**Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский государственный технический университет имени Н.Э. Баумана**

**(национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)**

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.01 Информатика и вычислительная техника**

**О т ч е т**

**по лабораторной работе № 3**

**Название: Организация памяти конвейерных суперскалярных электронных вычислительных машин**

**Дисциплина: Архитектура электронно-вычислительных систем**

**Вариант: 15**

Студент гр. ИУ7-55Б О.Н.Талышева

(Подпись, дата) (И.О. Фамилия)

Преподаватель А.Ю. Попов

(Подпись, дата) (И.О. Фамилия)

Оглавление

[Цель работы 3](#_Toc180394444)

[Эксперимент 1: «Исследования расслоения динамической памяти» 4](#_Toc180394445)

[Эксперимент 2: «Сравнение эффективности ссылочных и векторных структур» 6](#_Toc180394446)

[Эксперимент 3: «Исследование эффективности программной предвыборки» 7](#_Toc180394447)

[Эксперимент 4: «Исследование способов эффективного чтения оперативной памяти» 8](#_Toc180394448)

[Эксперимент 5: «Исследование конфликтов в кэш-памяти» 9](#_Toc180394449)

[Эксперимент 6: «Сравнение алгоритмов сортировки» 10](#_Toc180394450)

[Идентификация процессора 11](#_Toc180394451)

[Контрольные вопросы 14](#_Toc180394452)

# Цель работы

Цель работы - освоение принципов эффективного использования подсистемы памяти современных универсальных ЭВМ, обеспечивающей хранение и своевременную выдачу команд и данных в центральное процессорное устройство. Работа проводится с использованием программы для сбора и анализа производительности PCLAB.

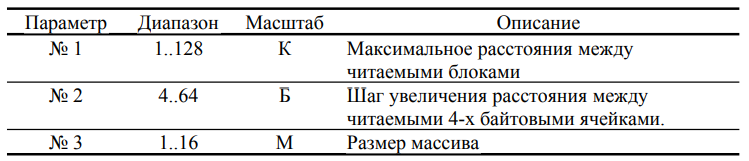
В ходе работы необходимо ознакомиться с теоретическим материалом, касающимся особенностей функционирования подсистемы памяти современных конвейерных суперскалярных ЭВМ, изучить возможности программы PCLAB, изучить средства идентификации микропроцессоров, провести исследования времени выполнения тестовых программ, сделать выводы о архитектурных особенностях используемых ЭВМ.

# Эксперимент 1: «Исследования расслоения динамической памяти»

Цель эксперимента: определение способа трансляции физического адреса, используемого при обращении к динамической памяти.

Исходные данные: размер линейки кэш-памяти верхнего уровня; объем физической памяти.

Для проведения эксперимента необходимо задать изменяемые параметры:



Результаты эксперимента: количество банков динамической памяти; размер одной страницы динамической памяти; количество страниц в динамической памяти.

