# « СРАВНИТЕЛЬНЫЙ АНАЛИЗ РАБОТЫ ПАРАЛЛЕЛЬНОГО АЛГОРИТМА МАСШТАБИРОВАНИЯ ГРАФИЧЕСКИХ ИЗОБРАЖЕНИЙ ДЛЯ МНОГОЯДЕРНЫХ CPU»

Выполнил Бугулов М.Р.
Научный руководитель: Мирошников А.С.

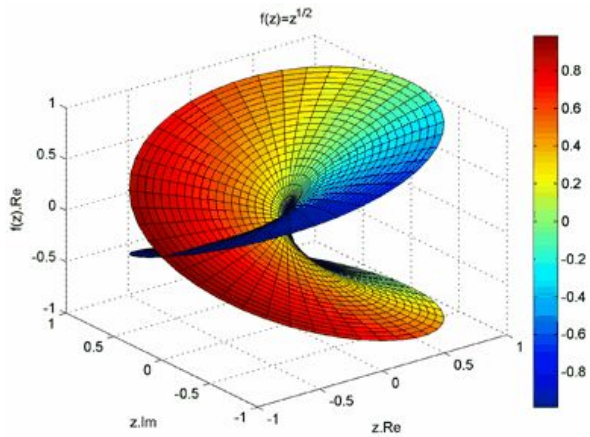
# Цели и задачи

- Цель: минимизация времени и сравнительный анализ работы параллельного алгоритма масштабирования изображений.

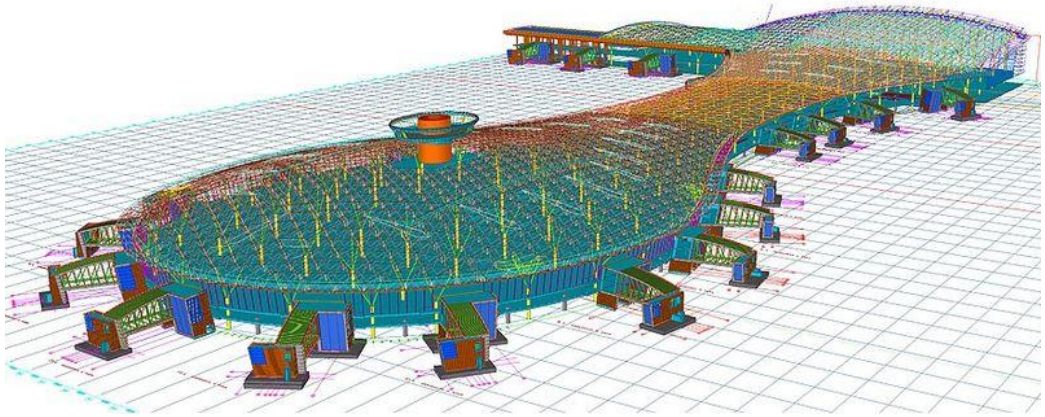
Задачи:

- Проведение аналитического обзора по данной теме;
- Выбор математической модели для минимизации времени масштабирования;
- Разработка параллельного алгоритма масштабирования;
- Программная реализация построенного алгоритма;
- Экспериментальная проверка эффективности выбранной математической модели и разработанного программного комплекса.

