Лабораторная работа №4

Методы внешней сортировки

Прямое слияние

Алгоритм сортировки прямым слиянием основывается на алгоритме простого слияния. Предположим, что имеется последовательный файл A, состоящий из записей a_1, a_2, …, a_n (для простоты предположим, что n представляет собой степень числа 2). Будем считать, что каждая запись состоит ровно из одного элемента, представляющего собой ключ сортировки. Для сортировки используются два вспомогательных файла B и C (размер каждого из них будет $n/2$).

Сортировка состоит из шагов, в каждом из которых выполняется распределение состояния файла A в файлы B и C, а затем слияние файлов B и C в A. На первом шаге для распределения последовательно читается файл A, и записи a_1, a_3, …, a_{n-1} пишутся в B, а записи a_2, a_4 , …, a_n – в C (начальное распределение). Начальное слияние производится над парами (a_1, a_2), (a_3, a_4), …, (a_{n-1}, a_n), и результат записывается в файл A. На втором шаге снова последовательно читается файл A, и в B записываются последовательные пары с нечетными номерами, а в файл C – с чётными. При слиянии образуются и пишутся в файл A упорядоченные четверки записей. И так далее. Перед выполнением последнего шага файл A будет содержать две упорядоченные подпоследовательности размером $n/2$ каждая. При распределении первая из них попадёт в файл B, а вторая – в файл C. После слияния файл A будет содержать полностью упорядоченную последовательность записей. На рисунке 1 показан пример внешней сортировки прямым слиянием.

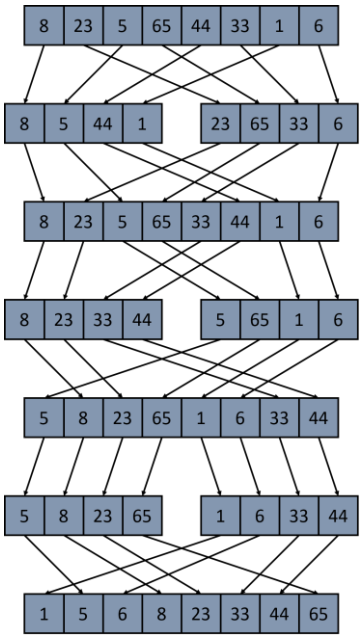

Рисунок 1 - Пример сортировки прямым слиянием

Для выполнения внешней сортировки методом прямого слияния в основной памяти требуется расположить две переменные – для размещения очередных записей из файлов B и C. Файлы A, B и C будут $O(\log(n))$ раз прочитаны и столько же раз записаны.

Естественное слияние

При использовании метода прямого слияния не принимается во внимание то, что исходный файл может быть частично отсортированным. Серией называется подпоследовательность записей a_i, a_{i+1}, …, a_j такая, что $a_k \le a_{k+1}$ для всех $i \le k < j$, $a_i < a_{i-1}$ и $a_j > a_{j+1}$. Метод естественного слияния основывается на распознавании серий при распределении и их использовании при последующем слиянии.

Как и в случае прямого слияния, сортировка выполняется за несколько шагов, в каждом из которых сначала выполняется распределение файла A по файлам B и C, а потом слияние B и C в файл A. При распределении распознается первая серия записей и переписывается в файл B, вторая – в файл C и т.д. При слиянии первая серия записей файла B сливается с первой серией файла C, вторая серия B со второй серией C и т.д. Если просмотр одного файла заканчивается раньше, чем просмотр другого (по причине разного числа серий), то остаток недопросмотренного файла целиком копируется в конец файла A. Процесс завершается, когда в файле A остается только одна серия. Пример сортировки файла показан на рисунке 1.

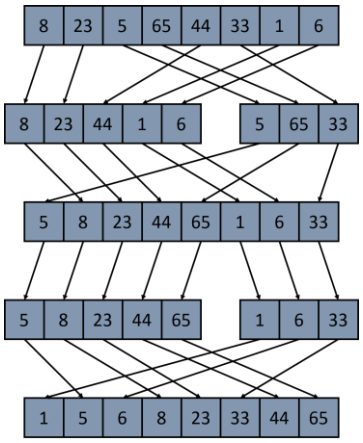

Рисунок 1 – Сортировка естественным слиянием

Очевидно, что число чтений/перезаписей файлов при использовании этого метода будет не хуже, чем при применении метода прямого слияния, а в среднем – лучше. С другой стороны, увеличивается число сравнений за счет тех, которые требуются для распознавания концов серий. Кроме того, поскольку длина серий может быть произвольной, то максимальный размер файлов B и C может быть близок к размеру файла A.

