



Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное 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 Программное обеспечение ЭВМ и информационные технологии (ИУ7)

О т ч е т

по лабораторной работе № 3

Название: Организация памяти суперскалярных ЭВМ

Дисциплина: Архитектура ЭВМ

Студент гр. ИУ7-53Б

(Подпись, дата)

Н.В.Куликов

(И.О. Фамилия)

Преподаватель

(Подпись, дата)

С.В. Ибрагимов

(И.О. Фамилия)

2025 год

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.

1. Исследования расслоения динамической памяти

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

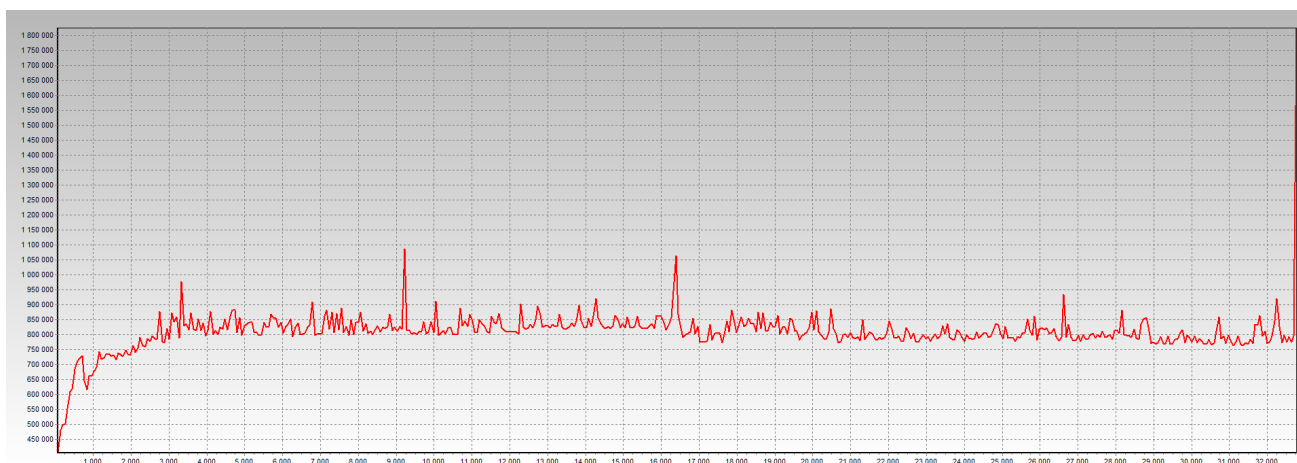
Исходные данные: размер линейки кэш-памяти верхнего уровня; объем физической памяти.

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

Условия эксперимента:

- Единицы измерения по Ох - Байты
- Единицы измерения по Оу - такты
- Параметр1: 32 (Максимальное расстояние между читаемыми блоками)
- Параметр2: 64 (Шаг увеличения расстояния между читаемыми 4-х байтовыми ячейками)
- Параметр3: 1 (Размер массива)

График:



Красный график показывает время (или количество тактов) работы алгоритма. Ось абсцисс отражает шаг приращения адреса читаемых данных. Ось ординат отображает время в микросекундах или количество тактов (в зависимости от заданного параметра «Единицы измерения по оси Y»).

По графику можно определить следующие параметры:

1. Минимальный шаг чтения динамической памяти, при котором происходит постоянное обращение к одному и тому же банку. При наличии нескольких банков памяти данный параметр соответствует первому локальному экстремуму полученной функции (точка T1). Отсутствие характерного ступенеобразного графика говорит об одном независимом банке динамической памяти. По полученному значению шага T1 можно определить количество банков памяти: $B = T1/P$, где P - объем данных, являющийся минимальной порцией обмена кэш-памяти верхнего уровня с оперативной памятью и соответствует размеру линейки кэш-памяти верхнего уровня.
2. При достижении глобального экстремума, после которого рост локальных экстремумов не происходит, определяется характерная точка T2. Соответствующий данной точке шаг чтения является наихудшим при обращении к динамической памяти, т.к. приводит к постоянному закрытию и открытию страниц динамической памяти. Таким образом шаг T2 соответствует расстоянию (в байтах) между началом двух последовательных страниц одного банка. Зная количество банков, определяем размер одной страницы: $PC = T2/B$.
3. Зная параметры PC и B, а также полный объем памяти O определяем количество страниц физической оперативной памяти: $C = O/(PC*B*P)$.