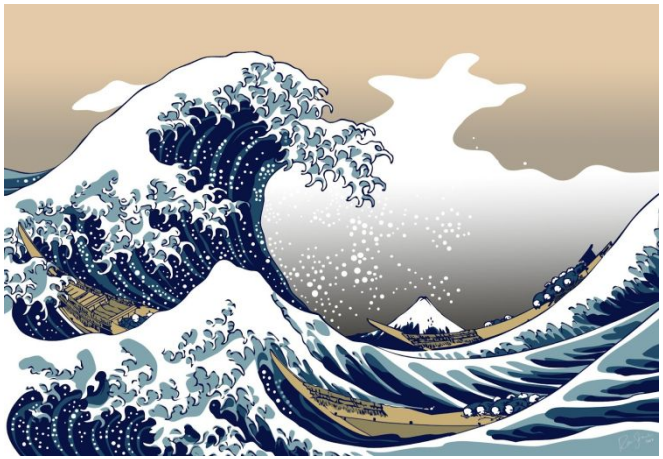
# Актуальность



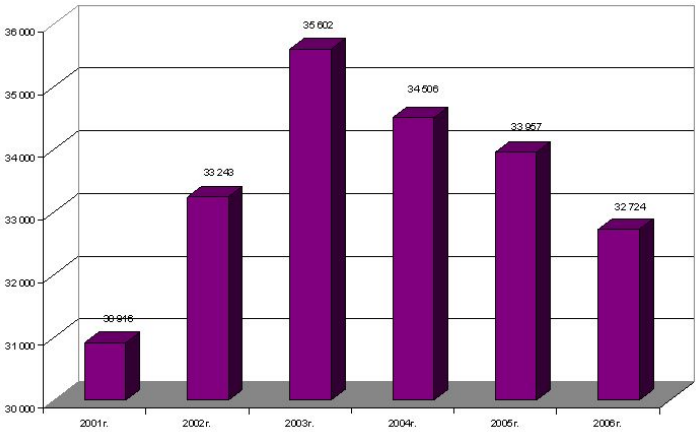

Научная графика



Конструкторская графика

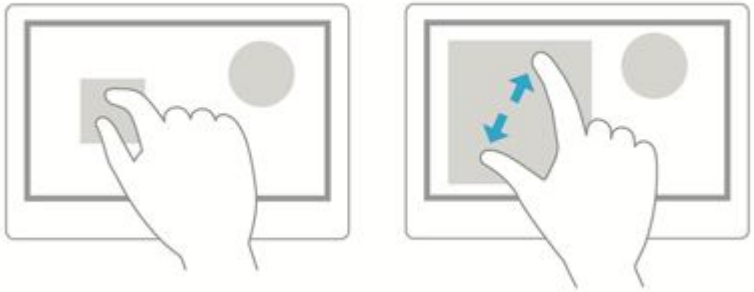


Художественная графика



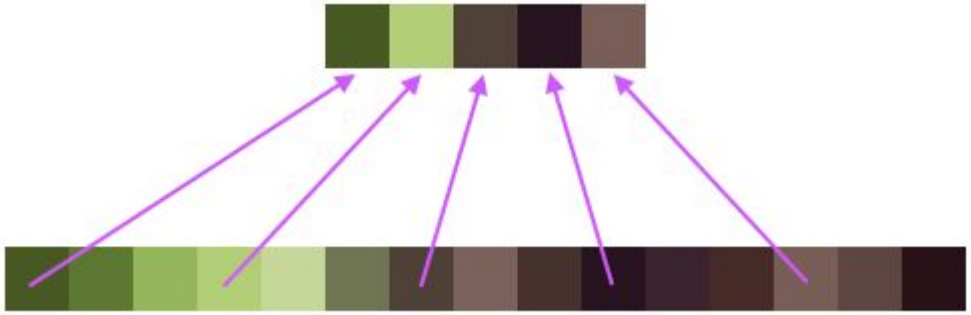

Деловая графика


ZOOM 28-300mm 1:3.5-6.3 MACRO
LENS MADE IN JAPAN

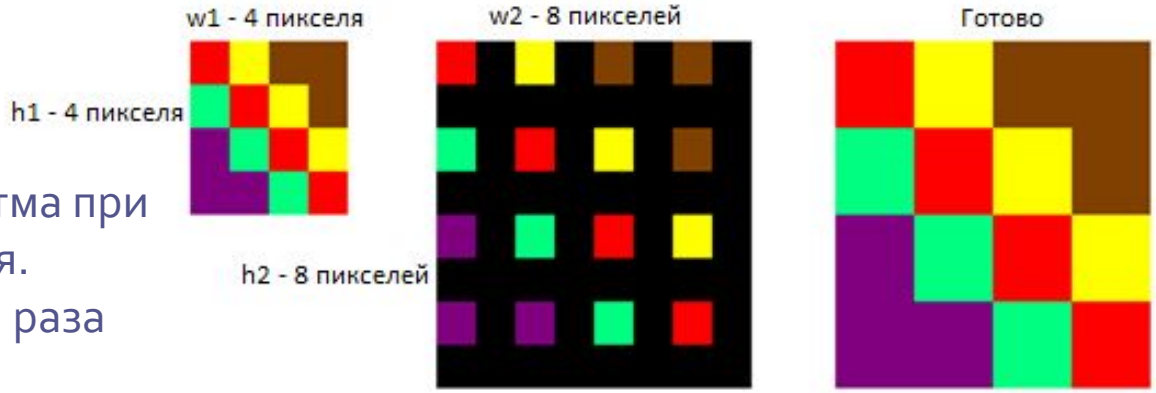



SAMSUNG
MKRU

# Описание алгоритма



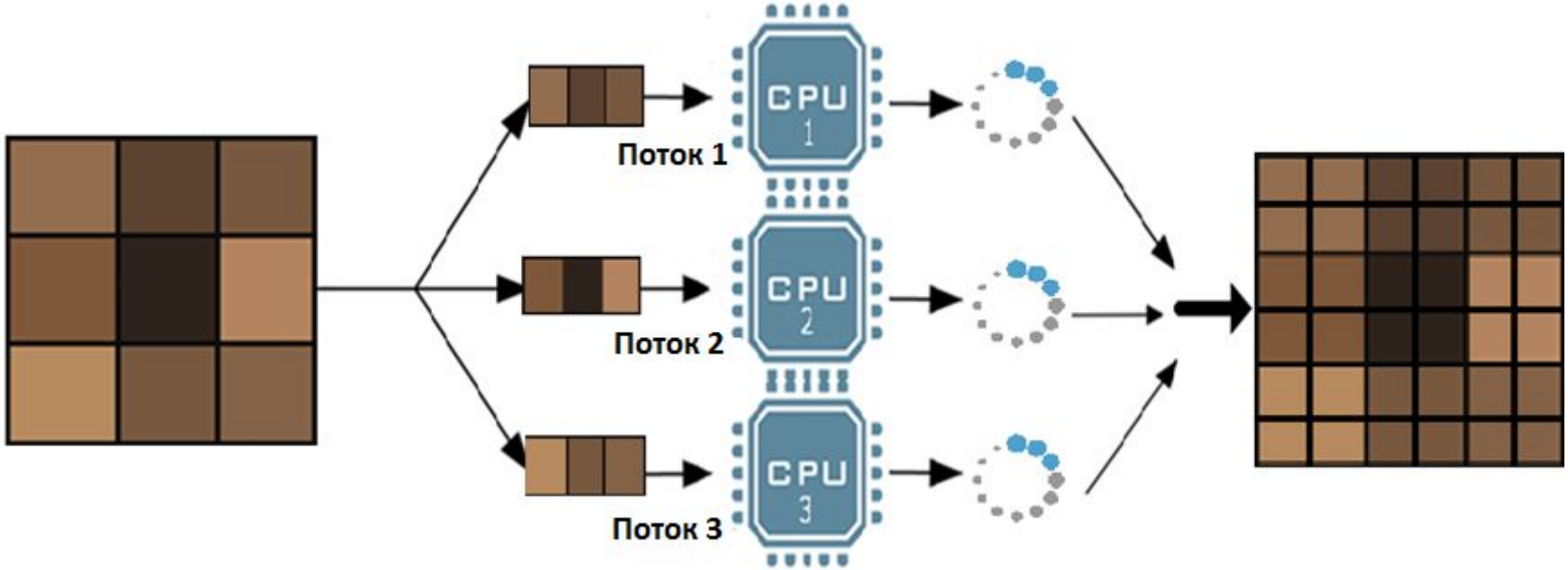
Принцип работы алгоритма при уменьшении изображения. Показано уменьшение в 3 раза.




Принцип работы алгоритма при увеличении изображения. Показано увеличение в 2 раза

# Пример масштабирования алгоритмом Nearest Neighbor

# Параллельная реализация алгоритма



# МАТЕМАТИЧЕСКАЯ МОДЕЛЬ

$$\begin{cases} T = c + f \cdot x + \dfrac{d(m,n)}{x} \to \min \\ 1 \le x \le P, \ \textit{целое} \end{cases} \qquad (1)$$

c – среднее время на организацию вычислений;

$f$ – среднее время на организацию работы одного потока;

$x$ – количество потоков, задействованных в параллельном алгоритме для масштабирования изображений;

$d(m,n)$ – время масштабирования изображения одним потоком;

$P$ – максимальное количество активных потоков, поддерживаемых CPU;

n – высота результирующего изображения;

m – ширина результирующего изображения.

# Расчет параметра d(m,n)

| | Стоимость | Повторы |
|---|---|---|
| Инициализация массива b | $C_1$ | 1 |
| For (i = 0; i < n; i++){ | $C_2$ | n |
| For (j = 0; j < m; j++) | $C_3$ | n*m |
| {процедура обработки пикселя b[i,j]} | $C_4$ | n*m |
| } | | |

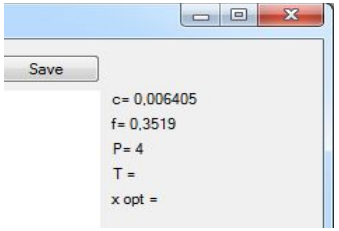
d(m, n) = C1+C2*n+C3*n*m+C4*n*m = (C3+C4)*n*m+C2*n+C1

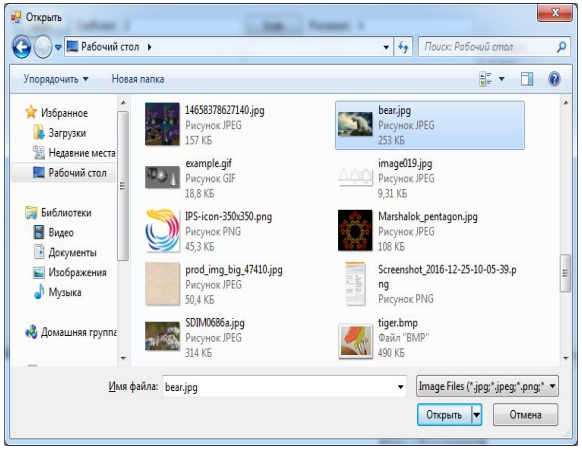
Обозначив *k1=C3+C4*, *k2=C2* и *k1=C1*,

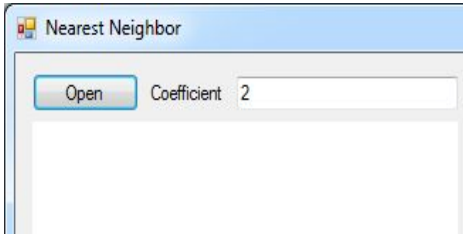
время работы алгоритма можно записать:

d(m,n) = k1*n*m+k2*n+k3. (2)

# Программный комплекс


Пример расчета оптимального количества пс

Дано изображение 1900x1900 px. P=4

$d(m,n)=1,84E\text{-}06*(n*m)+1,284E\text{-}07*n+9,8E\text{-}05$

$c =$ 6,361E-03 секунд

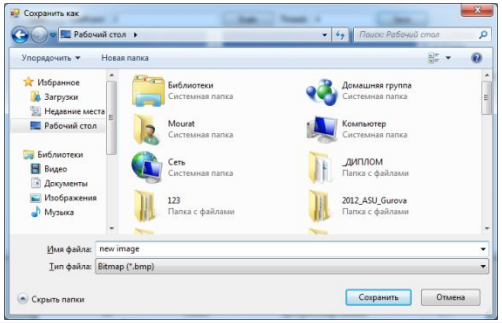
$f = 3,459E\text{-}01$ секунд.

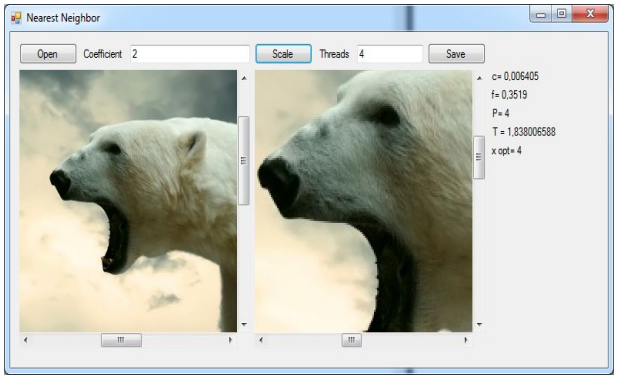
$$T'(x) = 0 + f + d * \left(-\frac{1}{x^2}\right) = 0$$

$$x = \sqrt{\frac{d}{f}} = \sqrt{\frac{1,84E-06(1900*1900)+1,284E-07*1900+9,8E-05}{3,459E-01}} = 4.36$$

Т.к. x > P, то оптимальное количество потоков будет 4. T(x) = 4.36 с.

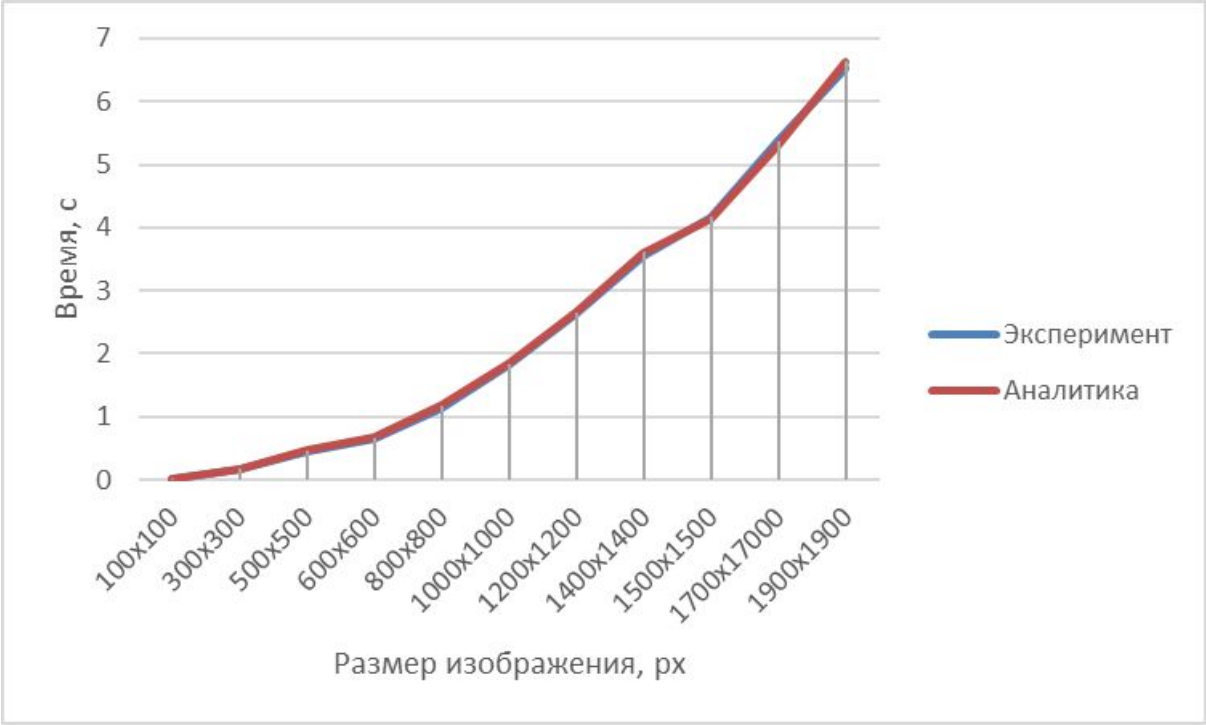
```
Thread t2 = new Thread(new ParameterizedThreadStart(scale2));
Thread t3 = new Thread(new ParameterizedThreadStart(scale3));
Thread t4 = new Thread(new ParameterizedThreadStart(scale4));
Thread t5 = new Thread(new ParameterizedThreadStart(scale5));
```