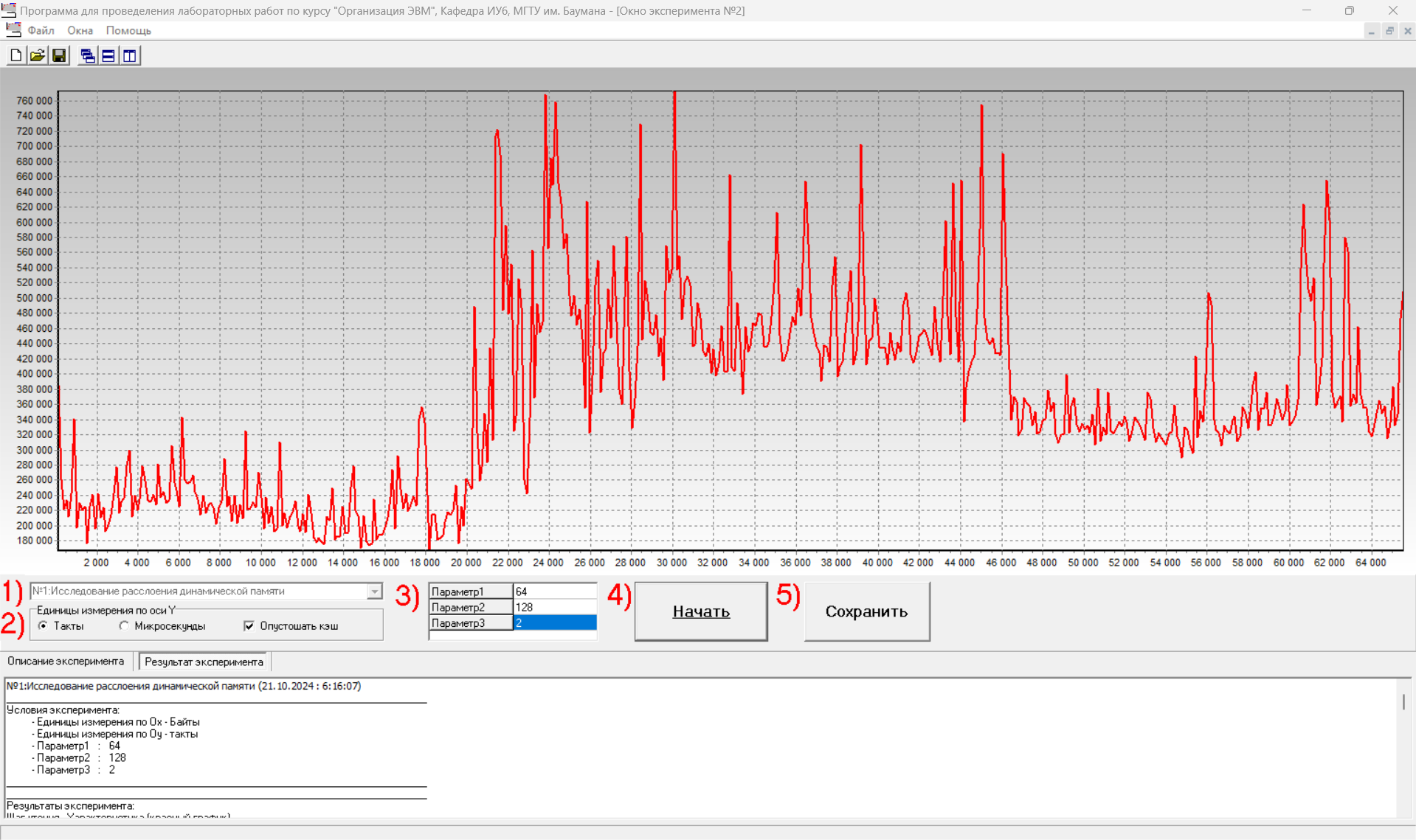


Рисунок 1 – Результат исследования расслоения динамической памяти (параметры: 64-128-2)

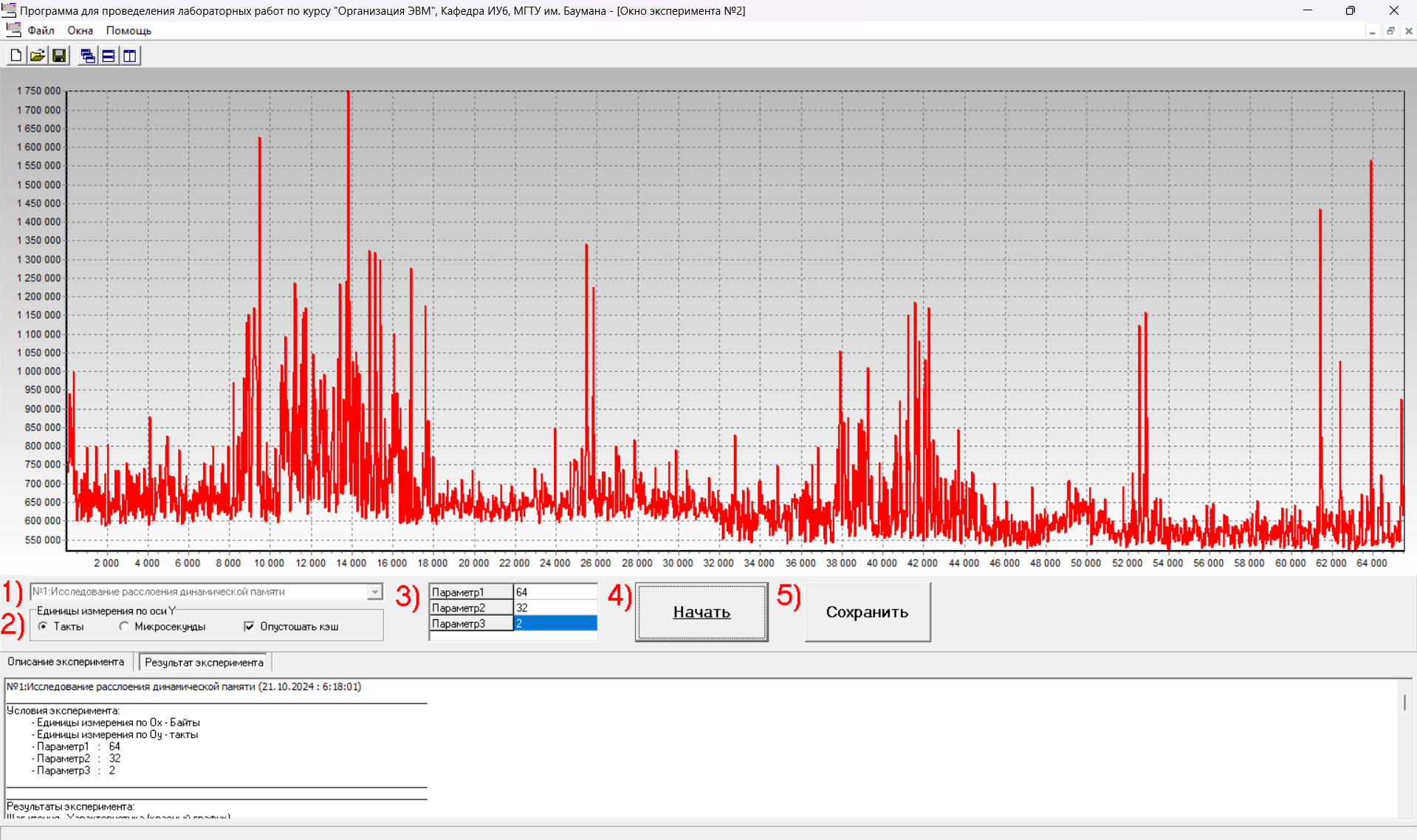


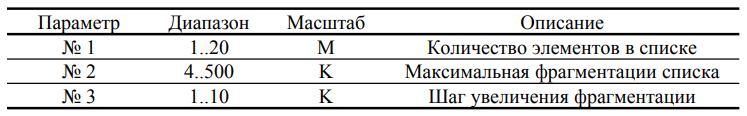
Рисунок 2 – Результат исследования расслоения динамической памяти (параметры: 64-32-2)

Вывод: мы узнали, что большой адресный шаг увеличивает время доступа, рандомный шаг даёт большее время доступа, чем последовательный; также мы увидели вживую принцип локальности.

