189 | 47930 | 38064 | 12,52516814 | 0,89465487 | 532896 |
| **16** | 38612 | 36437 | 36853 | 35493 | 36157 | 35493 | 13,43245147 | 0,83952822 | 567888 |
| **18** | 35277 | 34896 | 34543 | 35363 | 34272 | 34272 | 13,91100607 | 0,77283367 | 616896 |
| **20** | 33077 | 33378 | 33188 | 33516 | 32752 | 32752 | 14,55660723 | 0,72783036 | 655040 |
| **22** | 31980 | 31937 | 32118 | 31836 | 31727 | 31727 | 15,02688562 | 0,68304026 | 697994 |
| **24** | 30963 | 30770 | 30739 | 30677 | 30903 | 30677 | 15,54121981 | 0,64755083 | 736248 |

* + - 1. **диаграма на ускорението Sp**
      2. **диаграма на ефективността Ep**
      3. **диаграма на цената Cp**
    1. **параметър регион в комплекната равнина**
       1. **регион 1 : -2.0:0.0:-2.0:0.0**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| rectangle 1 : -2.0:0.0:-2.0:0.0 | | | | | | | | | |
| ***p*** | ***Тр*(1)** | ***Тр*(2)** | ***Тр*(3)** | ***Тр*(4)** | ***Тр*(5)** | ***Тр* = min()** | ***Sр* = *T*1*/Tp*** | ***Eр* = *Sp/p*** | ***Cp = p\*Tp*** |
| **1** | 270142 | 269523 | 272484 | 273674 | 268663 | 268663 | 1 | 1 | 268663 |
| **2** | 137037 | 139043 | 137456 | 138048 | 137043 | 137037 | 1,960514314 | 0,980257157 | 274074 |
| **4** | 70812 | 70794 | 71019 | 69726 | 70374 | 69726 | 3,85312509 | 0,963281272 | 278904 |
| **6** | 47749 | 48149 | 47991 | 47708 | 47802 | 47708 | 5,631403538 | 0,938567256 | 286248 |
| **8** | 36665 | 36808 | 36494 | 36535 | 36571 | 36494 | 7,361840303 | 0,920230038 | 291952 |
| **10** | 30090 | 29816 | 29734 | 30653 | 29633 | 29633 | 9,066344953 | 0,906634495 | 296330 |
| **12** | 25359 | 25955 | 26018 | 25062 | 25310 | 25062 | 10,71993456 | 0,89332788 | 300744 |
| **14** | 21743 | 22170 | 22946 | 22422 | 22551 | 21743 | 12,35629858 | 0,882592756 | 304402 |
| **16** | 20979 | 22285 | 21754 | 22017 | 23164 | 20979 | 12,80628247 | 0,800392655 | 335664 |
| **18** | 22944 | 20990 | 20275 | 20920 | 21898 | 20275 | 13,25094945 | 0,736163858 | 364950 |
| **20** | 20015 | 22303 | 21640 | 21467 | 21665 | 20015 | 13,42308269 | 0,671154134 | 400300 |
| **22** | 20479 | 22058 | 21486 | 19490 | 21271 | 19490 | 13,7846588 | 0,6265754 | 428780 |
| **24** | 21184 | 19521 | 17777 | 20936 | 21516 | 17777 | 15,11295494 | 0,629706456 | 426648 |

* + - 1. **регион 2 : 1.5:2.0:-0.25:0.25**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| rectangle 2 : 1.5:2.0:-0.25:0.25 | | | | | | | | | |
| ***p*** | ***Тр*(1)** | ***Тр*(2)** | ***Тр*(3)** | ***Тр*(4)** | ***Тр*(5)** | ***Тр* = min()** | ***Sр* = *T*1*/Tp*** | ***Eр* = *Sp/p*** | ***Cp = p\*Tp*** |
| **1** | 291528 | 293005 | 289279 | 290816 | 294826 | 289279 | 1 | 1 | 289279 |
| **2** | 149318 | 148756 | 148331 | 148927 | 148030 | 148030 | 1,954191718 | 0,977095859 | 296060 |
| **4** | 76202 | 74996 | 75257 | 75864 | 75441 | 74996 | 3,857259054 | 0,964314763 | 299984 |
| **6** | 51460 | 50973 | 51152 | 51060 | 51336 | 50973 | 5,675141742 | 0,945856957 | 305838 |
| **8** | 39604 | 39655 | 39085 | 39251 | 38908 | 38908 | 7,434949111 | 0,929368639 | 311264 |
| **10** | 31902 | 31963 | 32503 | 32463 | 32118 | 31902 | 9,0677387 | 0,90677387 | 319020 |
| **12** | 27243 | 27131 | 27130 | 27566 | 27130 | 27130 | 10,66269812 | 0,888558177 | 325560 |
| **14** | 23754 | 23581 | 23528 | 23664 | 24384 | 23528 | 12,29509521 | 0,878221086 | 329392 |
| **16** | 22883 | 21968 | 22710 | 21968 | 22779 | 21968 | 13,1681992 | 0,82301245 | 351488 |
| **18** | 21350 | 21544 | 21333 | 21191 | 21227 | 21191 | 13,6510311 | 0,758390617 | 381438 |
| **20** | 21068 | 20473 | 20499 | 20613 | 20665 | 20473 | 14,12978069 | 0,706489034 | 409460 |
| **22** | 19643 | 19609 | 19615 | 19899 | 19713 | 19609 | 14,75235861 | 0,670561755 | 431398 |
| **24** | 18951 | 19008 | 18993 | 19076 | 19048 | 18951 | 15,26457707 | 0,636024044 | 454824 |