# Эксперимент 1

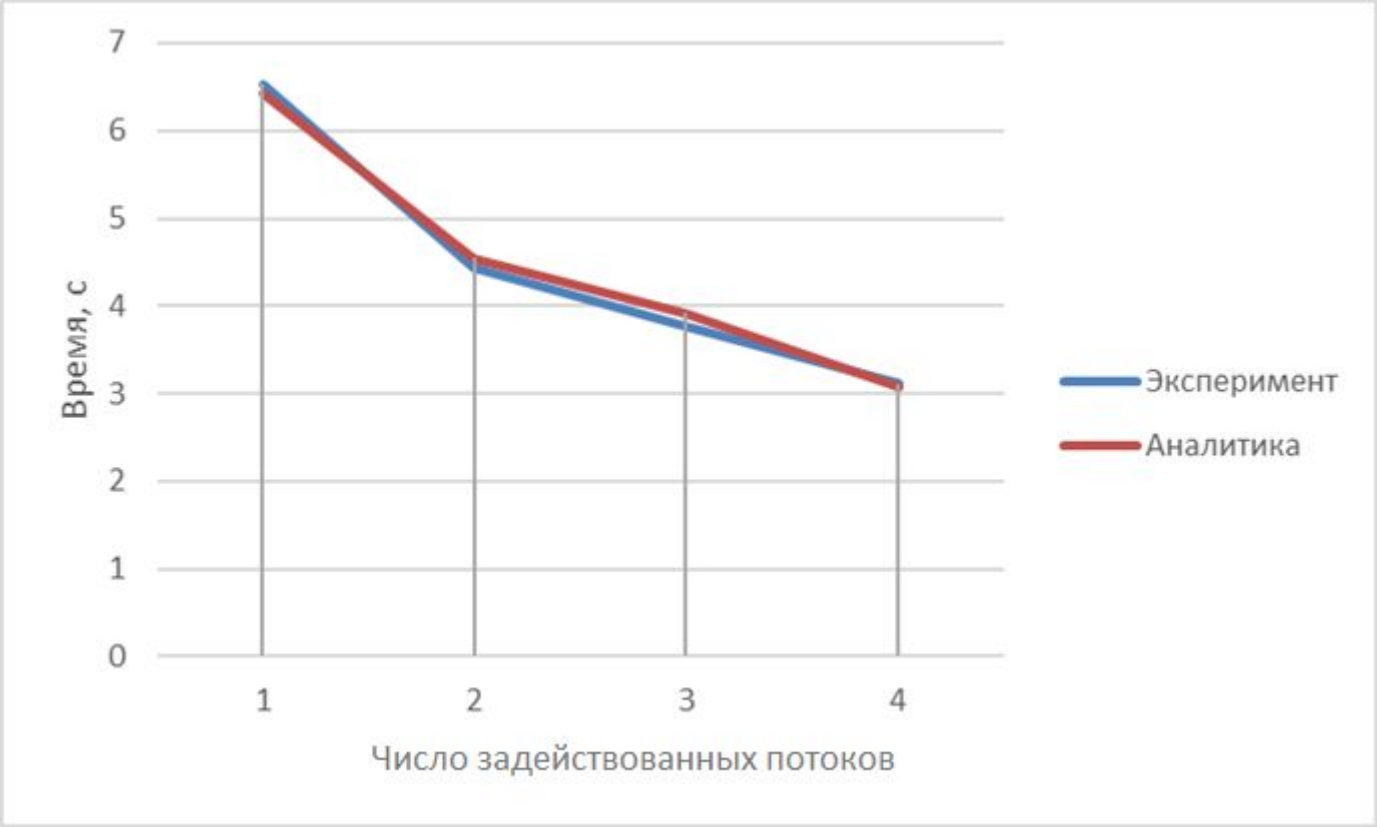
| Размер изображения, px | Эксперимент, с | Аналитика, с |
|---|---|---|
| 100x100 | 0,02087974 | 0,018733 |
| 300x300 | 0,16296232 | 0,165857 |
| 500x500 | 0,44846666 | 0,460105 |
| 600x600 | 0,64442029 | 0,662401 |
| 800x800 | 1,1347506 | 1,177335 |
| 1000x1000 | 1,80897122 | 1,839394 |
| 1200x1200 | 2,63456706 | 2,648576 |
| 1400x1400 | 3,55396282 | 3,604883 |
| 1500x1500 | 4,16899386 | 4,138208 |
| 1700x1700 | 5,37627342 | 5,315201 |
| 1900x1900 | 6,53036216 | 6,639319 |




k1=1,84E-06
k2= 1,284E-07
K3=9,8E-05

# Эксперимент 2

| P | Эксперимент | Аналитика |
|---|---|---|
| 1 | 6,53036216 | 6,639319 |
| 2 | 4,43177562 | 4,5483575 |
| 3 | 3,86364146 | 3,9181778 |
| 4 | 3,11221826 | 3,0392697 |



# Эксперимент 3

| Размер\ потоки | 1 | 4 |
|---|---|---|
| 100x100 | 0,02088 | 0,008942 |
| 300x300 | 0,162962 | 0,079999 |
| 500x500 | 0,448467 | 0,219866 |
| 600x600 | 0,64442 | 0,313288 |
| 800x800 | 1,134751 | 0,57777 |
| 1000x1000 | 1,80897 | 0,881983 |
| 1200x1200 | 2,634567 | 1,284812 |
| 1400x1400 | 3,553962 | 1,753722 |
| 1500x1500 | 4,168994 | 1,968062 |
| 1700x17000 | 5,376273 | 2,473851 |
| 1900x1900 | 6,530362 | 3,112218 |

# Заключение

- 1. Был проведен аналитический обзор предметной области;
- 2. Была выбрана математическая модель, минимизирующая время работы алгоритма масштабирования;
- 3. Был разработан параллельный алгоритм  масштабирования;
- 4. Создан программный комплекс, реализующий параллельный алгоритм Nearest Neighbor;
- 5. Был проведен сравнительный анализ параллельного алгоритма с его последовательной реализацией.

# Программный комплекс

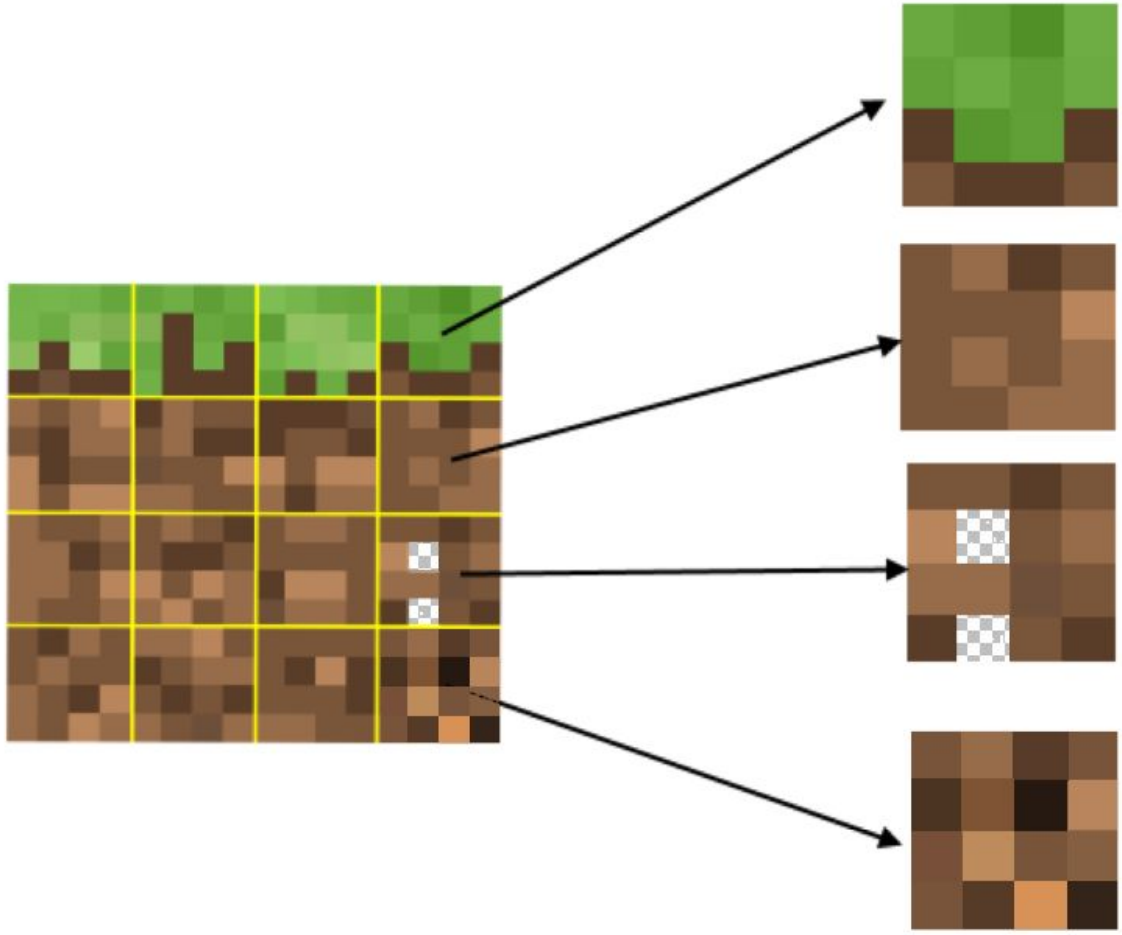
# Одна из важных частей при создании 3D сцен – текстуры моделей