# Эксперимент 2: «Сравнение эффективности ссылочных и векторных структур»

Цель эксперимента: оценка влияния зависимости команд по данным на эффективность вычислений.

Для проведения эксперимента требуется задать следующие настраиваемые параметры:



Результаты эксперимента: отношение времени работы алгоритма, использующего зависимые данные, ко времени обработки аналогичного алгоритма обработки независимых данных.

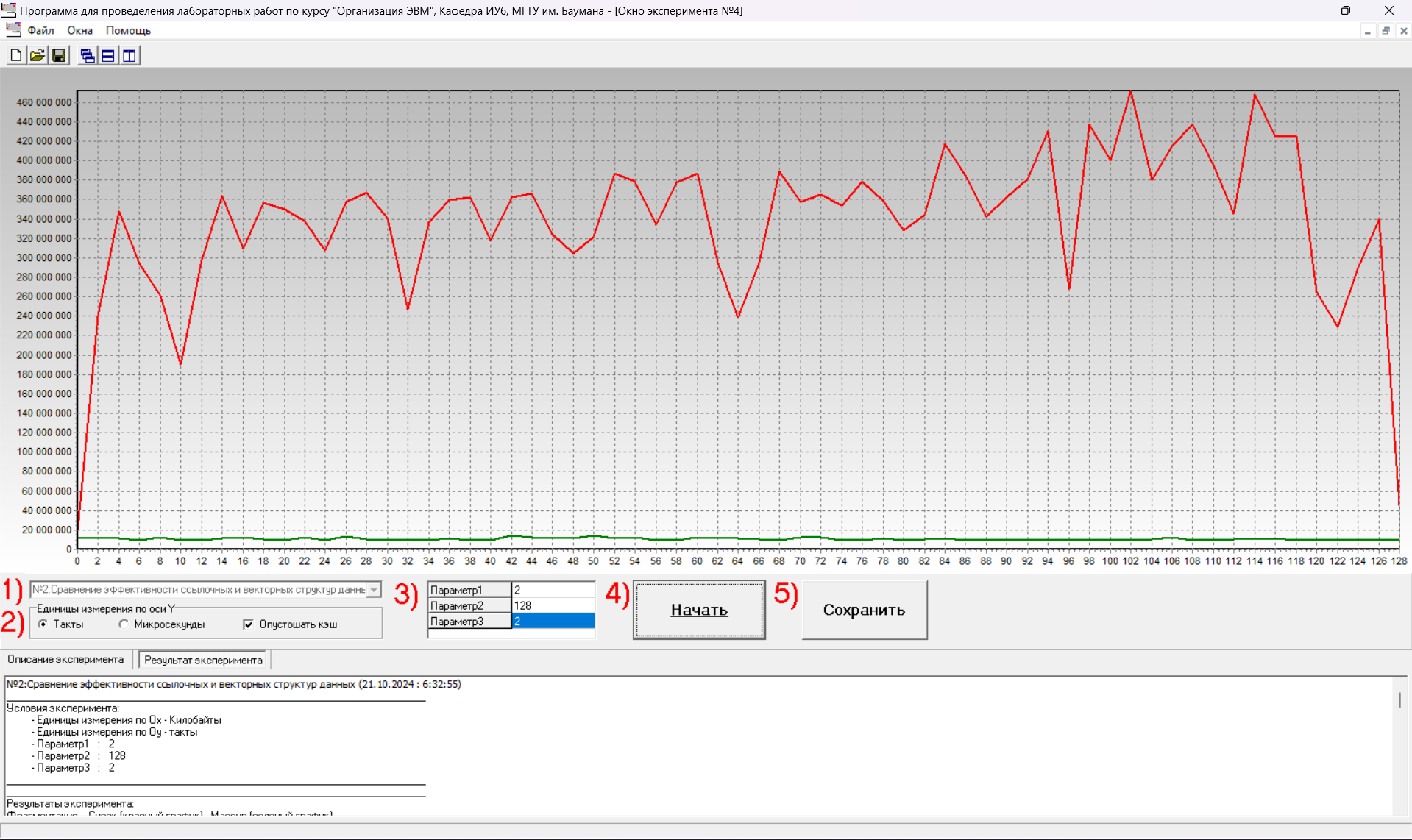


Рисунок 3 – Результат исследования эффективности ссылочных и векторных структур (параметры: 2-128-2)

Список обрабатывался в 31,548105 раз дольше.

Вывод: вычислительная машина предусматривает обработку векторных структур и хуже работает со ссылочными.

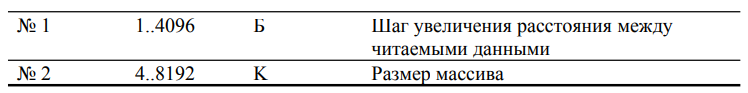
# Эксперимент 3: «Исследование эффективности программной предвыборки»

Цель эксперимента: выявление способов ускорения вычислений благодаря применению предвыборки данных.