2. Сравнение эффективности ссылочных и векторных структур

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

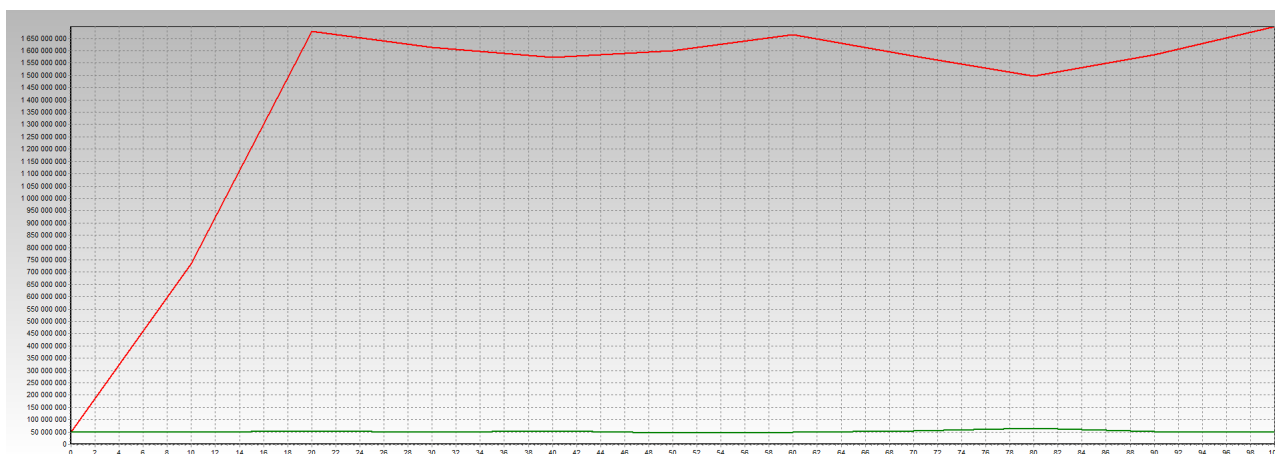
Результаты эксперимента: отношение времени работы алгоритма, использующего зависимые данные, ко времени обработки аналогичного алгоритма обработки независимых данных.

Суть эксперимента. Для сравнения эффективности векторных и списковых структур в эксперименте применяется профилировка кода двух алгоритмов поиска минимального значения. Первый алгоритм использует для хранения данных список, в то время как во втором применяется массив. Очевидно, что время работы алгоритма поиска минимального значения в списке зависит от его фрагментации, т.е. от среднего расстояния между элементами списка.

Условия эксперимента:

- Единицы измерения по Ох - Килобайты
- Единицы измерения по Оу - такты
- Параметр1: 5 (Количество элементов в списке)
- Параметр2: 100 (Максимальная фрагментации списка)
- Параметр3: 10 (Шаг увеличения фрагментации)

График:



Красный график показывает время (или количество тактов) работы алгоритма, использующего список. Зеленый график показывает время (или количество тактов) работы алгоритма, использующего массив. Ось абсцисс отражает фрагментацию списка. Ось ординат отображает время в микросекундах или количество тактов (в зависимости от заданного параметра «Единицы измерения по оси Y»). Список обрабатывался в 26,858059 раз дольше.

3. Исследование эффективности программной предвыборки

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

Исходные данные: степень ассоциативности и размер TLB данных.