# Текстуры позволяют имитировать жизненные сцены.

# Группа алгоритмов DXT

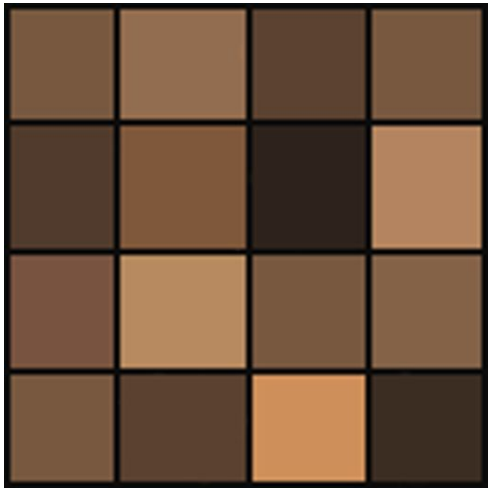
- DXT1 – сжатие текстур с однобитным альфа-каналом
- DXT3 – сжатие текстур с четырёхбитным альфа-каналом, содержащим произвольные значения
- DXT2 – аналогично DXT3, но с предумножением цвета на альфа-канал
- DXT5 – сжатие текстур с восьмибитным альфа-каналом, содержащим табличные значения
- DXT4 – аналогично DXT5, но с предумножением цвета на альфа канал

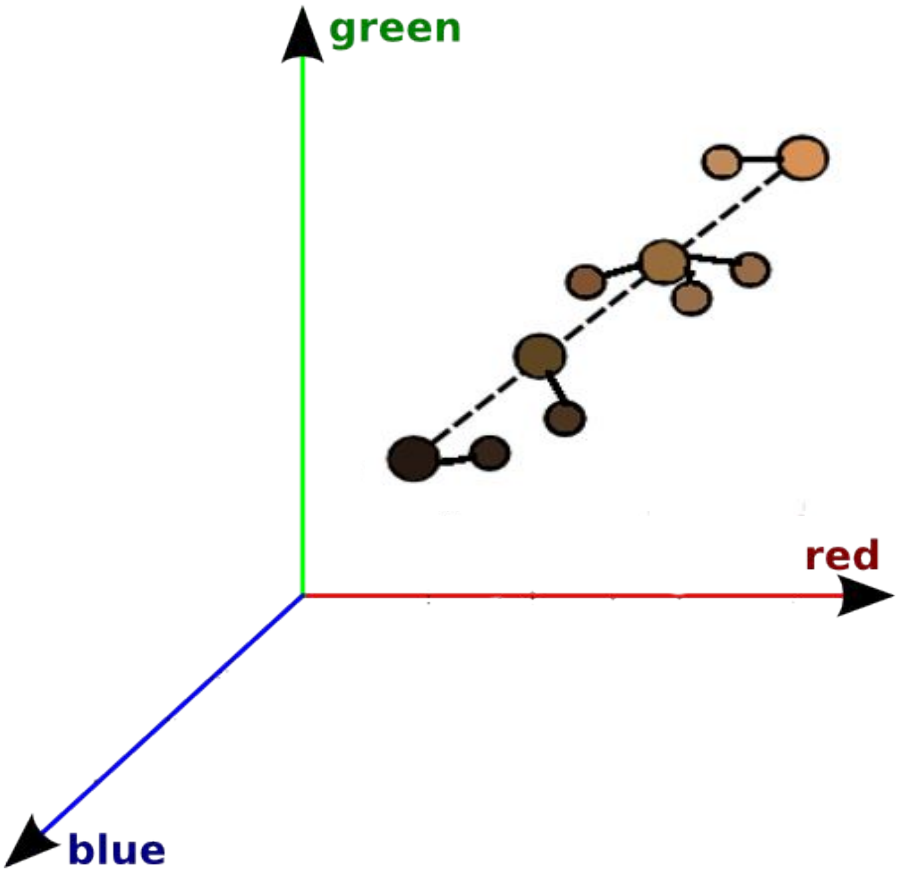
# Разбиение на блоки



# Пример обработки блока

Обрабатываемый блок
4x4 пикселя

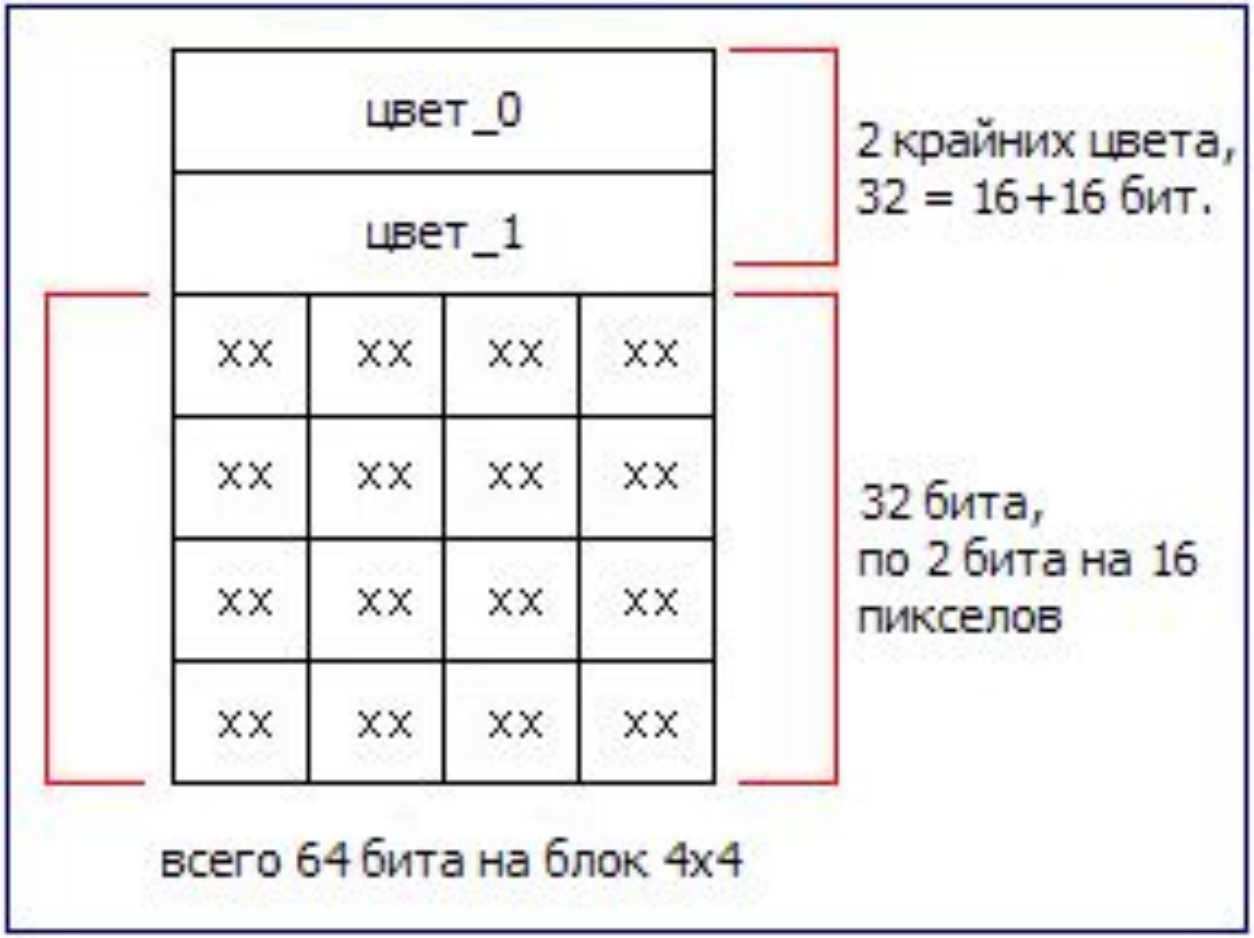




# Формирование таблицы


$$C_2 = \frac{1}{2} C_0 + \frac{1}{2} C_1$$

# Структура сжатого блока



# Результат DXT1

Несжатое изображение 1020* 960 px
3.7 MB

DXT1 сжатие 1020* 960 px
0,46 MB

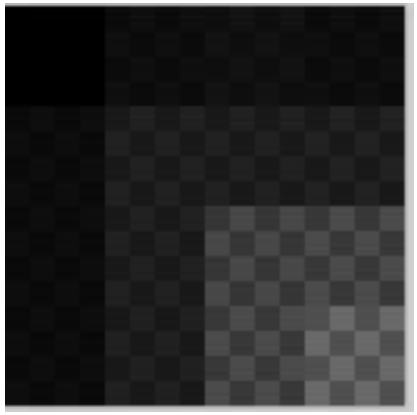
# DXT - сжатие с потерями

Увеличенный участок несжатого изображения

Увеличенный участок восстановленного изображения

# DXT3

## Преобразование альфа канала



Исходное изображение

Восстановленное изображение

| | | | |
|---|---|---|---|
| 255 | 240 | 240 | 240 |
| 240 | 220 | 220 | 220 |
| 240 | 220 | 180 | 180 |
| 240 | 220 | 180 | 150 |

Альфа исходного изображения

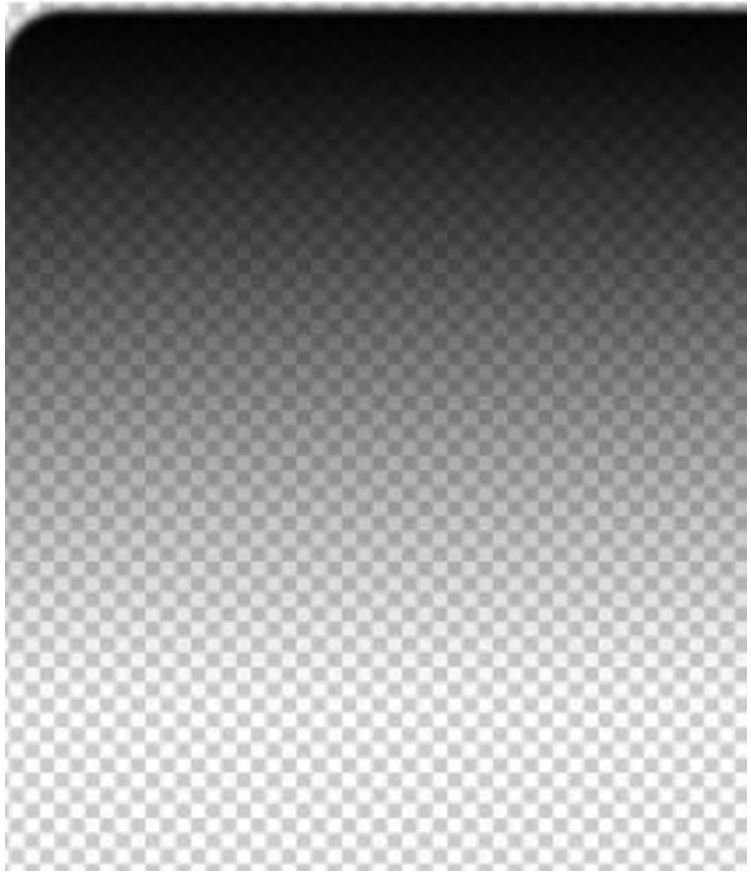
| | | | |
|---|---|---|---|
| 15 | 14 | 14 | 14 |
| 14 | 13 | 13 | 13 |
| 14 | 13 | 11 | 11 |
| 14 | 13 | 11 | 9 |