Исходные данные: степень ассоциативности и размер TLB данных.

Для проведения эксперимента необходимо задать изменяемые параметры:





Результаты эксперимента: отношение времени последовательной обработки блока данных ко времени обработки блока с применением предвыборки; время и количество тактов первого обращения к странице данных.

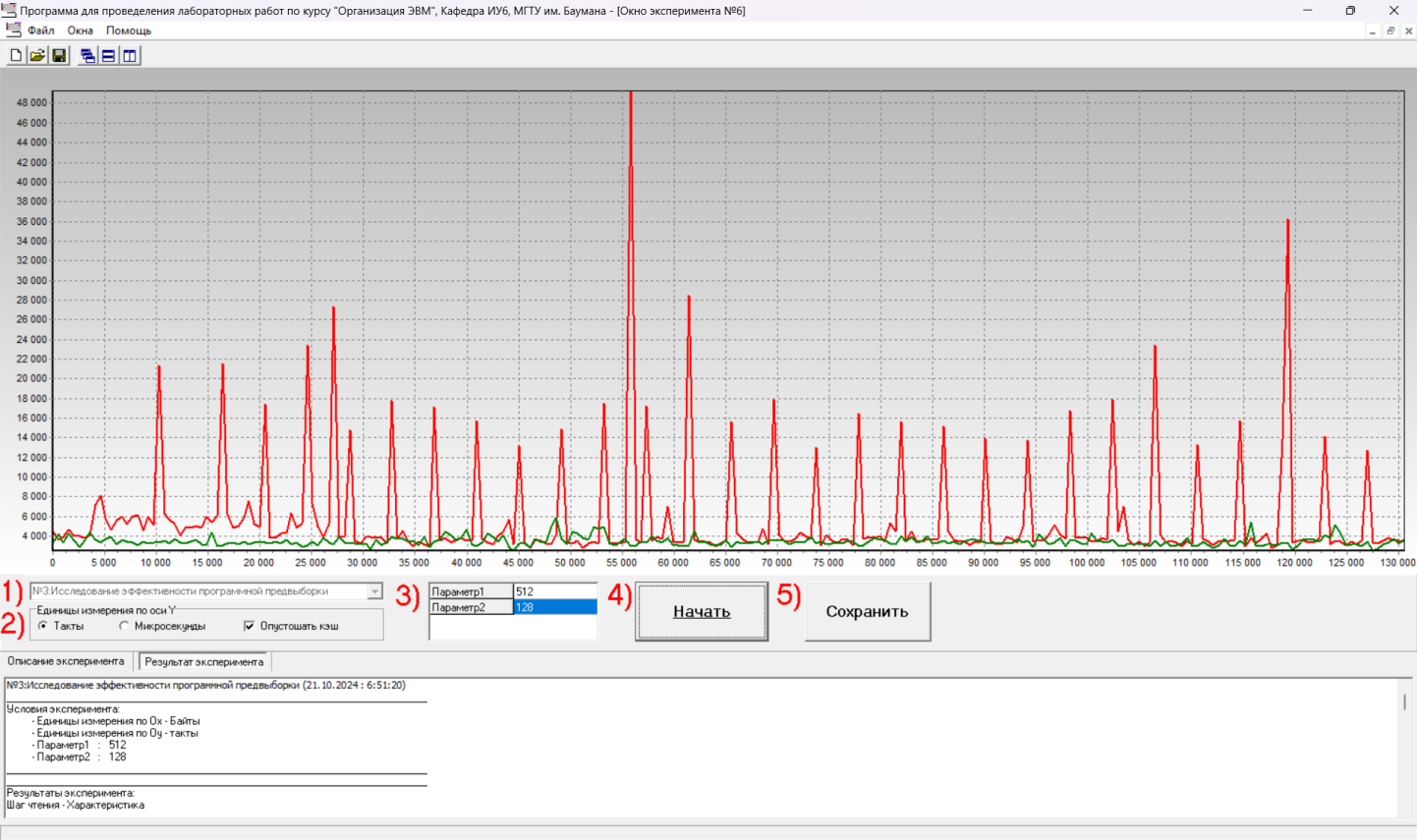


Рисунок 4 – Результат исследования эффективности предвыборки (параметры: 512-128)

Обработка без загрузки таблицы страниц в TLB производилась в 1,6890643 раз дольше.

Период выбросов ~4К.

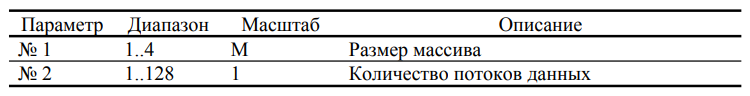
Вывод: получение дескриптора страницы это трудозатратная операция и кэширование данных перед обработкой ускорит последующую работу.

# Эксперимент 4: «Исследование способов эффективного чтения оперативной памяти»

Цель эксперимента: исследование возможности ускорения вычислений благодаря использованию структур данных, оптимизирующих механизм чтения оперативной памяти.

Исходные данные: Адресное расстояние между банками памяти, размер буфера чтения.

Для проведения эксперимента необходимо задать изменяемые параметры:



Результаты эксперимента: отношение времени обработки блока памяти неоптимизированной структуры ко времени обработки блока структуры, обеспечивающей эффективную загрузку и параллельную обработку данных.