* + - 1. **регион 3 : -0.75:0.75:0.0:1.25**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| rectangle 3 : -0.75:0.75:0.0:1.25 | | | | | | | | | |
| ***p*** | ***Тр*(1)** | ***Тр*(2)** | ***Тр*(3)** | ***Тр*(4)** | ***Тр*(5)** | ***Тр* = min()** | ***Sр* = *T*1*/Tp*** | ***Eр* = *Sp/p*** | ***Cp = p\*Tp*** |
| **1** | 250456 | 252913 | 253562 | 252998 | 253434 | 250456 | 1 | 1 | 250456 |
| **2** | 129493 | 130076 | 129139 | 126982 | 129299 | 126982 | 1,972374037 | 0,986187019 | 253964 |
| **4** | 66198 | 66203 | 66848 | 65994 | 67381 | 65994 | 3,795132891 | 0,948783223 | 263976 |
| **6** | 45027 | 44640 | 45177 | 45460 | 45519 | 44640 | 5,610573477 | 0,935095579 | 267840 |
| **8** | 34417 | 34390 | 34855 | 34125 | 34687 | 34125 | 7,339369963 | 0,917421245 | 273000 |
| **10** | 28234 | 28425 | 27951 | 28197 | 28292 | 27951 | 8,960538085 | 0,896053808 | 279510 |
| **12** | 23844 | 23830 | 23905 | 24015 | 24252 | 23830 | 10,5101133 | 0,875842775 | 285960 |
| **14** | 20859 | 20887 | 21192 | 20891 | 20808 | 20808 | 12,03652441 | 0,859751744 | 291312 |
| **16** | 19225 | 19239 | 19681 | 19282 | 20133 | 19225 | 13,02762029 | 0,814226268 | 307600 |
| **18** | 19090 | 18541 | 18905 | 18928 | 19038 | 18541 | 13,50822501 | 0,750456945 | 333738 |
| **20** | 18150 | 17981 | 17803 | 18343 | 18071 | 17803 | 14,06819075 | 0,703409538 | 356060 |
| **22** | 17529 | 17634 | 17393 | 17730 | 17409 | 17393 | 14,39981602 | 0,654537092 | 382646 |
| **24** | 16589 | 18967 | 18694 | 16759 | 17862 | 16589 | 15,09771535 | 0,629071473 | 398136 |

* + - 1. **диаграма на ускорението Sp**
      2. **диаграма на ефективността Ep**
      3. **диаграма на цената Cp**
    1. **параметър максимален брой итерации в точка**
       1. **итерации 350**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| iterations : 350 | | | | | | | | | |
| ***p*** | ***Тр*(1)** | ***Тр*(2)** | ***Тр*(3)** | ***Тр*(4)** | ***Тр*(5)** | ***Тр* = min()** | ***Sр* = *T*1*/Tp*** | ***Eр* = *Sp/p*** | ***Cp = p\*Tp*** |
| **1** | 186998 | 198551 | 185342 | 186588 | 186027 | 185342 | 1 | 1 | 185342 |
| **2** | 94673 | 95952 | 94616 | 94711 | 95020 | 94616 | 1,958886446 | 0,979443223 | 189232 |
| **4** | 48375 | 48747 | 48601 | 48978 | 48443 | 48375 | 3,831359173 | 0,957839793 | 193500 |
| **6** | 32913 | 32981 | 32809 | 32844 | 32822 | 32809 | 5,649120668 | 0,941520111 | 196854 |
| **8** | 25319 | 25247 | 25382 | 25200 | 25175 | 25175 | 7,362144985 | 0,920268123 | 201400 |
| **10** | 20530 | 20804 | 20547 | 20548 | 20498 | 20498 | 9,041955313 | 0,904195531 | 204980 |
| **12** | 17474 | 17450 | 17498 | 17580 | 17451 | 17450 | 10,62131805 | 0,885109838 | 209400 |
| **14** | 15235 | 15178 | 15306 | 15243 | 15466 | 15178 | 12,21122678 | 0,872230484 | 212492 |
| **16** | 14206 | 14184 | 14377 | 14259 | 14229 | 14184 | 13,06697688 | 0,816686055 | 226944 |
| **18** | 13984 | 14309 | 13995 | 13705 | 14055 | 13705 | 13,52367749 | 0,751315416 | 246690 |
| **20** | 13607 | 13535 | 13307 | 13767 | 13315 | 13307 | 13,92815811 | 0,696407906 | 266140 |
| **22** | 12939 | 12811 | 12688 | 12912 | 12813 | 12688 | 14,60766078 | 0,663984581 | 279136 |
| **24** | 12319 | 12534 | 16225 | 15473 | 12357 | 12319 | 15,04521471 | 0,626883946 | 295656 |

* + - 1. **итерации 550**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| iterations : 550 | | | | | | | | | |
| ***p*** | ***Тр*(1)** | ***Тр*(2)** | ***Тр*(3)** | ***Тр*(4)** | ***Тр*(5)** | ***Тр* = min()** | ***Sр* = *T*1*/Tp*** | ***Eр* = *Sp/p*** | ***Cp = p\*Tp*** |
| **1** | 292263 | 292943 | 294102 | 310037 | 296266 | 292263 | 1 | 1 | 292263 |
| **2** | 152349 | 150565 | 151983 | 148862 | 148979 | 148862 | 1,963315017 | 0,981657508 | 297724 |
| **4** | 75853 | 76315 | 77385 | 75996 | 75392 | 75392 | 3,876578417 | 0,969144604 | 301568 |
| **6** | 51325 | 51656 | 51363 | 51911 | 51515 | 51325 | 5,694359474 | 0,949059912 | 307950 |
| **8** | 39212 | 39248 | 39987 | 39274 | 39257 | 39212 | 7,45340712 | 0,93167589 | 313696 |
| **10** | 31841 | 31866 | 31952 | 32025 | 32014 | 31841 | 9,178826042 | 0,917882604 | 318410 |
| **12** | 27089 | 27097 | 27134 | 27005 | 27145 | 27005 | 10,82255138 | 0,901879282 | 324060 |
| **14** | 23492 | 23573 | 23755 | 23589 | 23474 | 23474 | 12,45049842 | 0,889321316 | 328636 |
| **16** | 22112 | 22268 | 22691 | 22265 | 22658 | 22112 | 13,21739327 | 0,826087079 | 353792 |
| **18** | 21773 | 21181 | 21629 | 21535 | 21247 | 21181 | 13,79835702 | 0,76657539 | 381258 |
| **20** | 20350 | 20602 | 20654 | 21331 | 21146 | 20350 | 14,36181818 | 0,718090909 | 407000 |
| **22** | 19759 | 19917 | 19828 | 19884 | 19887 | 19759 | 14,7913862 | 0,672335737 | 434698 |
| **24** | 19119 | 19215 | 19193 | 19181 | 18793 | 18793 | 15,55169478 | 0,647987282 | 451032 |