Альфа в сжатом блоке

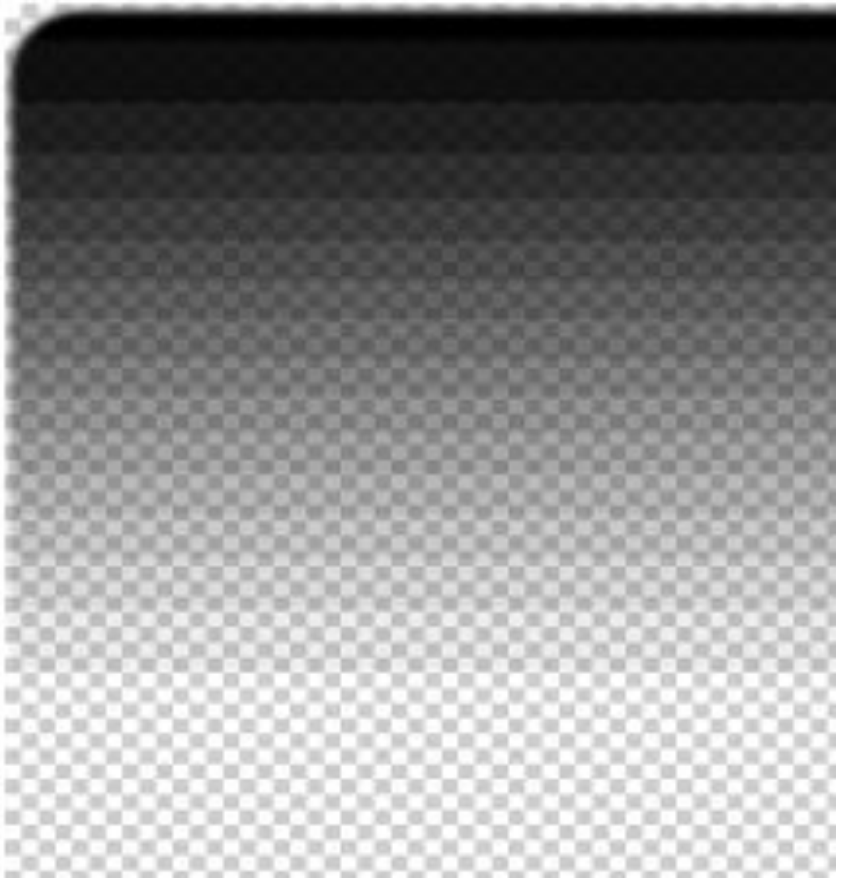
| | | | |
|---|---|---|---|
| 255 | 238 | 238 | 238 |
| 238 | 221 | 221 | 221 |
| 238 | 221 | 187 | 187 |
| 238 | 221 | 187 | 153 |

Альфа в восстановленном блоке

# Результат DXT3

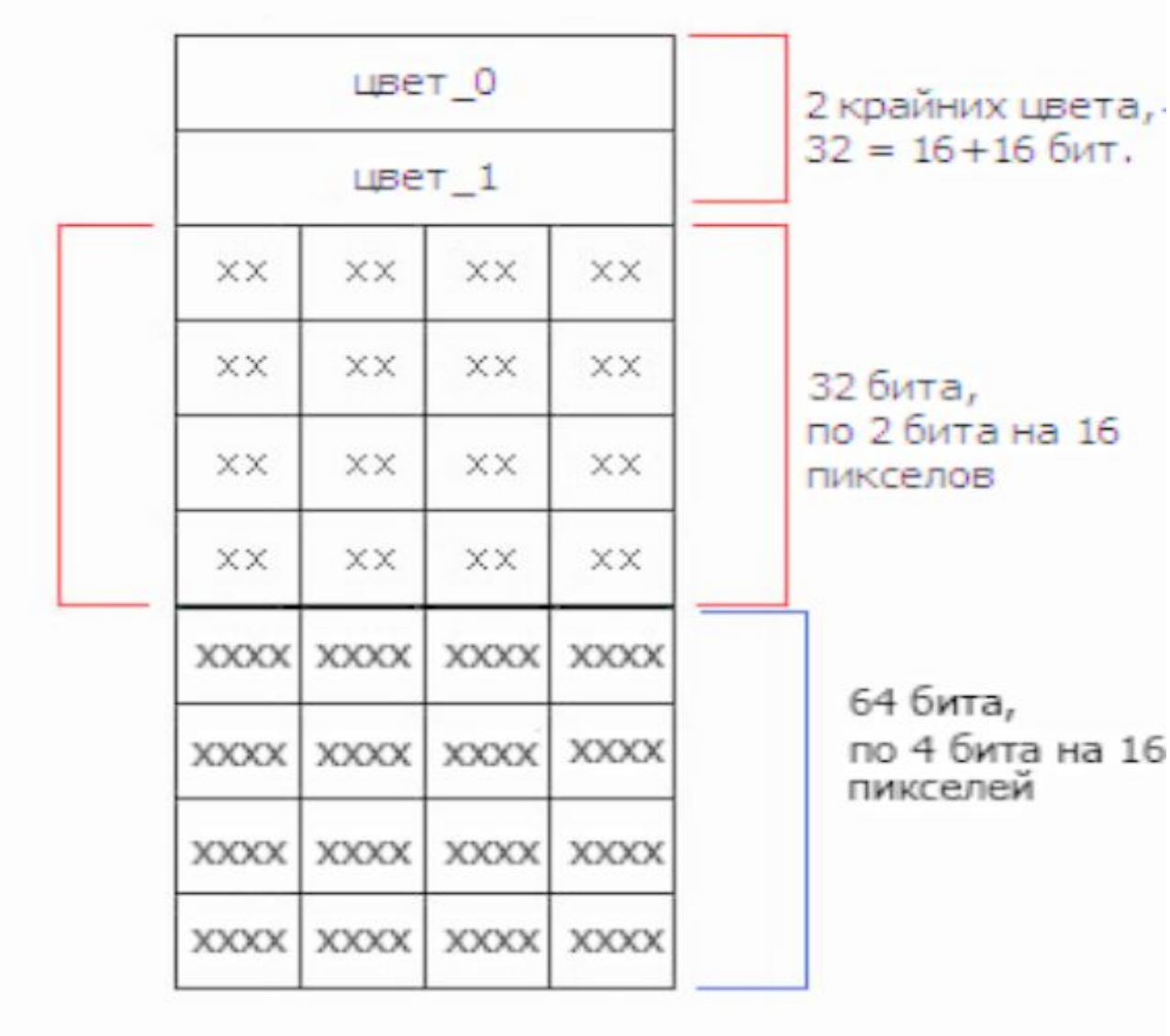


Несжатое изображение
1,47 MB

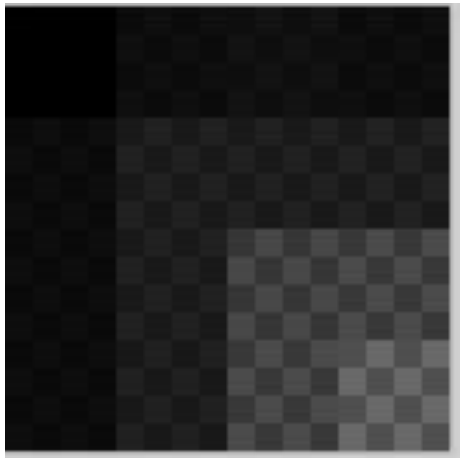


Сжатое изображение
0,35 MB

# Структура DXT3 блока



# DXT5. Пример

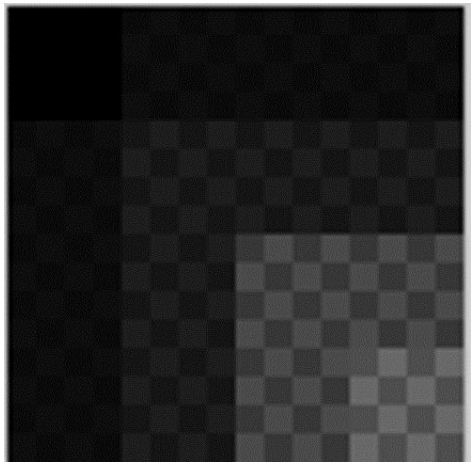


Исходный блок

| | | | |
|---|---|---|---|
| 255 | 240 | 240 | 240 |
| 240 | 220 | 220 | 220 |
| 240 | 220 | 180 | 180 |
| 240 | 220 | 180 | 150 |