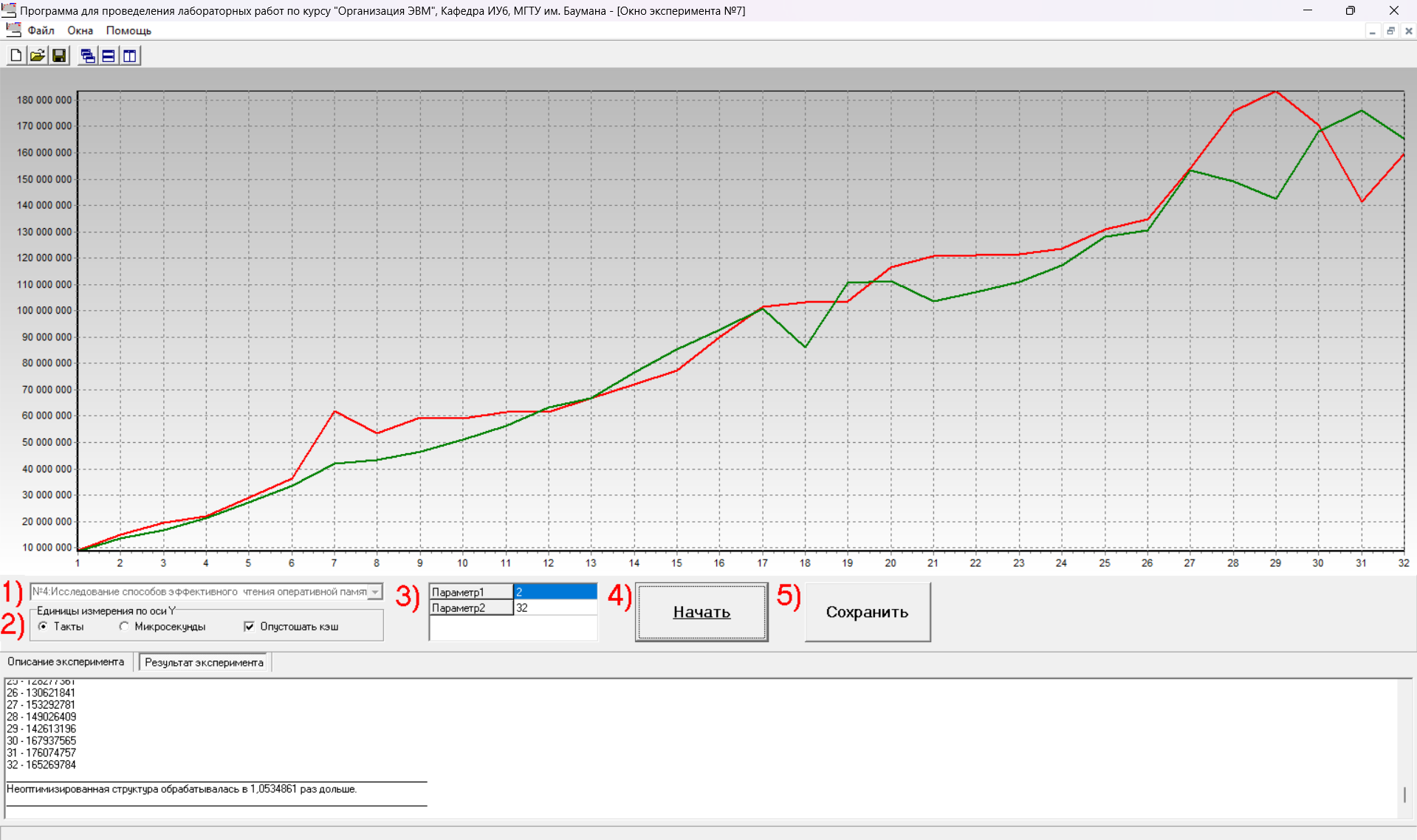


Рисунок 5 – Результат исследования оптимизирующих структур данных (параметры: 2-32)

Неоптимизированная структура обрабатывалась в 1,0534861 раз дольше.

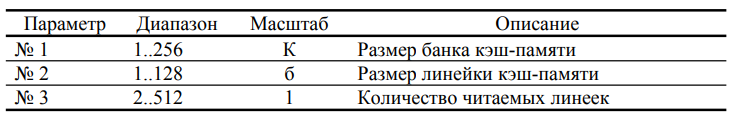
Вывод: программировать надо не объектно или процедурно, а пакетно, чтобы ускорить время работы. Если в дальшейшем нужно использовать весь блок памяти, а в данный момент только его часть, эффективнее прочитать сразу весь блок.

# Эксперимент 5: «Исследование конфликтов в кэш-памяти»

Цель эксперимента: исследование влияния конфликтов кэш-памяти на эффективность вычислений.

Исходные данные: Размер банка кэш-памяти данных первого и второго уровня, степень ассоциативности кэш-памяти первого и второго уровня, размер линейки кэш-памяти первого и второго уровня.

Для проведения эксперимента необходимо задать изменяемые параметры:



Результаты эксперимента: отношение времени обработки массива с конфликтами в кэш-памяти ко времени обработки массива без конфликтов.