* + - 1. **итерации 850**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| iterations : 850 | | | | | | | | | |
| ***p*** | ***Тр*(1)** | ***Тр*(2)** | ***Тр*(3)** | ***Тр*(4)** | ***Тр*(5)** | ***Тр* = min()** | ***Sр* = *T*1*/Tp*** | ***Eр* = *Sp/p*** | ***Cp = p\*Tp*** |
| **1** | 452568 | 460258 | 464503 | 455185 | 461355 | 452568 | 1 | 1 | 452568 |
| **2** | 230574 | 229857 | 239497 | 239423 | 235621 | 229857 | 1,968911106 | 0,984455553 | 459714 |
| **4** | 125203 | 117862 | 119168 | 129826 | 119036 | 117862 | 3,839812662 | 0,959953166 | 471448 |
| **6** | 82389 | 81266 | 84232 | 89798 | 80019 | 80019 | 5,655756758 | 0,942626126 | 480114 |
| **8** | 61269 | 62968 | 62472 | 65953 | 75693 | 61269 | 7,386573961 | 0,923321745 | 490152 |
| **10** | 50298 | 49450 | 49750 | 50330 | 50636 | 49450 | 9,152032356 | 0,915203236 | 494500 |
| **12** | 47884 | 49905 | 53656 | 58158 | 41920 | 41920 | 10,79599237 | 0,899666031 | 503040 |
| **14** | 37236 | 36733 | 38561 | 38488 | 40553 | 36733 | 12,32047478 | 0,880033913 | 514262 |
| **16** | 35480 | 36360 | 35872 | 36055 | 36623 | 35480 | 12,75558061 | 0,797223788 | 567680 |
| **18** | 44726 | 39151 | 35199 | 33929 | 33597 | 33597 | 13,47048844 | 0,748360469 | 604746 |
| **20** | 32152 | 34047 | 32287 | 32328 | 32340 | 32152 | 14,07588952 | 0,703794476 | 643040 |
| **22** | 30684 | 32553 | 30881 | 31779 | 30890 | 30684 | 14,7493156 | 0,670423437 | 675048 |
| **24** | 28824 | 30503 | 29731 | 29492 | 30358 | 28824 | 15,70108243 | 0,654211768 | 691776 |

* + - 1. **итерации 1150**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| iterations : 1150 | | | | | | | | | |
| ***p*** | ***Тр*(1)** | ***Тр*(2)** | ***Тр*(3)** | ***Тр*(4)** | ***Тр*(5)** | ***Тр* = min()** | ***Sр* = *T*1*/Tp*** | ***Eр* = *Sp/p*** | ***Cp = p\*Tp*** |
| **1** | 637922 | 688684 | 655121 | 668565 | 645231 | 637922 | 1 | 1 | 637922 |
| **2** | 328146 | 329253 | 322819 | 343612 | 332127 | 322819 | 1,976098061 | 0,988049031 | 645638 |
| **4** | 166848 | 167218 | 168379 | 173552 | 161464 | 161464 | 3,950862112 | 0,987715528 | 645856 |
| **6** | 108971 | 110446 | 110237 | 111535 | 110119 | 108971 | 5,854052913 | 0,975675486 | 653826 |
| **8** | 81127 | 81353 | 81311 | 81552 | 81317 | 81127 | 7,863251445 | 0,982906431 | 649016 |
| **10** | 66608 | 65972 | 66134 | 66363 | 66669 | 65972 | 9,669587098 | 0,96695871 | 659720 |
| **12** | 56141 | 56238 | 56956 | 56291 | 59779 | 56141 | 11,36285424 | 0,94690452 | 673692 |
| **14** | 48590 | 48871 | 49824 | 48961 | 48662 | 48590 | 13,12866845 | 0,937762032 | 680260 |
| **16** | 45627 | 45218 | 46658 | 45690 | 45620 | 45218 | 14,10770047 | 0,88173128 | 723488 |
| **18** | 44462 | 44285 | 43646 | 44928 | 44086 | 43646 | 14,61581817 | 0,811989899 | 785628 |
| **20** | 42078 | 42711 | 42394 | 42018 | 42827 | 42018 | 15,18211243 | 0,759105621 | 840360 |
| **22** | 40687 | 41588 | 40725 | 40779 | 41146 | 40687 | 15,67876717 | 0,712671235 | 895114 |
| **24** | 39239 | 39310 | 39474 | 38921 | 39147 | 38921 | 16,39017497 | 0,682923957 | 934104 |

