Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики
Факультет информационных технологий и программирования
Кафедра «Компьютерные Технологии»

Шаламов В.В.

Отчет по курсовой работе
«External Memory: K-Way merge sort»

Санкт-Петербург
2017

1. Введение

1.1. Постановка задачи

Реализовать алгоритм сортировки K-Way Merge Sort в external memory.

1.2. Пояснение сложности задачи

В external memory можно читать и писать только блоками фиксированного размера, поэтому обычная сортировка работала бы на external memory крайне неэффективно: для каждого обращения к одному любому элементу читался/записывался бы сразу целый блок. Невозможно загрузить в оперативную память сразу все данные и отсортировать, даже если читать из с диска и писать на диск долгами фиксированного размера.

2. Реализация

Для реализации поставленной задачи был выбран язык программирования Java. Использовалась последняя на момент написания кода сборка jdk 8u112.

Задача была разбита на 3 части:

1. Разработка класса **ExternalSort**, который предоставляет единственный метод **sort**, позволяющий отсортировать данные в указанном исходном файле и записать сохраненный результат в указанный выходной файл. В процессе сортировки может создаваться множество временных файлов, количество которых ограничено заранее заданной константой **TEMP_FILES_LIMIT**, которая по умолчанию равна **1024**.
2. Разработка класса **BinaryFileOfIntsBuffer**, который инкапсулирует логику поблокового чтения и десериализации сортируемых объектов (в данном случае - 4-байтовых целых чисел).
3. Разработка методов для генерации файлов со случайными данными и для тестирования работы **ExternaSort**.

2.1. ExternalSort

Класс **ExternalSort** конструируется для сортировки данных в файле. Использует финкциональность, предоставляемую классом **BinaryFileOfIntsBuffer** для чтения файлов с диска. Предоставляет метод:

- *sort(inputFile, outputFile)* — сортирует данные в inputFile, записывает данные в outputFile. ;

Кроме того, класс **ExternalSort** в процессе работы создает множество временных файлов в директории для временных файлов текущего пользователя, а затем автоматически удаляет их.

Сортировка проводится с помощью алгоритма K-Way Merge Sort. Рассмотрим его подробнее.

2.2. BinaryFileOfIntsBuffer

Данный класс реализует функциональность чтения нужных объектов из переданного файла по одному, либо порцией не превышающей указанный размер (может быть меньше, если файл закончился).

Список поддерживаемых методов:

- *BinaryFileOfIntsBuffer(file: File)* – публично доступный конструктор. Создает класс, считывает первый блок из файла, инициализирует, кеширует первый элемент;
- *isEmpty()* – возвращает **true**, если в файле больше нет доступных для чтения объектов;
- *pop()* – возвращает закешированный элемент, инвалидирует кэш. Если доступны еще элементы, то кэширует следующий. В противном случае, переходит в терминальное состояние, и с этого момента метод isEmpty() всегда возвращает true, а методы peek(), pop() и popBatch() при вызове выбрасывают исключение;
- *popBatch(**amount : int**)* – повторяет операцию pop() **amount** раз, либо пока не закончатся элементы в файле. Возвращает полученные элементы;
- *peek()* – возвращает закодированный элемент;
- *close()* – закрывает открытый на чтение файл. Нужен для освобождения ресурсов;
- *delete()* – закрывает открытый на чтение файл и удаляет его. Нужен для освобождения ресурсов;

2.3. Тесты

Для тестирования **ExternalSort** были разработаны следующие тесты:

- Небольшие тесты, проверяющие работоспособность и корректность основных операций и методов.
- Тест, производящий *external merge sort* набора случайных чисел. В данном случае использовались бинарные файлы различного размера, в который каждые последовательные 4 байта интерпретируются как одно 4-байтовое число типа **int**.