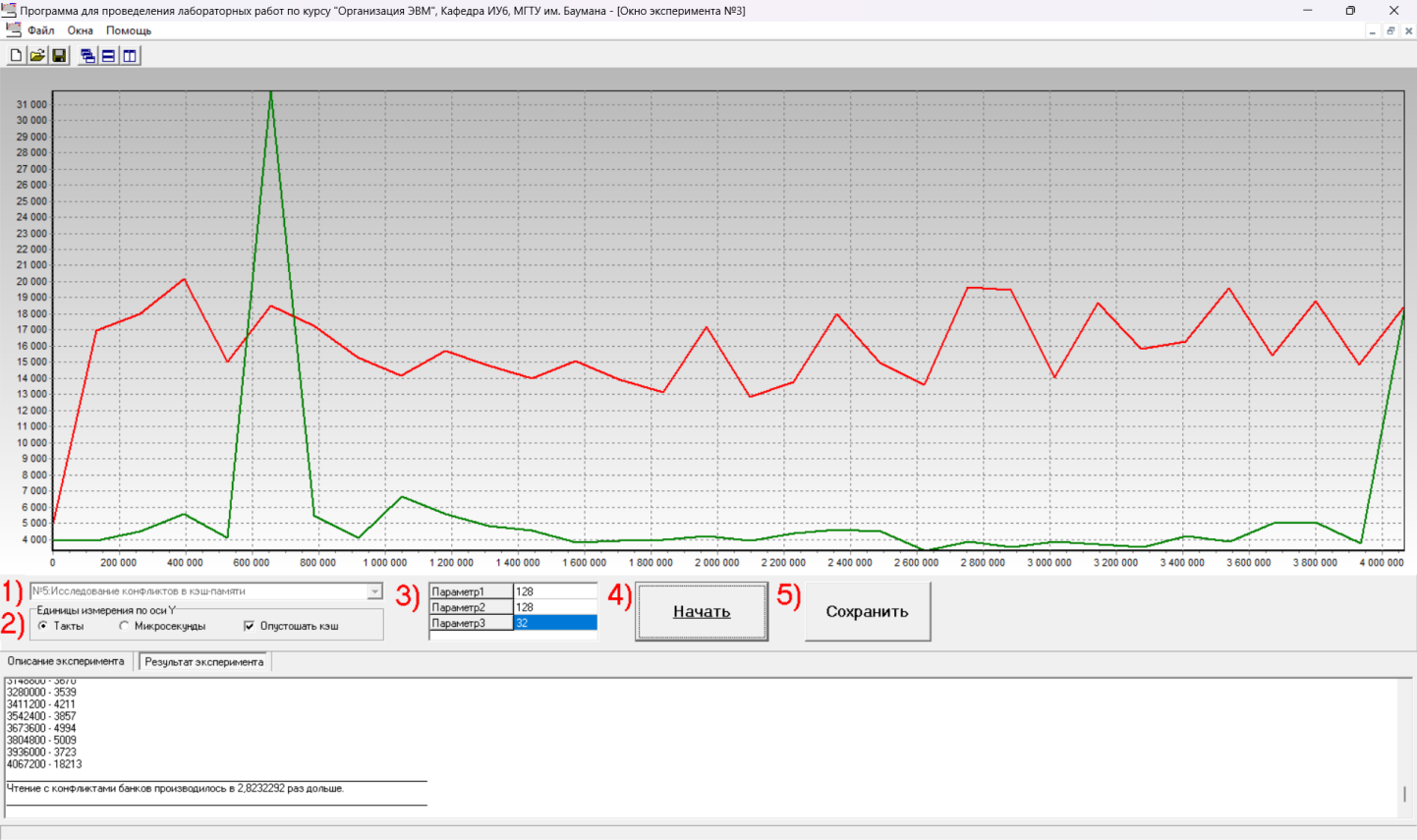


Рисунок 6 – Результат исследования конфликтов в кэш-памяти (параметры: 128-128-32)

Чтение с конфликтами банков производилось в 2,8232292 раз дольше.

Вывод: не стоит размещать и ходить по памяти с одинаковым, но длинным шагом, чтобы на уменьшать использованную область кэш-памяти.

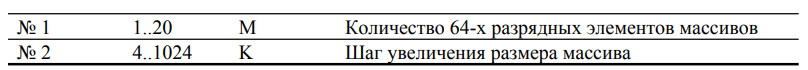
# Эксперимент 6: «Сравнение алгоритмов сортировки»

Цель эксперимента: исследование способов эффективного использования памяти и выявление наиболее эффективных алгоритмов сортировки, применимых в вычислительных системах.

Исходные данные: количество процессоров вычислительной системы, размер пакета, количество элементов в массиве, разрядность элементов массива.

Для проведения эксперимента необходимо задать изменяемые параметры:





Результаты эксперимента: отношение времени сортировки массива алгоритмом QuickSort ко времени сортировки алгоритмом Radix-Counting Sort и ко времени сортировки Radix-Counting Sort, оптимизированной под 8-процессорную вычислительную систему.