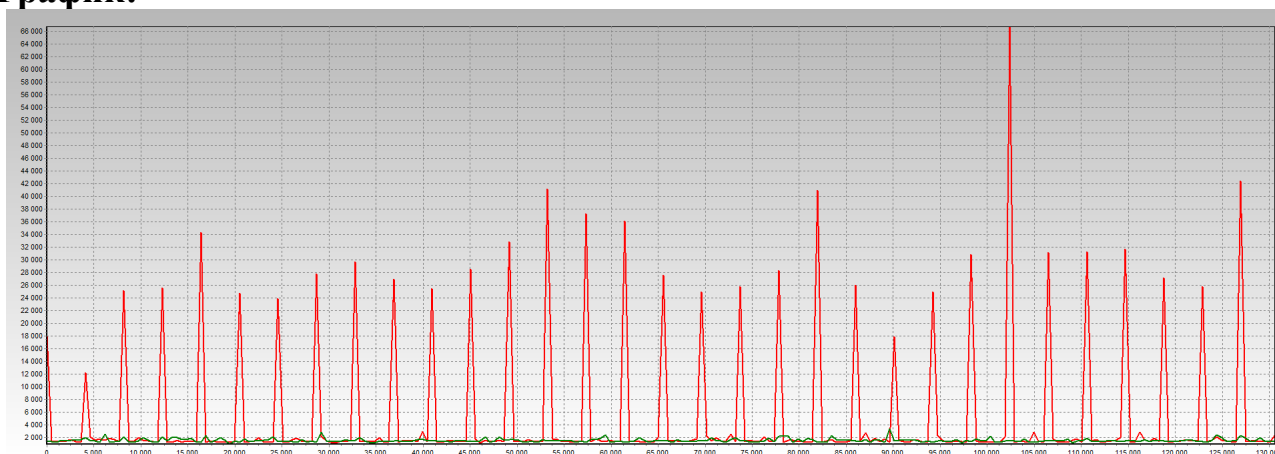
Результаты эксперимента: отношение времени последовательной обработки блока данных ко времени обработки блока с применением предвыборки; время и количество тактов первого обращения к странице данных.

Суть эксперимента. Эксперимент основан на замере времени двух вариантов подпрограмм последовательного чтения страниц оперативной памяти. В первом варианте выполняется последовательное чтение без дополнительной оптимизации, что приводит к дополнительным двойным обращениям. Во втором варианте перед циклом чтения страниц используется дополнительный цикл предвыборки, обеспечивающий своевременную загрузку информации в TLB данных.

Условия эксперимента:

- Единицы измерения по Ох - Байты
- Единицы измерения по Оу - такты
- Параметр1: 512 (Шаг увеличения расстояния между читаемыми данными)
- Параметр2: 128 (Размер массива)

График:



Красный график показывает время (или количество тактов) работы алгоритма без предвыборки. Зеленый график показывает время (или количество тактов) работы алгоритма с использованием предвыборки. Ось абсцисс отражает смещение читаемых данных от начала блока. Ось ординат отображает

время в микросекундах или количество тактов (в зависимости от заданного параметра «Единицы измерения по оси Y»). Обработка без загрузки таблицы страниц в TLB производилась в 3,3356413 раз дольше.

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 блока переупорядочивания запросов к памяти.

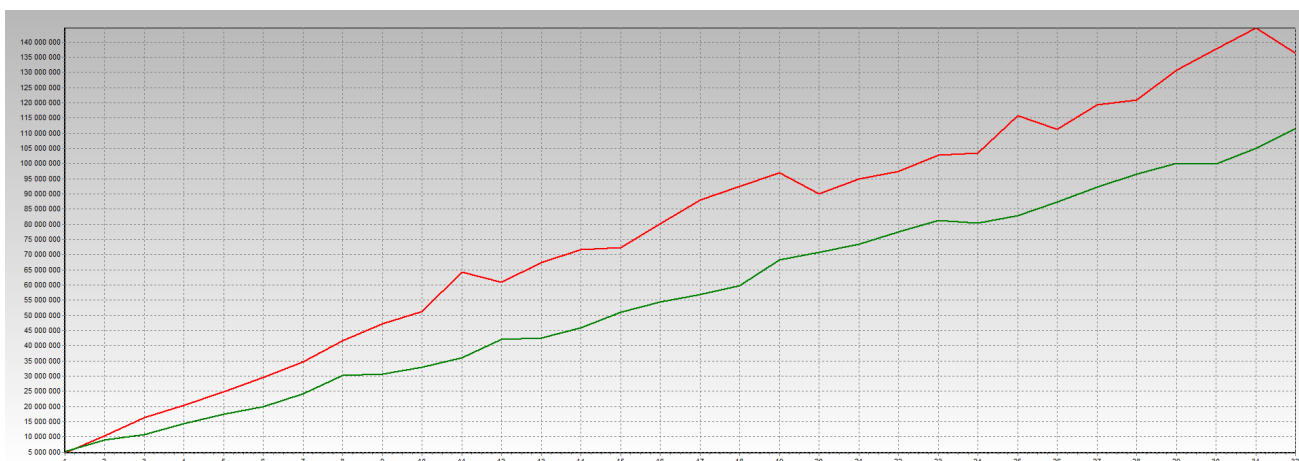
Результаты эксперимента: отношение времени обработки блока памяти неоптимизированной структуры ко времени обработки блока структуры, обеспечивающей эффективную загрузку и параллельную обработку данных.