* + - 1. **диаграма на ускорението Sp**
      2. **диаграма на ефективността Ep**
      3. **диаграма на цената Cp**
  1. **Тестване на програмата с динамично балансиране на изпълнението**
     1. **параметър големина на изображението**
        1. **размер 640x480**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| imageSize : 640x480 | | | | | | | | | | |
| **#** | ***p*** | ***Тр*(1)** | ***Тр*(2)** | ***Тр*(3)** | ***Тр*(4)** | ***Тр*(5)** | ***Тр* = min()** | ***Sр* = *T*1*/Tp*** | ***Eр* = *Sp/p*** | ***Cp = p\*Tp*** |
| 1 | **1** | 32302 | 31760 | 30196 | 29664 | 29569 | 29569 | 1 | 1 | 29569 |
| 2 | **2** | 15578 | 15911 | 15481 | 15764 | 15545 | 15481 | 1,910018733 | 0,95500937 | 30962 |
| 3 | **4** | 8137 | 8299 | 8042 | 8184 | 8251 | 8042 | 3,676821686 | 0,91920542 | 32168 |
| 4 | **6** | 5719 | 5703 | 5854 | 5637 | 5641 | 5637 | 5,245520667 | 0,87425344 | 33822 |
| 5 | **8** | 4485 | 4468 | 4476 | 4515 | 4427 | 4427 | 6,679241021 | 0,83490513 | 35416 |
| 6 | **10** | 3744 | 3667 | 3720 | 3701 | 3715 | 3667 | 8,063539678 | 0,80635397 | 36670 |
| 7 | **12** | 3201 | 3211 | 3115 | 3237 | 3172 | 3115 | 9,492455859 | 0,79103799 | 37380 |
| 8 | **14** | 2865 | 2847 | 2810 | 2832 | 2822 | 2810 | 10,5227758 | 0,75162684 | 39340 |
| 9 | **16** | 2660 | 2654 | 2676 | 2715 | 2663 | 2654 | 11,14129616 | 0,69633101 | 42464 |
| 10 | **18** | 2563 | 2578 | 2577 | 2508 | 2568 | 2508 | 11,78987241 | 0,65499291 | 45144 |
| 11 | **20** | 2456 | 2420 | 2434 | 2395 | 2413 | 2395 | 12,34613779 | 0,61730689 | 47900 |
| 12 | **22** | 2306 | 2321 | 2300 | 2356 | 2327 | 2300 | 12,85608696 | 0,58436759 | 50600 |
| 13 | **24** | 2323 | 2299 | 2287 | 2296 | 2225 | 2225 | 13,2894382 | 0,55372659 | 53400 |

* + - 1. **размер 1280x960**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| imageSize : 1280x960 | | | | | | | | | | |
| **#** | ***p*** | ***Тр*(1)** | ***Тр*(2)** | ***Тр*(3)** | ***Тр*(4)** | ***Тр*(5)** | ***Тр* = min()** | ***Sр* = *T*1*/Tp*** | ***Eр* = *Sp/p*** | ***Cp = p\*Tp*** |
| 1 | **1** | 119495 | 120895 | 121182 | 119109 | 118987 | 118987 | 1 | 1 | 118987 |
| 2 | **2** | 60491 | 61222 | 59312 | 61066 | 60926 | 59312 | 2,006120178 | 1,00306009 | 118624 |
| 3 | **4** | 30921 | 30709 | 30637 | 30619 | 30685 | 30619 | 3,886051145 | 0,97151279 | 122476 |
| 4 | **6** | 21017 | 21102 | 20939 | 21006 | 21072 | 20939 | 5,682554086 | 0,94709235 | 125634 |
| 5 | **8** | 16196 | 16355 | 16011 | 16303 | 16252 | 16011 | 7,43157829 | 0,92894729 | 128088 |
| 6 | **10** | 13212 | 13139 | 13382 | 13268 | 13228 | 13139 | 9,05601644 | 0,90560164 | 131390 |
| 7 | **12** | 11307 | 11218 | 11211 | 11272 | 11319 | 11211 | 10,6134154 | 0,88445128 | 134532 |
| 8 | **14** | 9866 | 9816 | 9839 | 9896 | 9894 | 9816 | 12,12174002 | 0,86583857 | 137424 |
| 9 | **16** | 8963 | 9066 | 8912 | 8812 | 8874 | 8812 | 13,50283704 | 0,84392732 | 140992 |
| 10 | **18** | 8681 | 8621 | 8608 | 8739 | 8643 | 8608 | 13,82283922 | 0,76793551 | 154944 |
| 11 | **20** | 8473 | 8354 | 8330 | 8300 | 8405 | 8300 | 14,33578313 | 0,71678916 | 166000 |
| 12 | **22** | 8128 | 8074 | 8123 | 8173 | 8097 | 8074 | 14,73705722 | 0,66986624 | 177628 |
| 13 | **24** | 7829 | 7816 | 7932 | 7734 | 7823 | 7734 | 15,38492371 | 0,64103849 | 185616 |

* + - 1. **размер 1920x1440**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| imageSize : 1920x1440 | | | | | | | | | | |
| **#** | ***p*** | ***Тр*(1)** | ***Тр*(2)** | ***Тр*(3)** | ***Тр*(4)** | ***Тр*(5)** | ***Тр* = min()** | ***Sр* = *T*1*/Tp*** | ***Eр* = *Sp/p*** | ***Cp = p\*Tp*** |
| 1 | **1** | 271522 | 268995 | 270432 | 272053 | 273960 | 268995 | 1 | 1 | 268995 |
| 2 | **2** | 138942 | 138919 | 138670 | 135316 | 137046 | 135316 | 1,987902391 | 0,9939512 | 270632 |
| 3 | **4** | 68893 | 68693 | 69103 | 69631 | 68624 | 68624 | 3,91983854 | 0,97995964 | 274496 |
| 4 | **6** | 46638 | 47105 | 47067 | 47498 | 47397 | 46638 | 5,7677216 | 0,96128693 | 279828 |
| 5 | **8** | 36131 | 36158 | 35914 | 36254 | 36370 | 35914 | 7,489976054 | 0,93624701 | 287312 |
| 6 | **10** | 29262 | 29056 | 29022 | 29023 | 29017 | 29017 | 9,270255368 | 0,92702554 | 290170 |
| 7 | **12** | 24583 | 24732 | 24608 | 24652 | 24857 | 24583 | 10,94231786 | 0,91185982 | 294996 |
| 8 | **14** | 21321 | 21282 | 21393 | 21338 | 21375 | 21282 | 12,63955455 | 0,90282533 | 297948 |
| 9 | **16** | 19382 | 19044 | 19086 | 19191 | 19254 | 19044 | 14,12492124 | 0,88280758 | 304704 |
| 10 | **18** | 18987 | 19005 | 18921 | 18824 | 18991 | 18824 | 14,29000212 | 0,79388901 | 338832 |
| 11 | **20** | 18293 | 18223 | 18375 | 18295 | 18336 | 18223 | 14,76129068 | 0,73806453 | 364460 |
| 12 | **22** | 17570 | 17647 | 17726 | 17804 | 17696 | 17570 | 15,30990324 | 0,69590469 | 386540 |
| 13 | **24** | 17137 | 17256 | 17014 | 17177 | 16967 | 16967 | 15,85401073 | 0,66058378 | 407208 |