Альфа исходного изображения

| |
|---|
| A0 = 255 |
| A1=150 |
| A2=240 |
| A3=225 |
| A4=210 |
| A5=195 |
| A6=180 |
| A7=165 |



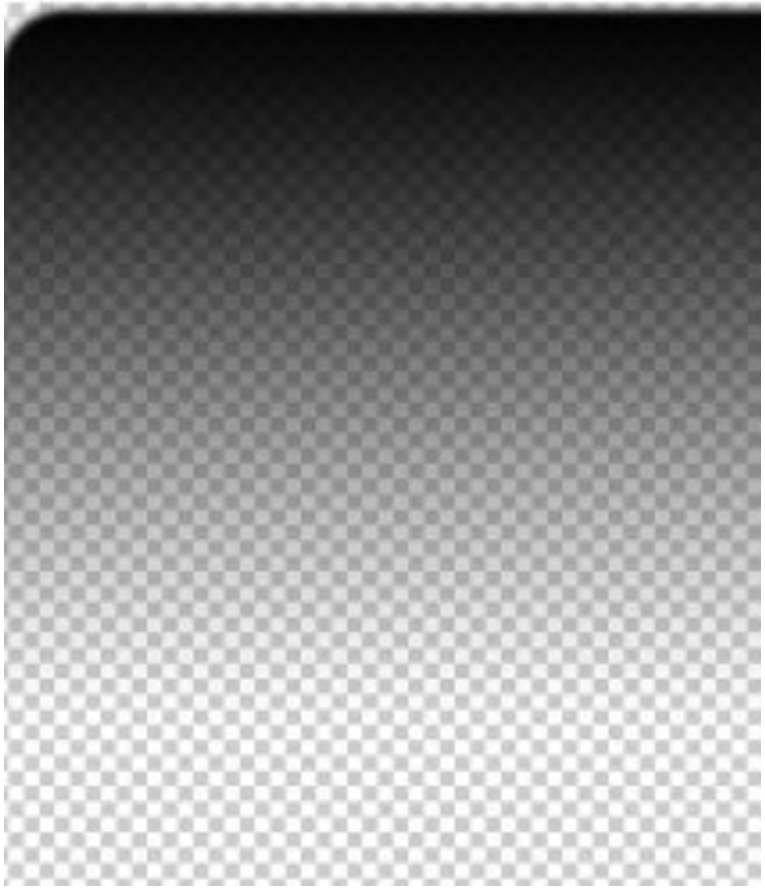
Восстановленное изображение

| | | | |
|---|---|---|---|
| 255 | 240 | 240 | 240 |
| 240 | 225 | 225 | 225 |
| 240 | 225 | 180 | 180 |
| 240 | 225 | 180 | 150 |

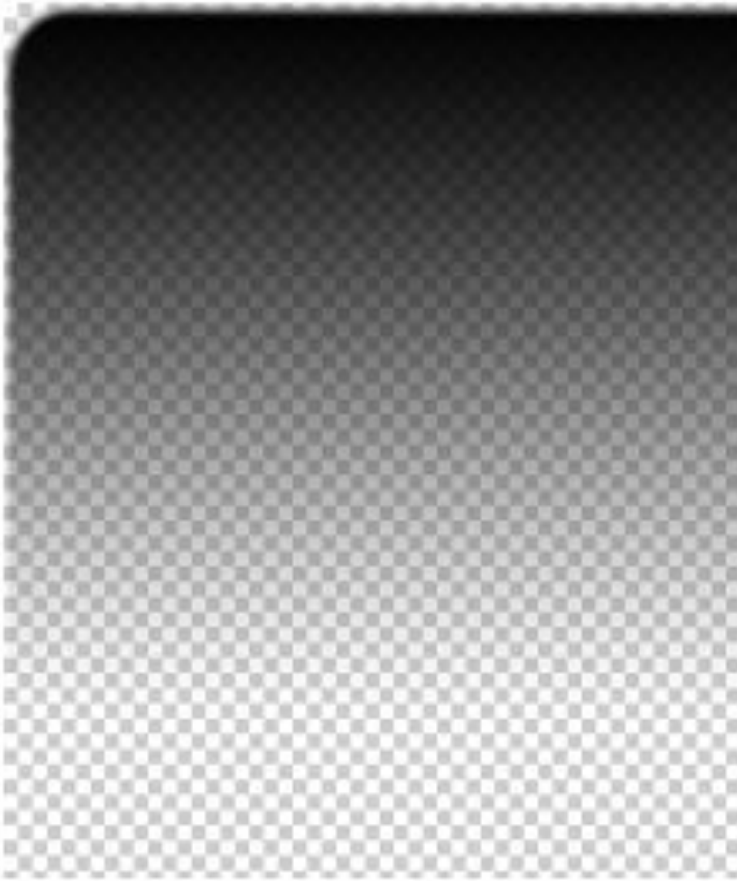
Альфа восстановленном блоке

Таблица индексов

# DXT5

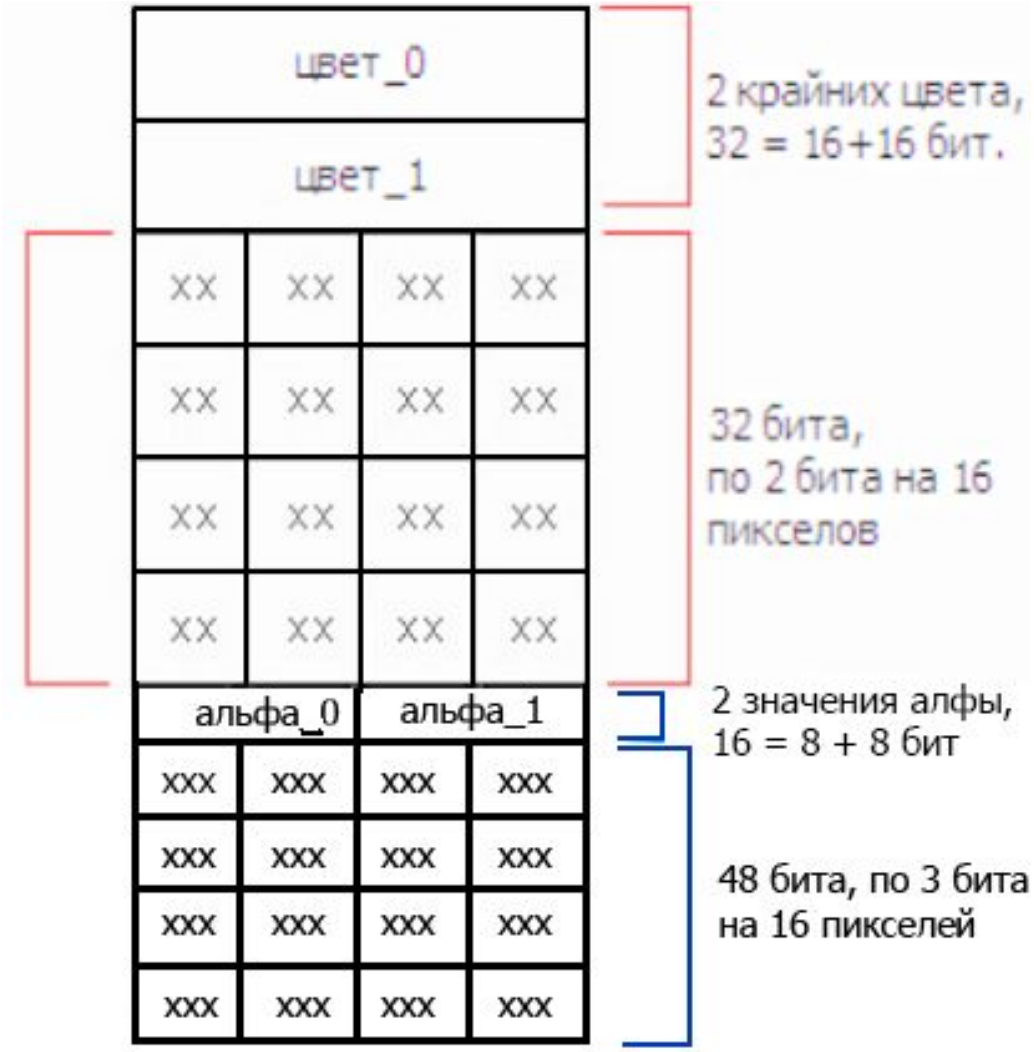


Несжатое изображение
1,47 MB

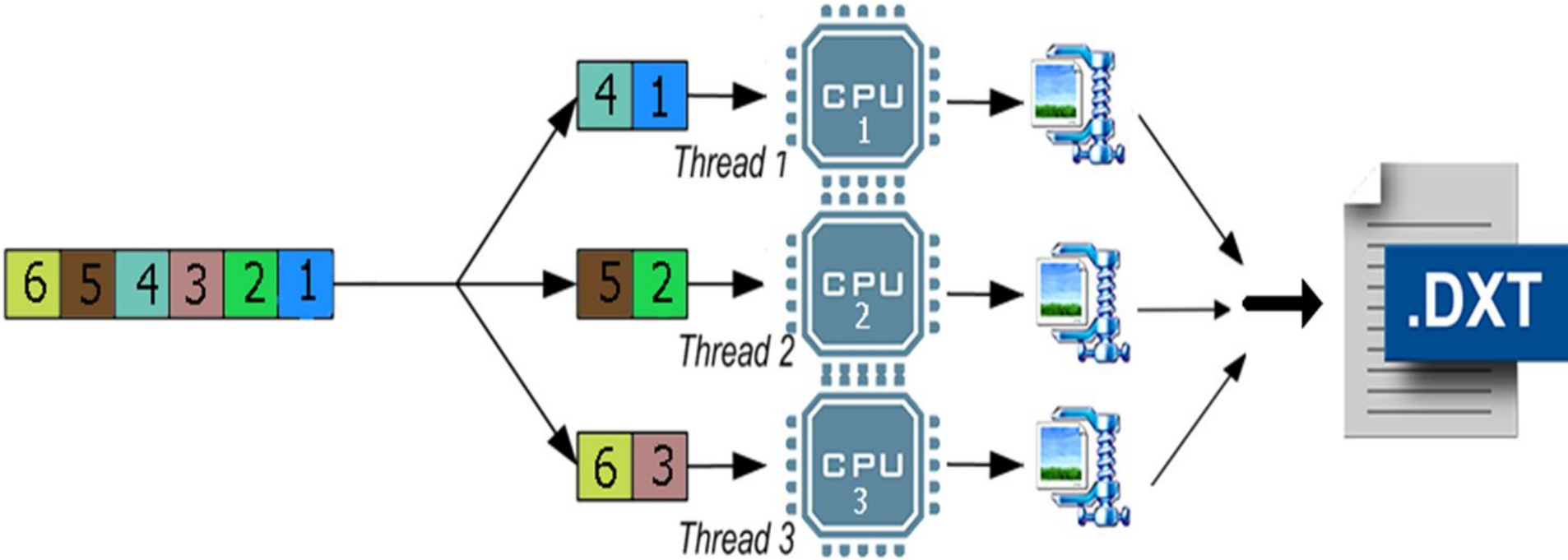


Сжатое изображение
0,35 MB

# Структура DXT5 блока



# Параллельный алгоритм



# МАТЕМАТИЧЕСКАЯ МОДЕЛЬ

$$\begin{cases} T = c + f \cdot x + \dfrac{d}{x} \to \min \\ 1 \le x \le P \end{cases} \qquad (1)$$