Суть эксперимента. Для сравнения производительности алгоритмов, использующих оптимизированные и неоптимизированные структуры данных используется профилировка кода двух подпрограмм, каждая из которых должна выполнить обработку нескольких блоков оперативной памяти. В алгоритмах обрабатываются двойные слова данных (4 байта), что существенно меньше размера пакета (32 - 128 байт). Неоптимизированный вариант структуры данных представляет собой несколько массивов в оперативной памяти, в то время как оптимизированная структура состоит из чередующихся данных каждого массива.

Условия эксперимента:

- Единицы измерения по Ох - Количество параллельных потоков
- Единицы измерения по Оу - такты
- Параметр1: 1 (Размер массива)
- Параметр2: 32 (Количество потоков данных)

График:



Красный график показывает время (или количество тактов) работы алгоритма, использующего неоптимизированную структуру. Зеленый график показывает время (или количество тактов) работы алгоритма с использованием оптимизированной структуры. Ось абсцисс отражает количество одновременно обрабатываемых массивов. Ось ординат отображает время в микросекундах или количество тактов (в зависимости от заданного параметра «Единицы измерения по оси Y»). Неоптимизированная структура обрабатывалась в 1,3699369 раз дольше.

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.

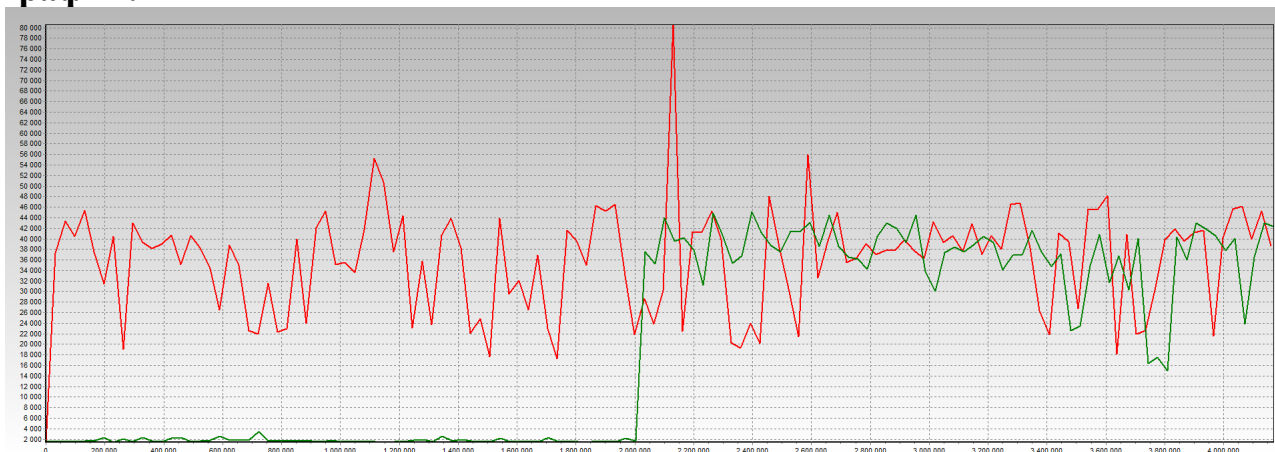
Результаты эксперимента: отношение времени обработки массива с конфликтами в кэш-памяти ко времени обработки массива без конфликтов.

Суть эксперимента. Для определения степени влияния конфликтов в кэш-памяти на эффективность вычислений используется профилировка двух процедур чтения и обработки данных. Первая процедура построена таким образом, что чтение данных выполняется с шагом, кратным размеру банка. Это порождает постоянные конфликты в кэш-памяти. Вторая процедура оптимизирует размещение данных в кэш с помощью задания смещения востребованных данных на некоторый шаг, достаточный для выбора другого набора. Этот шаг соответствует размеру линейки.