* + - 1. **размер 2560x1920**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| imageSize : 2560x1920 | | | | | | | | | | |
| **#** | ***p*** | ***Тр*(1)** | ***Тр*(2)** | ***Тр*(3)** | ***Тр*(4)** | ***Тр*(5)** | ***Тр* = min()** | ***Sр* = *T*1*/Tp*** | ***Eр* = *Sp/p*** | ***Cp = p\*Tp*** |
| 1 | **1** | 480287 | 479894 | 476918 | 477579 | 476960 | 476918 | 1 | 1 | 476918 |
| 2 | **2** | 242684 | 240366 | 240982 | 242092 | 240832 | 240366 | 1,984132531 | 0,99206627 | 480732 |
| 3 | **4** | 121531 | 120625 | 121664 | 120984 | 121737 | 120625 | 3,953724352 | 0,98843109 | 482500 |
| 4 | **6** | 82702 | 82490 | 83962 | 82986 | 82661 | 82490 | 5,781525033 | 0,96358751 | 494940 |
| 5 | **8** | 63152 | 63144 | 63154 | 63146 | 63116 | 63116 | 7,556213955 | 0,94452674 | 504928 |
| 6 | **10** | 51413 | 51487 | 51406 | 51251 | 51306 | 51251 | 9,305535502 | 0,93055355 | 512510 |
| 7 | **12** | 43555 | 43351 | 43475 | 43162 | 43371 | 43162 | 11,04948798 | 0,92079066 | 517944 |
| 8 | **14** | 37949 | 38191 | 37793 | 37725 | 37989 | 37725 | 12,64196156 | 0,90299725 | 528150 |
| 9 | **16** | 33704 | 33864 | 33730 | 33767 | 33975 | 33704 | 14,15018989 | 0,88438687 | 539264 |
| 10 | **18** | 32821 | 33146 | 33037 | 32985 | 33026 | 32821 | 14,53087962 | 0,80727109 | 590778 |
| 11 | **20** | 32080 | 31935 | 31993 | 32020 | 31943 | 31935 | 14,93402223 | 0,74670111 | 638700 |
| 12 | **22** | 30919 | 30879 | 30828 | 30896 | 30868 | 30828 | 15,47028675 | 0,70319485 | 678216 |
| 13 | **24** | 29849 | 30121 | 29882 | 29707 | 29858 | 29707 | 16,05406133 | 0,66891922 | 712968 |

* + - 1. **диаграма на ускорението Sp**
      2. **диаграма на ефективността Еp**
      3. **диаграма на цената Cp**
    1. **параметър регион в комплекната равнина**
       1. **регион 1 : -2.0:0.0:-2.0:0.0**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| rectangle 1 : -2.0:0.0:-2.0:0.0 | | | | | | | | | | |
| **#** | ***p*** | ***Тр*(1)** | ***Тр*(2)** | ***Тр*(3)** | ***Тр*(4)** | ***Тр*(5)** | ***Тр* = min()** | ***Sр* = *T*1*/Tp*** | ***Eр* = *Sp/p*** | ***Cp = p\*Tp*** |
| 1 | **1** | 306317 | 354035 | 297704 | 293062 | 294099 | 293062 | 1 | 1 | 293062 |
| 2 | **2** | 150418 | 152541 | 150540 | 150156 | 151739 | 150156 | 1,951716881 | 0,975858441 | 300312 |
| 3 | **4** | 75646 | 75948 | 76203 | 74576 | 75065 | 74576 | 3,92970929 | 0,982427322 | 298304 |
| 4 | **6** | 49334 | 50356 | 49756 | 49611 | 50612 | 49334 | 5,940365671 | 0,990060945 | 296004 |
| 5 | **8** | 37241 | 38462 | 37504 | 37362 | 37291 | 37241 | 7,869337558 | 0,983667195 | 297928 |
| 6 | **10** | 29551 | 29540 | 29542 | 29585 | 29601 | 29540 | 9,920853081 | 0,992085308 | 295400 |
| 7 | **12** | 25082 | 25096 | 25274 | 34209 | 24796 | 24796 | 11,81892241 | 0,984910201 | 297552 |
| 8 | **14** | 21793 | 21615 | 21920 | 22024 | 21809 | 21615 | 13,55826972 | 0,968447837 | 302610 |
| 9 | **16** | 20136 | 20216 | 20474 | 20210 | 20400 | 20136 | 14,5541319 | 0,909633244 | 322176 |
| 10 | **18** | 19757 | 19762 | 19797 | 19375 | 19939 | 19375 | 15,12578065 | 0,840321147 | 348750 |
| 11 | **20** | 20068 | 19965 | 19298 | 18606 | 19341 | 18606 | 15,75094056 | 0,787547028 | 372120 |
| 12 | **22** | 18683 | 18590 | 17904 | 18472 | 18515 | 17904 | 16,368521 | 0,744023682 | 393888 |
| 13 | **24** | 18191 | 18190 | 18024 | 19125 | 18163 | 18024 | 16,25954283 | 0,677480951 | 432576 |