Сбалансированное многопутевое слияние

В основе метода внешней сортировки сбалансированным многопутевым слиянием является распределение серий исходного файла по m вспомогательным файлам $B1$, $B2$, …, Bm и их слияние в m вспомогательных файлов C_1, C_2, …, C_m. На следующем шаге производится слияние файлов $C1$, $C2$, …, Cm в файлы $B1$, $B2$, …, Bm и так далее, пока в $B1$ или $C1$ не образуется одна серия.

Многопутевое слияние является естественным развитием идеи обычного (двухпутевого) слияния. Пример трёхпутевого слияния показан на рисунке 2.

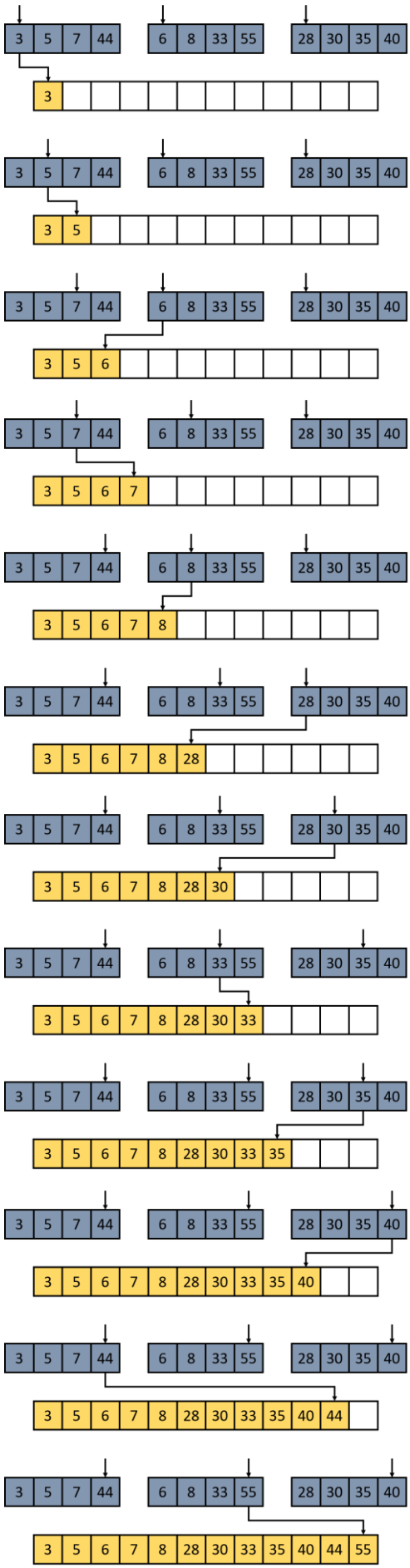

Рисунок 2 – Пример трёхпутевого слияния

На рисунке 3 показан простой пример применения сортировки многопутевым слиянием.

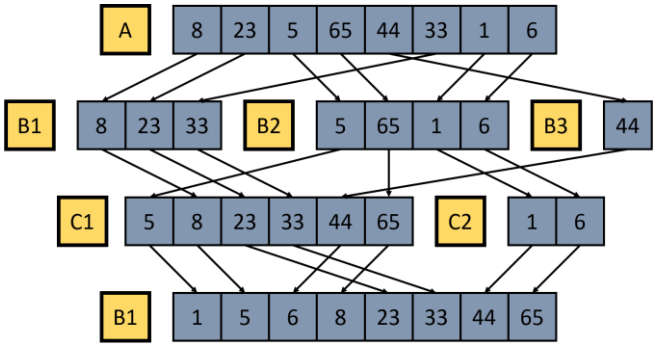

Рисунок 3 – Пример сортировки многопутевым слиянием

Он, конечно, слишком тривиален, чтобы продемонстрировать несколько шагов выполнения алгоритма, однако достаточен в качестве иллюстрации общей идеи метода. Заметим, что, как показывает этот пример, по мере увеличения длины серий вспомогательные файлы с большими номерами (начиная с номера n) перестают использоваться, поскольку им "не достается" ни одной серии. Преимуществом сортировки сбалансированным многопутевым слиянием является то, что число проходов алгоритма оценивается как $O\big(log(n)\big)$ (n – число записей в исходном файле), где логарифм берется по основанию n. Порядок числа копирований записей – $O\big(log(n)\big)$. Конечно, число сравнений не будет меньше, чем при применении метода простого слияния.