Условия эксперимента:

- Единицы измерения по Ох - Смещение от начала блока
- Единицы измерения по Оу - такты
- Параметр1: 32 (Размер банка кэш-памяти)
- Параметр2: 64 (Размер линейки кэш-памяти)
- Параметр3: 128 (Количество читаемых линеек)

График:



Красный график показывает время (или количество тактов) работы процедуры, читающей данные с конфликтами в кэш-памяти. Зеленый график показывает время (или количество тактов) работы процедуры, не вызывающей конфликтов в кэш-памяти. Ось абсцисс отражает смещение читаемой ячейки от начала блока данных. Ось ординат отображает время в микросекундах или количество тактов (в зависимости от заданного параметра «Единицы измерения по оси Y»). Чтение с конфликтами банков производилось в 1,7990065 раз дольше.

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.

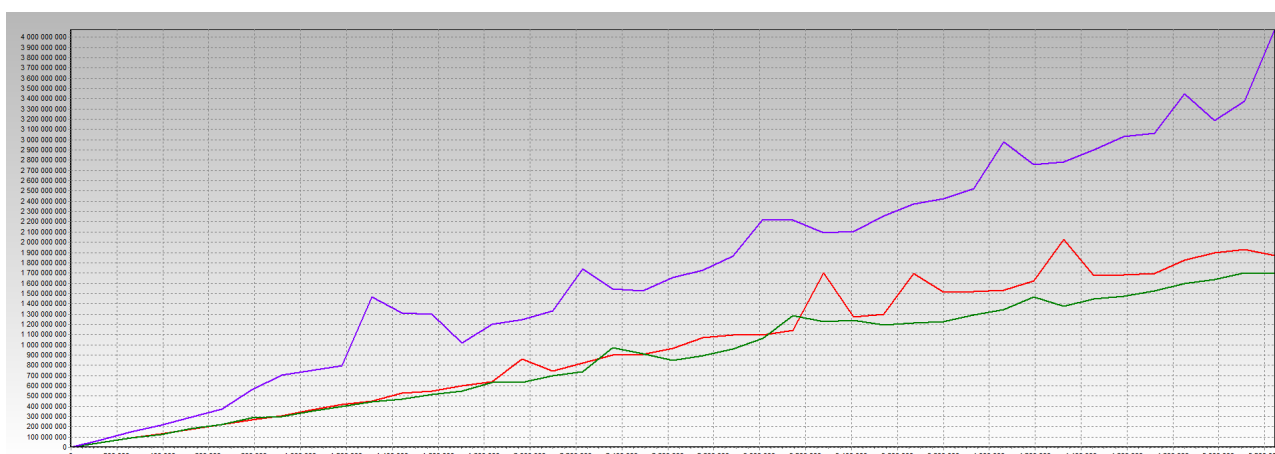
Результаты эксперимента: отношение времени сортировки массива алгоритмом QuickSort ко времени сортировки алгоритмом Radix-Counting Sort .

Суть эксперимента. Эксперимент основан на замере времени трех вариантов алгоритмов сортировки (Quick Sort, Radix Sort, Оптимизированный Radix Sort).

Условия эксперимента:

- Единицы измерения по Ох - Размер массива
- Единицы измерения по Оу - такты
- Параметр1: 5 (Количество 64-х разрядных элементов массивов)
- Параметр2: 128 (Шаг увеличения размера массива)

График:



Фиолетовый график (верхний) показывает время или количество тактов работы алгоритма QuickSort. Красный график (средний) показывает время или

количество тактов работы неоптимизированного алгоритма Radix-Counting. Зеленый график (нижний) показывает время или количество тактов работы оптимизированного под 8-процессорную вычислительную систему алгоритма Radix-Counting. Ось абсцисс отражает количество 64-х разрядных элементов сортируемых массивов. Ось ординат отображает время в микросекундах или количество тактов (в зависимости от заданного параметра «Единицы измерения по оси Y»). QuickSort работал в 1,7662467 раз дольше Radix-Counting Sort. QuickSort работал в 2,0042042 раз дольше Radix-Counting Sort, оптимизированного под 8-процессорную ЭВМ.

7. Идентификация процессора, использованного при проведении экспериментов

Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz 2.59 GHz