* + - 1. **регион 2 :**  **1.5:2.0:-0.25:0.25**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| rectangle 2 : 1.5:2.0:-0.25:0.25 | | | | | | | | | | |
| **#** | ***p*** | ***Тр*(1)** | ***Тр*(2)** | ***Тр*(3)** | ***Тр*(4)** | ***Тр*(5)** | ***Тр* = min()** | ***Sр* = *T*1*/Tp*** | ***Eр* = *Sp/p*** | ***Cp = p\*Tp*** |
| 1 | **1** | 291632 | 291742 | 291296 | 290918 | 293693 | 290918 | 1 | 1 | 290918 |
| 2 | **2** | 152282 | 150555 | 153452 | 168369 | 151519 | 150555 | 1,932303809 | 0,966151905 | 301110 |
| 3 | **4** | 75234 | 75124 | 75464 | 75608 | 75208 | 75124 | 3,872504127 | 0,968126032 | 300496 |
| 4 | **6** | 51241 | 51413 | 51033 | 51271 | 51333 | 51033 | 5,700585895 | 0,950097649 | 306198 |
| 5 | **8** | 38998 | 39122 | 39074 | 39058 | 39129 | 38998 | 7,459818452 | 0,932477307 | 311984 |
| 6 | **10** | 31901 | 31995 | 32153 | 31695 | 31919 | 31695 | 9,178671715 | 0,917867171 | 316950 |
| 7 | **12** | 27580 | 27208 | 27205 | 27291 | 27115 | 27115 | 10,72904297 | 0,894086914 | 325380 |
| 8 | **14** | 23564 | 23606 | 23470 | 23530 | 23679 | 23470 | 12,39531317 | 0,885379512 | 328580 |
| 9 | **16** | 21444 | 21798 | 21529 | 21589 | 21726 | 21444 | 13,56640552 | 0,847900345 | 343104 |
| 10 | **18** | 20818 | 20878 | 21399 | 20990 | 20924 | 20818 | 13,97434912 | 0,776352729 | 374724 |
| 11 | **20** | 20488 | 20069 | 20036 | 22387 | 28948 | 20036 | 14,51976442 | 0,725988221 | 400720 |
| 12 | **22** | 26607 | 30449 | 30739 | 25246 | 19279 | 19279 | 15,08989055 | 0,685904116 | 424138 |
| 13 | **24** | 18586 | 18633 | 18620 | 18728 | 18578 | 18578 | 15,65927441 | 0,652469767 | 445872 |

* + - 1. **регион 3 : -0.75:0.75:0.0:1.25**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| rectangle 3 : -0.75:0.75:0.0:1.25 | | | | | | | | | | |
| **#** | ***p*** | ***Тр*(1)** | ***Тр*(2)** | ***Тр*(3)** | ***Тр*(4)** | ***Тр*(5)** | ***Тр* = min()** | ***Sр* = *T*1*/Tp*** | ***Eр* = *Sp/p*** | ***Cp = p\*Tp*** |
| 1 | **1** | 253079 | 252735 | 256815 | 253625 | 254947 | 252735 | 1 | 1 | 252735 |
| 2 | **2** | 128865 | 128376 | 128817 | 127408 | 127002 | 127002 | 1,990008031 | 0,995004016 | 254004 |
| 3 | **4** | 65611 | 64794 | 64875 | 65130 | 65480 | 64794 | 3,900592647 | 0,975148162 | 259176 |
| 4 | **6** | 44044 | 44697 | 44516 | 44582 | 44313 | 44044 | 5,738239034 | 0,956373172 | 264264 |
| 5 | **8** | 34293 | 33455 | 34160 | 33864 | 34209 | 33455 | 7,554476162 | 0,94430952 | 267640 |
| 6 | **10** | 27806 | 28000 | 27822 | 27794 | 28200 | 27794 | 9,093149601 | 0,90931496 | 277940 |
| 7 | **12** | 23795 | 23797 | 23618 | 23518 | 23542 | 23518 | 10,74644953 | 0,895537461 | 282216 |
| 8 | **14** | 20685 | 20556 | 20794 | 20854 | 20486 | 20486 | 12,33696183 | 0,881211559 | 286804 |
| 9 | **16** | 19059 | 18818 | 18810 | 18998 | 19182 | 18810 | 13,43620415 | 0,839762759 | 300960 |
| 10 | **18** | 18211 | 18207 | 18391 | 18388 | 18350 | 18207 | 13,88119954 | 0,771177752 | 327726 |
| 11 | **20** | 17642 | 17746 | 17622 | 17666 | 17633 | 17622 | 14,34201566 | 0,717100783 | 352440 |
| 12 | **22** | 16950 | 17157 | 17088 | 16973 | 17018 | 16950 | 14,91061947 | 0,67775543 | 372900 |
| 13 | **24** | 16506 | 16326 | 16450 | 16493 | 16547 | 16326 | 15,48052187 | 0,645021744 | 391824 |

* + - 1. **диаграма на ускорението Sp**
      2. **диаграма на ефективността Еp**
      3. **диаграма на цената Cp**
    1. **параметър максимален брой итерации в точка**
       1. **итерации 350**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| iterations : 350 | | | | | | | | | | |
| **#** | ***p*** | ***Тр*(1)** | ***Тр*(2)** | ***Тр*(3)** | ***Тр*(4)** | ***Тр*(5)** | ***Тр* = min()** | ***Sр* = *T*1*/Tp*** | ***Eр* = *Sp/p*** | ***Cp = p\*Tp*** |
| 1 | **1** | 214913 | 241368 | 209733 | 188510 | 192277 | 188510 | 1 | 1 | 188510 |
| 2 | **2** | 95843 | 95177 | 95754 | 95781 | 96154 | 95177 | 1,98062557 | 0,990312786 | 190354 |
| 3 | **4** | 48400 | 48040 | 47914 | 48166 | 48362 | 47914 | 3,93434069 | 0,983585173 | 191656 |
| 4 | **6** | 32672 | 32748 | 32780 | 32802 | 32695 | 32672 | 5,76977228 | 0,961628714 | 196032 |
| 5 | **8** | 25183 | 25139 | 25251 | 25431 | 25534 | 25139 | 7,49870719 | 0,937338399 | 201112 |
| 6 | **10** | 20489 | 20520 | 20644 | 20600 | 20729 | 20489 | 9,20054663 | 0,920054663 | 204890 |
| 7 | **12** | 17429 | 17574 | 17722 | 17796 | 17784 | 17429 | 10,8158816 | 0,901323465 | 209148 |
| 8 | **14** | 15354 | 15161 | 15224 | 15135 | 15318 | 15135 | 12,4552362 | 0,889659729 | 211890 |
| 9 | **16** | 14051 | 14324 | 14287 | 14359 | 13819 | 13819 | 13,6413633 | 0,852585209 | 221104 |
| 10 | **18** | 13793 | 13749 | 13747 | 13839 | 13807 | 13747 | 13,7128101 | 0,761822782 | 247446 |
| 11 | **20** | 13330 | 13423 | 13513 | 13292 | 13181 | 13181 | 14,3016463 | 0,715082315 | 263620 |
| 12 | **22** | 12811 | 12772 | 12784 | 12908 | 13031 | 12772 | 14,7596304 | 0,670892293 | 280984 |
| 13 | **24** | 12519 | 12749 | 12998 | 13652 | 14439 | 12519 | 15,057912 | 0,627412999 | 300456 |