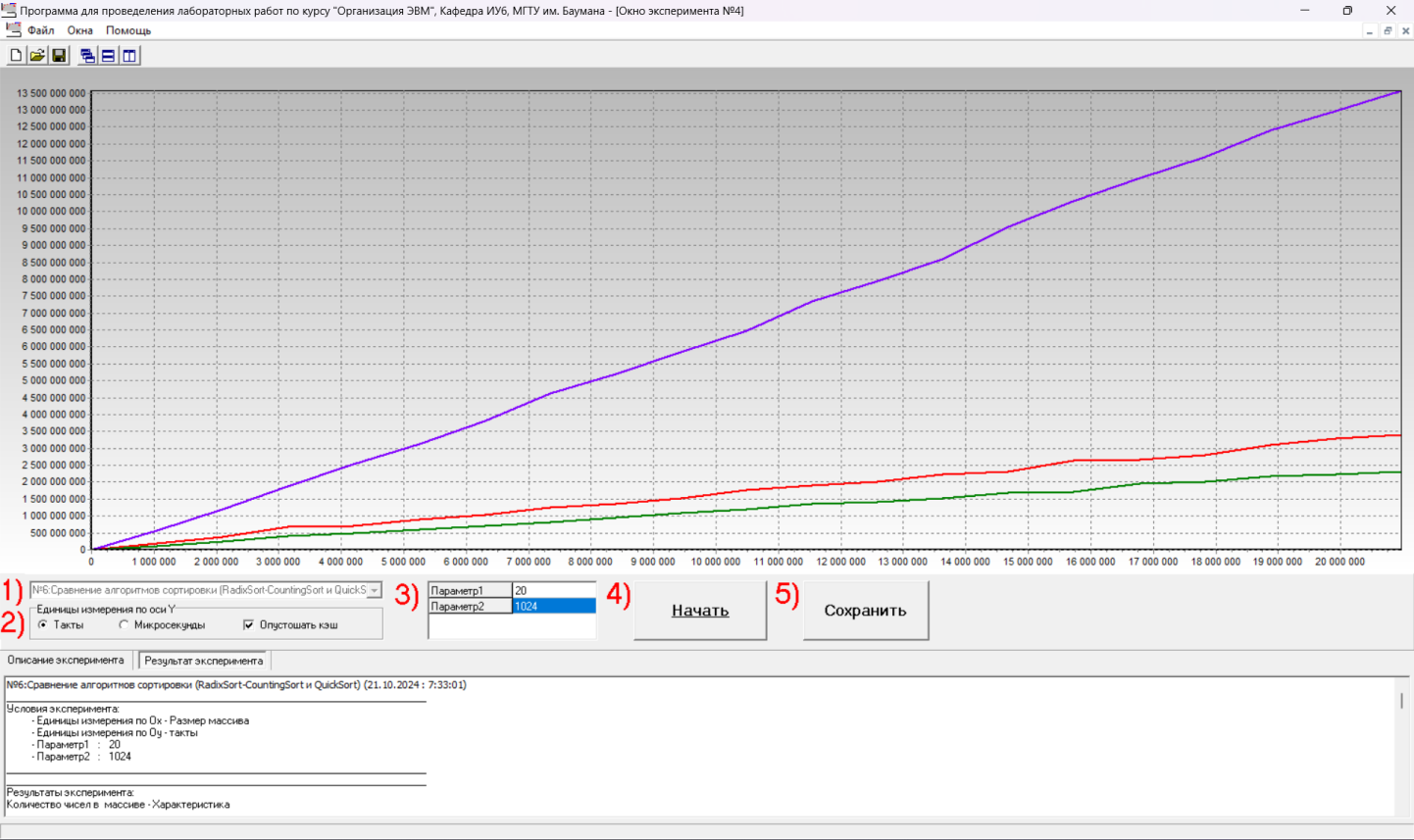


Рисунок 7 – Результат исследования алгоритмов сортировки (параметры: 20-1024)

QuickSort работал в 3,8834998 раз дольше Radix-Counting Sort.

QuickSort работал в 5,6224092 раз дольше Radix-Counting Sort, оптимизированного под 8-процессорную ЭВМ.