Время работы алгоритма можно записать следующим образом:

$$d = T(m, n) = k_1 (m * n) + k_2 \qquad (2)$$

c – среднее время на организацию вычислений;

$f$ – среднее время на организацию одного потока;

$x$ – количество потоков, задействованных в параллельном сжатии;

$d$ - время сжатия изображения одним потоком;

$P$ - максимальное количество активных потоков, поддерживаемых CPU;

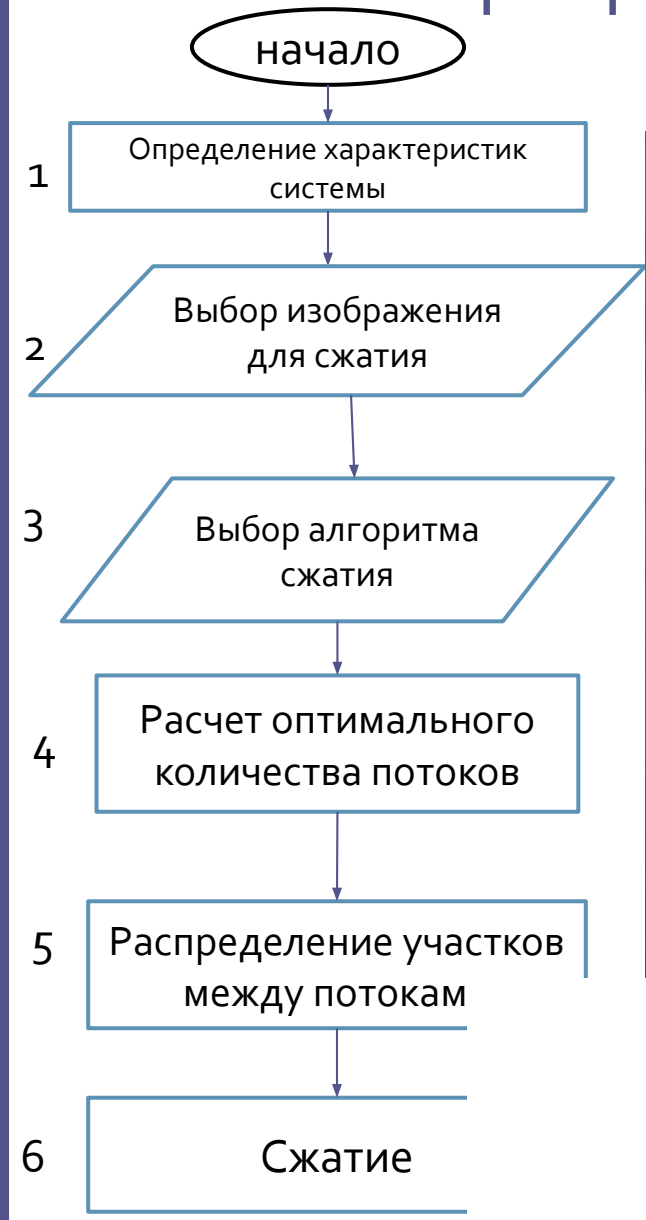
n - высота изображения;

m – ширина изображения;

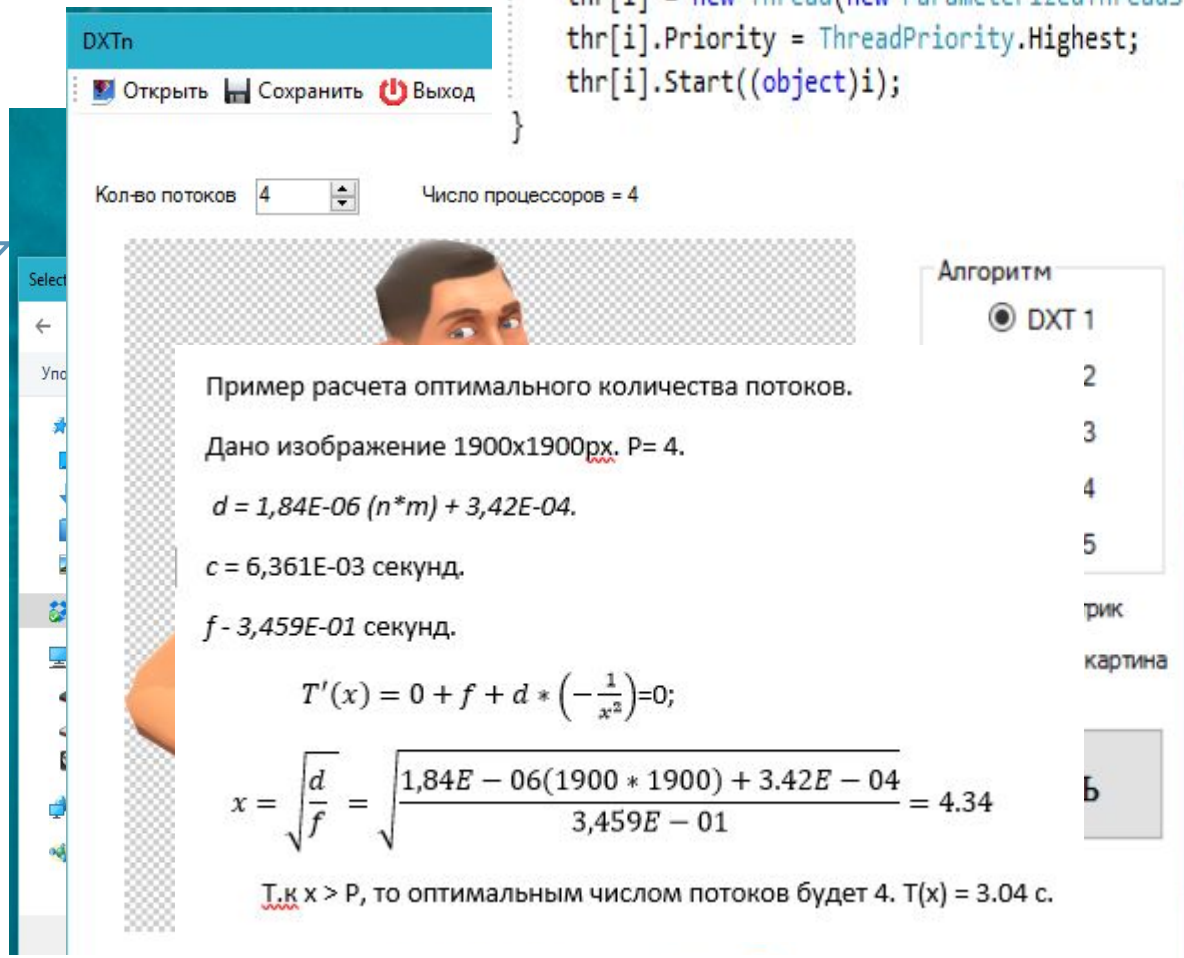
k1 – время обработки одного блока;

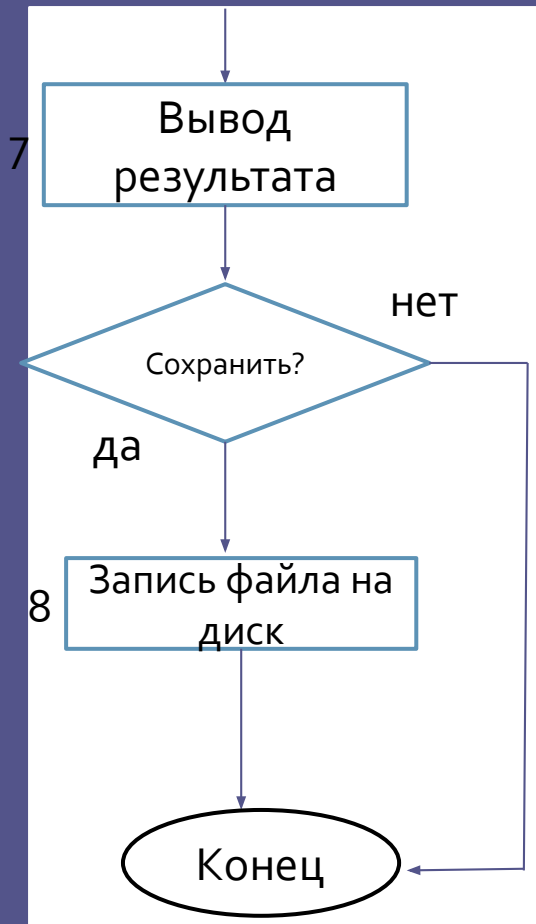
k2 – среднее время на организацию вычислений

# Программный комп




```
for (int i = 0; i < w; i++)
{
    thr[i] = new Thread(new ParameterizedThreadStart(
    thr[i].Priority = ThreadPriority.Highest;
    thr[i].Start((object)i);
}
```