* + - 1. **итерации 550**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| iterations : 550 | | | | | | | | | | |
| **#** | ***p*** | ***Тр*(1)** | ***Тр*(2)** | ***Тр*(3)** | ***Тр*(4)** | ***Тр*(5)** | ***Тр* = min()** | ***Sр* = *T*1*/Tp*** | ***Eр* = *Sp/p*** | ***Cp = p\*Tp*** |
| 1 | **1** | 294165 | 293828 | 293235 | 293792 | 293577 | 293235 | 1 | 1 | 293235 |
| 2 | **2** | 148329 | 148960 | 148464 | 150827 | 148220 | 148220 | 1,97837674 | 0,989188369 | 296440 |
| 3 | **4** | 75613 | 75121 | 75123 | 75223 | 75532 | 75121 | 3,90350235 | 0,975875587 | 300484 |
| 4 | **6** | 51224 | 51378 | 51314 | 51660 | 51261 | 51224 | 5,7245627 | 0,954093784 | 307344 |
| 5 | **8** | 39114 | 39118 | 39103 | 39062 | 39088 | 39062 | 7,50691209 | 0,938364011 | 312496 |
| 6 | **10** | 31990 | 31790 | 31861 | 31808 | 31858 | 31790 | 9,22412708 | 0,922412708 | 317900 |
| 7 | **12** | 26944 | 27657 | 26983 | 26921 | 27010 | 26921 | 10,892426 | 0,907702166 | 323052 |
| 8 | **14** | 23429 | 23461 | 23462 | 23497 | 23416 | 23416 | 12,5228476 | 0,894489116 | 327824 |
| 9 | **16** | 21367 | 21429 | 21429 | 21687 | 22171 | 21367 | 13,7237329 | 0,857733304 | 341872 |
| 10 | **18** | 20889 | 20903 | 20854 | 20916 | 21011 | 20854 | 14,0613312 | 0,781185064 | 375372 |
| 11 | **20** | 20284 | 20584 | 20410 | 20340 | 20292 | 20284 | 14,4564682 | 0,722823408 | 405680 |
| 12 | **22** | 19422 | 19561 | 19503 | 19577 | 19702 | 19422 | 15,0980846 | 0,686276575 | 427284 |
| 13 | **24** | 18994 | 18984 | 18783 | 18731 | 18784 | 18731 | 15,6550638 | 0,652294325 | 449544 |

* + - 1. **итерации 850**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| iterations : 850 | | | | | | | | | | |
| **#** | ***p*** | ***Тр*(1)** | ***Тр*(2)** | ***Тр*(3)** | ***Тр*(4)** | ***Тр*(5)** | ***Тр* = min()** | ***Sр* = *T*1*/Tp*** | ***Eр* = *Sp/p*** | ***Cp = p\*Tp*** |
| 1 | **1** | 449598 | 451398 | 451440 | 453221 | 450485 | 449598 | 1 | 1 | 449598 |
| 2 | **2** | 228828 | 228726 | 228762 | 228279 | 228756 | 228279 | 1,96951099 | 0,984755497 | 456558 |
| 3 | **4** | 116063 | 116361 | 116263 | 116891 | 116479 | 116063 | 3,87374099 | 0,968435246 | 464252 |
| 4 | **6** | 78708 | 80563 | 78972 | 78890 | 78943 | 78708 | 5,71222747 | 0,952037912 | 472248 |
| 5 | **8** | 60213 | 60589 | 60277 | 60352 | 60134 | 60134 | 7,47660225 | 0,934575282 | 481072 |
| 6 | **10** | 49484 | 49003 | 49049 | 48845 | 48785 | 48785 | 9,21590653 | 0,921590653 | 487850 |
| 7 | **12** | 41494 | 41463 | 41397 | 41524 | 41373 | 41373 | 10,8669422 | 0,905578517 | 496476 |
| 8 | **14** | 35965 | 36120 | 36153 | 36149 | 36166 | 35965 | 12,5009871 | 0,892927648 | 503510 |
| 9 | **16** | 32922 | 33411 | 33222 | 33238 | 33101 | 32922 | 13,6564607 | 0,853528795 | 526752 |
| 10 | **18** | 32010 | 32743 | 32080 | 32157 | 32729 | 32010 | 14,0455483 | 0,780308237 | 576180 |
| 11 | **20** | 32203 | 31493 | 30965 | 31105 | 30957 | 30957 | 14,5233065 | 0,726165326 | 619140 |
| 12 | **22** | 29722 | 30303 | 29887 | 30108 | 29942 | 29722 | 15,1267748 | 0,687580672 | 653884 |
| 13 | **24** | 28827 | 28769 | 29150 | 28847 | 28850 | 28769 | 15,6278633 | 0,651160972 | 690456 |