При запуске программы указываются опции, определяющие её поведение:

- -g <file.bin> <intNum> – программа создает указанный файл и заполняет его указанным количеством 4-байтовых чисел.
- -c <file.bin> – программа проверяет, что 4-байтовые челый числа в указанном файле расположены в порядке неубывания
- -s <file.in.bin> <file.out.bin> – программа сортирует содержимое файла **file.in.bin** и записать содержимое в файл **file.bin.out**, интерпретируя каждые 4 байта как целое число.
- -t – запуск тестов.

3. Результаты

Работа алгоритма тестировалась со следующими вариациями. Было сгенерировано **3** файла, состоящие из **1 024 000**, **10 240 000** и **102 400 000** 4-байтовых целых чисел, размером, соответственно (**4000 Kb**, **40 000 Kb** и **400 000 Kb**) соответственно. В результате тестирования был построен график зависимости потраченного времени на сортировку входного файла от размера буфера, для каждого из трех наборов входных данных.

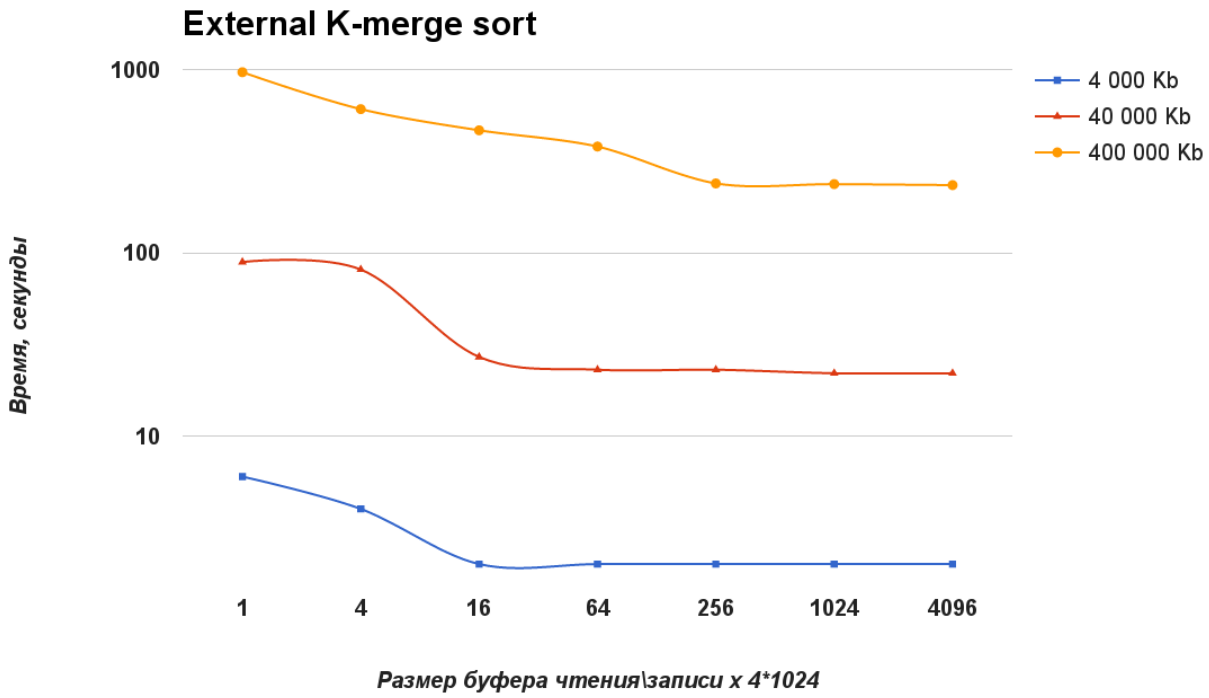

Рисунок 1. Зависимость времени работы алгоритма сортировки от размера буфера обмена для файлов разного размера.

4. Источники

1. Исходный код: https://github.com/sslavian812/external-merge-sort
2. Википедия: https://en.wikipedia.org/wiki/External_sorting
3. Java: https://www.oracle.com/java/index.html