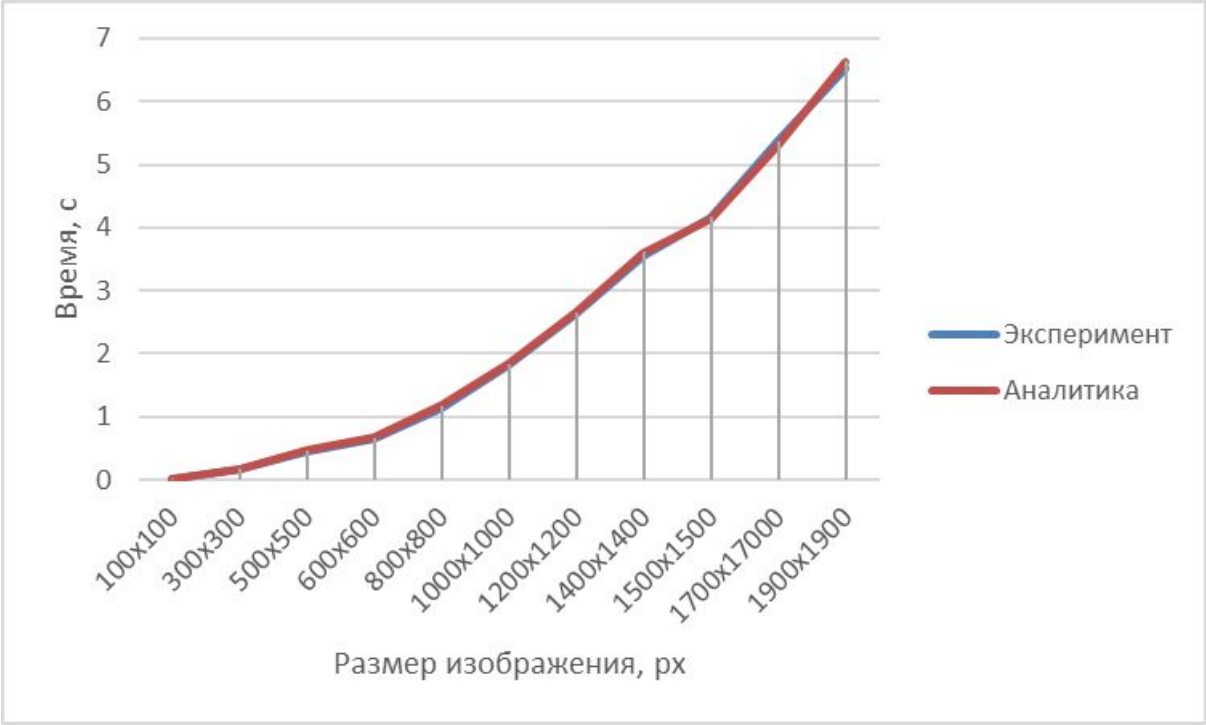



7
Вывод результата
нет
Сохранить?
да
8
Запись файла на диск
Конец


DXTn
Открыть
Сохранить
Выход
Кол-во потоков
4
Число процессоров = 4
.DXT
Zoom
75%
Разрешение: 1020 x 960
Размер исходного файла: 3,739 MB
Размер несжатого изображения: 3,735 MB
00:00:01.4344351
Размер: 956,252 KB

# Эксперимент 1

| Размер изображения, px | Эксперимент, с | Аналитика, с |
|---|---|---|
| 100x100 | 0,02087974 | 0,018733 |
| 300x300 | 0,16296232 | 0,165857 |
| 500x500 | 0,44846666 | 0,460105 |
| 600x600 | 0,64442029 | 0,662401 |
| 800x800 | 1,1347506 | 1,177335 |
| 1000x1000 | 1,80897122 | 1,839394 |
| 1200x1200 | 2,63456706 | 2,648576 |
| 1400x1400 | 3,55396282 | 3,604883 |
| 1500x1500 | 4,16899386 | 4,138208 |
| 1700x1700 | 5,37627342 | 5,315201 |
| 1900x1900 | 6,53036216 | 6,639319 |

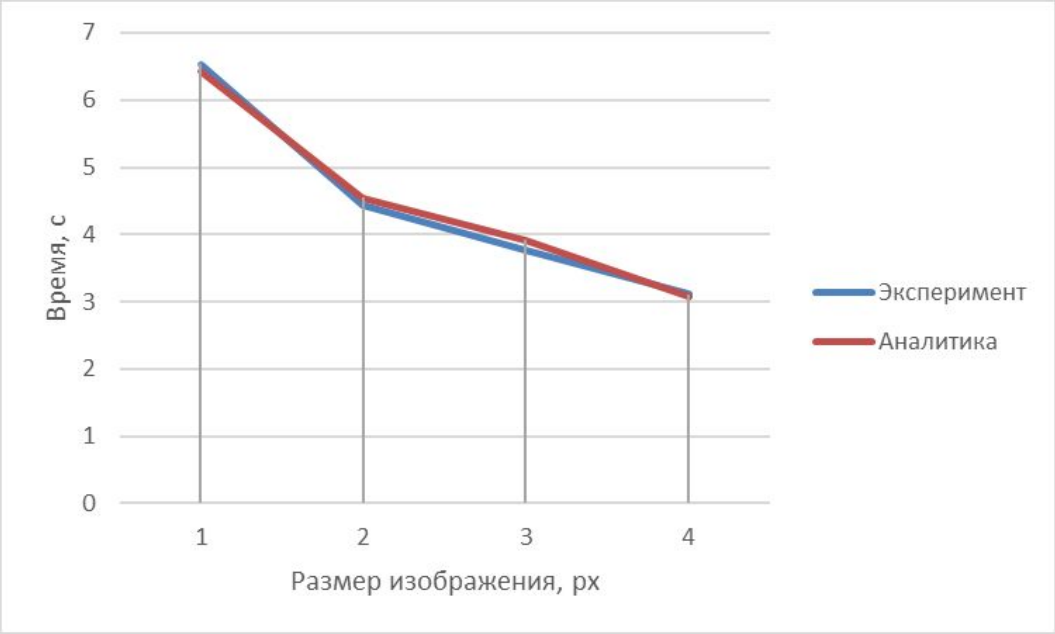



k1=1,84E-06
k2= 3,42E-04

# Эксперимент 2

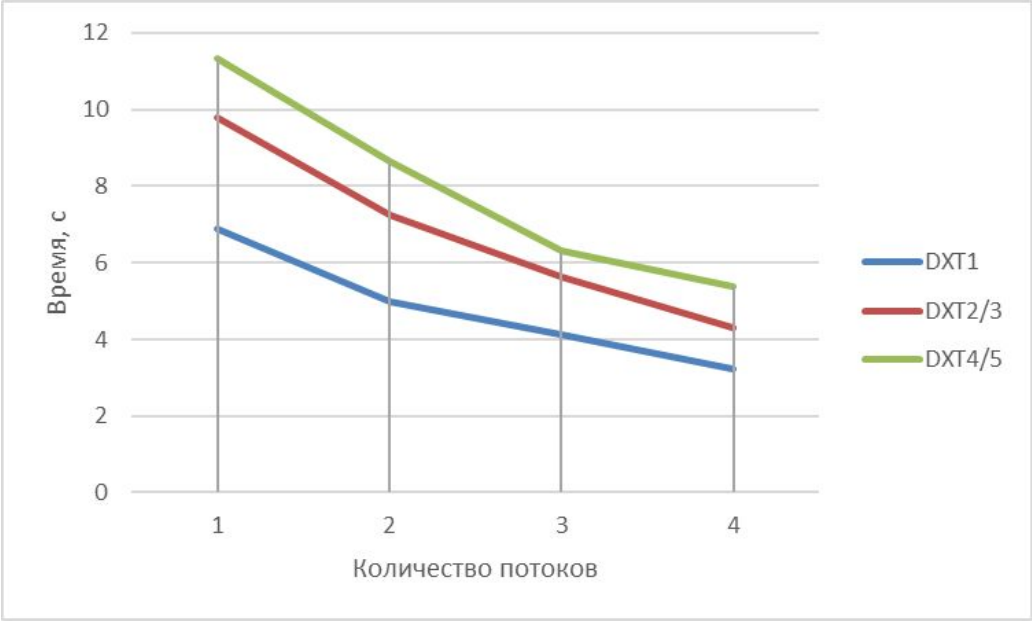
DXT1

| P | Эксперимент | Аналитика |
|---|---|---|
| 1 | 6,53036216 | 6,639319 |
| 2 | 4,43177562 | 4,5483575 |
| 3 | 3,86364146 | 3,9181778 |
| 4 | 3,11221826 | 3,0392697 |



# Эксперимент 3

| Размер\потоки | 1 | 4 |
|---|---|---|
| 100x100 | 0,02088 | 0,008942 |
| 300x300 | 0,162962 | 0,079999 |
| 500x500 | 0,448467 | 0,219866 |
| 600x600 | 0,64442 | 0,313288 |
| 800x800 | 1,134751 | 0,57777 |
| 1000x1000 | 1,80897 | 0,881983 |
| 1200x1200 | 2,634567 | 1,284812 |
| 1400x1400 | 3,553962 | 1,753722 |
| 1500x1500 | 4,168994 | 1,968062 |
| 1700x17000 | 5,376273 | 2,473851 |
| 1900x1900 | 6,530362 | 3,112218 |




| Алгоритм\потоки | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| DXT1 | 6,90211 | 4,99452 | 4,1231 | 3,22342 |
| DXT2/3 | 9,78657 | 7,23214 | 5,64125 | 4,288987 |
| DXT4/5 | 11,32468 | 8,659836 | 6,299455 | 5,383394 |

# Заключение

- 1. Был проведен сравнительный анализ предметной области.
- 2. Была выбрана математическая модель, минимизирующая время работы параллельного алгоритма.
- 3. Был разработан параллельный алгоритм  сжатия
- 4. Создан программный комплекс, реализующий параллельный DXTn алгоритмы.
- 5. Был проведен сравнительный анализ параллельных DXTn алгоритмов.