* + - 1. **итерации 1150**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| iterations : 1150 | | | | | | | | | | |
| **#** | ***p*** | ***Тр*(1)** | ***Тр*(2)** | ***Тр*(3)** | ***Тр*(4)** | ***Тр*(5)** | ***Тр* = min()** | ***Sр* = *T*1*/Tp*** | ***Eр* = *Sp/p*** | ***Cp = p\*Tp*** |
| 1 | **1** | 609015 | 608432 | 610006 | 610611 | 609518 | 608432 | 1 | 1 | 608432 |
| 2 | **2** | 308485 | 309761 | 308567 | 319427 | 309230 | 308485 | 1,9723228 | 0,986161402 | 616970 |
| 3 | **4** | 158997 | 156728 | 156658 | 156889 | 156548 | 156548 | 3,88655237 | 0,971638092 | 626192 |
| 4 | **6** | 106745 | 106451 | 108319 | 106474 | 106657 | 106451 | 5,71560624 | 0,95260104 | 638706 |
| 5 | **8** | 81303 | 81429 | 81084 | 81321 | 81122 | 81084 | 7,50372453 | 0,937965567 | 648672 |
| 6 | **10** | 66216 | 66233 | 67193 | 66216 | 66098 | 66098 | 9,20499864 | 0,920499864 | 660980 |
| 7 | **12** | 56017 | 55720 | 56041 | 56126 | 57959 | 55720 | 10,9194544 | 0,909954535 | 668640 |
| 8 | **14** | 48723 | 48712 | 49227 | 48489 | 48559 | 48489 | 12,5478356 | 0,896273971 | 678846 |
| 9 | **16** | 44918 | 45323 | 45171 | 44783 | 44723 | 44723 | 13,6044541 | 0,85027838 | 715568 |
| 10 | **18** | 43353 | 43473 | 43763 | 43216 | 43305 | 43216 | 14,0788597 | 0,782158871 | 777888 |
| 11 | **20** | 42117 | 42086 | 42028 | 41962 | 41782 | 41782 | 14,5620602 | 0,728103011 | 835640 |
| 12 | **22** | 40179 | 40320 | 40252 | 40336 | 40256 | 40179 | 15,1430349 | 0,688319769 | 883938 |
| 13 | **24** | 38992 | 38593 | 38759 | 38555 | 38628 | 38555 | 15,7808845 | 0,657536852 | 925320 |

* + - 1. **диаграма на ускорението Sp**
      2. **диаграма на ефективността Еp**
      3. **диаграма на цената Cp**

1. **Анализ на получените резултати**
   1. **при статично балансиране**

При програмата със статичното балансиране тестовете, в при параметъра големина на изображението колкото по-голямо става размера, толкова по-добро ускорение се получаваше. При най-малкия размер 640x480 ускорението е най-малко като след 12-14 нишки, то нараства значително по-бавно. От това следва и че при този размер ефективността намалява най-бързо.

При тестовете на различните области постигнатите ускорения са сравнително близки, максималното ускорение е около 15 при 24 нишки. Кривите на ефективността са идентични. При област 2 цената **Cp** е най-висока, което се дължи на това, че времената за изпълнение са най-бавните от трите области.

Тестовете с различен брой итерации показват подобни криви като тестовете с различния размер на изображението, колкото повече итерациите растат, толкова по-добро ускорение е постигнато.

При резултатите за ускорението не е наблюдавана нито суперлинейна аномалия, нито немонотонна аномалия.

* 1. **при динамично балансиране**

При тестването с различен размер на изображението видът на кривите на ускорението, ефективността и цената са аналогични с тези при статичното балансиране. При най-големи тестван размер е постигнато ускорение от 16,05, което е с 0,5 по-добро от това при статичното балансиране.

При тестването с различни области ускорението е по-добро средно с 0,6 от това при статичното балансиране.

Тестовете с различен брой итерации са идентични с тези при статичното.

* 1. **заключение**

Трябва да се отбележи факта, че при динамичното балансиране времената за изпълнение са с около 0,5 - 1 секунда по-бавни, но тъй като е постигнато малко по-високо ускорение, цената Cp и при двата начина на балансиране е еднаква.

Въпреки това динамичното балансиране е за предпочитане пред статичното, когато разглеждаме небалансирани области, в които постоянно се редуват точки вътре и извън множеството на фрактала.

1. **Източници**

[1]Bastian Fredriksson, “An introduction to the Mandelbrot set”, публикуван: 01.2015г, (<https://www.kth.se/social/files/5504b42ff276543e4aa5f5a1/An_introduction_to_the_Mandelbrot_Set.pdf>)

[2]Martin Johnson, “Massey University, Parallel Computing, lecture 3 Parallel techniques, lecture 7 Asynchronous computations”, 2009г, (https://www.massey.ac.nz/~mjjohnso/notes/59735/)

[3]Douglas Thain, „University of Notre Dame, Operating System Principles, Project 3“, 2020г, (https://www3.nd.edu/~dthain/courses/cse30341/spring2020/)

[4]Thomas Uhrig, “Calculating Mandelbrot Set with Java Tasks”, публикуван: 05.11.2012г, (<https://tuhrig.de/calculating-mandelbrot-set-with-java-tasks/>)

(<https://bitbucket.org/wordless/mandelbrot/src/master/Mandelbrot/>)

[5]Brian Goetz, “Java Concurrency In Practice” 2006, издател : “Addison-Wesley”

[6]Eugen Paraschiv, “LinkedBlockingQueue vs ConcurrentLinkedQueue”, последна модификация:03.06.2020г,

(<https://www.baeldung.com/java-queue-linkedblocking-concurrentlinked>)

1. **Списък с фигури**

Фигура 1. Декомпозиция на данните на блокове и статично балансиране на изпълнението, източник [3] (<https://tuhrig.de/calculating-mandelbrot-set-with-java-tasks/>)

Фигура 2. Декомпозиция на данните по редове и статично балансиране на изпълнението, източник [3]( <https://tuhrig.de/calculating-mandelbrot-set-with-java-tasks/>)

Фигура 3. Клас диаграма

Фигура 4. Изпълнение на програмата с опцията за показване на помощ

Фигура 5. Изпълнение на програмата с опцията за показване на помощ

Фигура 6. фрактал, област -2.0:2.0:-2.0:2.0, оригинален размер 1920x1440

Фигура 7. област -2.0:2.0:-2.0:2.0, оригинален размер 1920x1440

Фигура 8. област -1.5:2.0:-0.25:0.25, оригинален размер 1920x1440

Фигура 9. област -0.75:0.75:0.0:1.25, оригинален размер 1920x1440