# Идентификация процессора

eax in eax ebx ecx edx

00000000 00000020 756e6547 6c65746e 49656e69

00000001 000b06a2 19400800 fffaf38b bfcbfbff

00000002 00feff01 000000f0 00000000 00000000

00000003 00000000 00000000 00000000 00000000

00000004 00000000 00000000 00000000 00000000

00000005 00000040 00000040 00000003 00002020

00000006 009f8ff3 00000002 00000409 00030003

00000007 00000000 00000000 00000000 00000000

00000008 00000000 00000000 00000000 00000000

00000009 00000000 00000000 00000000 00000000

0000000a 07300605 00000000 00000007 00008603

0000000b 00000000 00000000 00000038 00000019

0000000c 00000000 00000000 00000000 00000000

0000000d 00000000 00000000 00000000 00000000

0000000e 00000000 00000000 00000000 00000000

0000000f 00000000 00000000 00000000 00000000

00000010 00000000 00000000 00000000 00000000

00000011 00000000 00000000 00000000 00000000

00000012 00000000 00000000 00000000 00000000

00000013 00000000 00000000 00000000 00000000

00000014 00000000 00000000 00000000 00000000

00000015 00000002 000000a6 0249f000 00000000

00000016 00000000 00000000 00000000 00000000

00000017 00000000 00000000 00000000 00000000

00000018 00000000 00000000 00000000 00000000

00000019 00000000 00000000 00000000 00000000

0000001a 00000000 00000000 00000000 00000000

0000001b 00000000 00000000 00000000 00000000

0000001c 4000000b 00000007 00000007 00000000

0000001d 00000000 00000000 00000000 00000000

0000001e 00000000 00000000 00000000 00000000

0000001f 00000000 00000000 00000038 00000019

00000020 00000000 00000001 00000000 00000000

80000000 80000008 00000000 00000000 00000000

80000001 00000000 00000000 00000121 2c100800

80000002 68743331 6e654720 746e4920 52286c65

80000003 6f432029 54286572 6920294d 33312d35

80000004 48303035 00000000 00000000 00000000

80000005 00000000 00000000 00000000 00000000

80000006 00000000 00000000 05007040 00000000

80000007 00000000 00000000 00000000 00000100

80000008 00003027 00000000 00000000 00000000

Undocument layers

80860000 00000000 00000000 00000000 00000000

80860001 00000000 00000000 00000000 00000000

80860002 00000000 00000000 00000000 00000000

80860003 00000000 00000000 00000000 00000000

80860004 00000000 00000000 00000000 00000000

80860005 00000000 00000000 00000000 00000000

80860006 00000000 00000000 00000000 00000000

80860007 00000000 00000000 00000000 00000000

c0000000 00000000 00000000 00000000 00000000

c0000001 00000000 00000000 00000000 00000000

8ffffffe 00000000 00000000 00000000 00000000

8fffffff 00000000 00000000 00000000 00000000

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Vendor ID: "GenuineIntel"; CPUID level 32

Дополнительные функции Intel:

Верисия 000b06a2:

Type 0 - Original OEM

Family 6 - Pentium Pro

Model 10 -

Stepping 2

Reserved 44

Extended brand string: "13th Gen Intel(R) Core(TM) i5-13500H"

CLFLUSH instruction cache line size: 8

Initial APIC ID: 25

Hyper threading siblings: 64

Feature flags bfcbfbff:

0 FPU Присутствует Математический сопроцессор

1 VME Поддержка расширенных возможностей обработки прерываний в режиме виртуального i8086

2 DE Поддержка отладки

3 PSE Поддержка страниц размером 4 MB

4 TSC Счетчик меток реального времени

5 MSR Поддержка команд rdmsr и wrmsr

6 PAE Поддержка физического адреса более 32 бит

7 MCE Поддержка исключений 18 - об аппаратных ошибках

8 CX8 Поддержка инструкции cmpxchg8b

9 APIC Микропроцессор содержит программно доступный контроллер прерываний

11 SEP Поддержка инструкций быстрых системных вызовов sysenter и sysexit

12 MTRR Поддержка регистра mtrr\_cap (относится к MSR-регистрам)

13 PGE Поддержка глобальных страниц

14 MCA Поддержка архитектуры машинного контроля

15 CMOV Поддержка инструкций условной пересылки cmov, fcmovcc, fcomi

16 PAT Процессор поддерживает таблицу атрибутов страницы

17 PSE-36 Процессор поддерживает 4 MB страницы, которые способны адресовать физическую память до 64 GB

19 CLFLSH Поддержка инструкции CLFLUSH

22 ACPI Управление охлаждением процессора с помощью пустых циклов в зависимости от температуры

23 MMX Поддержка MMX

24 FXSR Поддержка инструкций FXSAVE и FXRSTOR

25 SSE Поддержка SSE

26 SSE2 Поддержка SSE2

27 SS Управление конфликтующими типами памяти

28 HTT Поддержка Hyper-Threading

29 TM Поддержка автоматического мониторинга температуры

31 SBF Сигнал Останова при FERR

TLB and cache info:

ff: unknown TLB/cache descriptor

fe: unknown TLB/cache descriptor

f0: unknown TLB/cache descriptor

Processor serial: 000B-06A2-BFCB-FBFF-FFFA-F38B

Контрольные вопросы

1. Назовите причины расслоения оперативной памяти.

* Повышение скорости доступа к памяти за счет одновременного обращения к разным банкам памяти.
* Уменьшение задержек при последовательных операциях чтения/записи.
* Оптимизация работы с памятью при параллельной обработке данных.

1. Как в современных процессорах реализована аппаратная предвыборка.

Аппаратная предвыборка в современных процессорах реализована с использованием специализированных механизмов предсказания обращения к памяти, которые анализируют паттерны доступа к данным. Эти механизмы заранее загружают данные в кэш из основной памяти, основываясь на прошлых обращениях, что снижает задержки при доступе к данным.

1. Какая информация храниться в TLB.

TLB хранит отображение виртуальных адресов в физические. Для каждого виртуального адреса сохраняется соответствующий физический адрес, а также атрибуты доступа (например, права на чтение/запись, наличие данных в кэше, и т.д.).

1. Какой тип ассоциативной памяти используется в кэш-памяти второго уровня современных ЭВМ и почему.

В кэш-памяти второго уровня чаще всего используется частично ассоциативная (set-associative) память. Это компромисс между полностью ассоциативной и прямой памятью, позволяющий снизить вероятность конфликтов при одновременном доступе к одному и тому же набору данных, сохраняя при этом высокую производительность.

1. Приведите пример программной предвыборки

Программная предвыборка может быть реализована с помощью инструкций, которые явно загружают данные в кэш до их фактического использования. Например, в языке C это может быть реализовано с использованием инструкции \_\_builtin\_prefetch(), которая заранее загружает данные из памяти в кэш, чтобы минимизировать задержки при дальнейшем доступе к ним.