Многофазная сортировка

При использовании рассмотренного выше метода сбалансированной многопутевой внешней сортировки на каждом шаге примерно половина вспомогательных файлов используется для ввода данных и примерно столько же для вывода сливаемых серий. Идея многофазной сортировки состоит в том, что из имеющихся m вспомогательных файлов $(m-1)$ файл служит для ввода сливаемых последовательностей, а один – для вывода образуемых серий. Как только один из файлов ввода становится пустым, его начинают использовать для вывода серий, получаемых при слиянии серий нового набора $(m-1)$ файлов. Таким образом, имеется первый шаг, при котором серии исходного файла распределяются по $m-1$ вспомогательному файлу, а затем выполняется многопутевое слияние серий из $(m-1)$ файла, пока в одном из них не образуется одна серия.

Очевидно, что при произвольном начальном распределении серий по вспомогательным файлам алгоритм может не сойтись, поскольку в единственном непустом файле будет существовать более чем одна серия. Предположим, например, что используется три файла $B1$, $B2$ и $B3$, и при начальном распределении в файл $B1$ помещены 10 серий, а в файл $B2$ – 6. При слиянии $B1$ и $B2$ к моменту, когда мы дойдем до конца $B2$, в $B1$ останутся 4 серии, а в $B3$ попадут 6 серий. Продолжится слияние $B1$ и $B3$, и при завершении просмотра $B1$ в $B2$ будут содержаться 4 серии, а в $B3$ останутся 2 серии. После слияния $B2$ и $B3$ в каждом из файлов $B1$ и $B2$ будет содержаться по 2 серии, которые будут слиты и образуют 2 серии в $B3$ при том, что $B1$ и $B2$ – пусты. Тем самым, алгоритм не сошелся (таблица 2).

Таблица 2 – Пример начального распределения серий, при котором трехфазная внешняя сортировка не приводит к нужному результату

Число серий в файле B1	**Число серий в файле B2**	**Число серий в файле B3**
10	6	0
4	0	6
0	4	2
2	2	0
0	0	2

Попробуем понять, каким должно быть начальное распределение серий, чтобы алгоритм трехфазной сортировки благополучно завершал работу и выполнялся максимально эффективно. Для этого рассмотрим работу алгоритма в обратном порядке, начиная от желательного конечного состояния вспомогательных файлов. Нас устраивает любая комбинация конечного числа серий в файлах $B1$, $B2$ и $B3$ из $(1, 0, 0)$, $(0, 1, 0)$ и $(0, 0, 1)$. Для определенности выберем первую комбинацию. Для того, чтобы она сложилась, необходимо, чтобы на непосредственно предыдущем этапе слияний существовало распределение серий $(0, 1, 1)$. Чтобы получить такое распределение, необходимо, чтобы на непосредственно предыдущем этапе слияний распределение выглядело как $(1, 2, 0)$ или $(1, 0, 2)$. Опять для определенности остановимся на первом варианте. Чтобы его получить, на предыдущем этапе годились бы следующие распределения: $(3, 0, 2)$ и $(0, 3, 1)$. Но второй вариант хуже, поскольку он приводится к слиянию только одной серии из файлов $B2$ и $B3$, в то время как при наличии первого варианта распределения будут слиты две серии из файлов $B1$ и $B3$. Пожеланием к предыдущему этапу было бы наличие распределения $(0, 3, 5)$, еще раньше – $(5, 0, 8)$, еще раньше – $(13, 8, 0)$ и так далее.

Это рассмотрение показывает, что метод трехфазной внешней сортировки дает желаемый результат и работает максимально эффективно (на каждом этапе сливается максимальное число серий), если начальное распределение серий между вспомогательными файлами описывается соседними числами Фибоначчи. Напомним, что последовательность чисел Фибоначчи начинается с 0, 1, а каждое следующее число образуется как сумма двух предыдущих: $0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, \ldots$.

Аналогичные (хотя и более громоздкие) рассуждения показывают, что в общем виде при использовании m вспомогательных файлов условием успешного завершения и эффективной работы метода многофазной внешней сортировки является то, чтобы начальное распределение серий между $m - 1$ файлами описывалось суммами соседних $m - 1$, $m - 2$, …, 1 чисел Фибоначчи порядка $m - 2$. Последовательность чисел Фибоначчи порядка p начинается с p нулей, $(p + 1)$-й элемент равен 1, а каждый следующий равняется сумме предыдущих $p + 1$ элементов. Начало последовательности чисел Фибоначчи порядка 4: $0, 0, 0, 0, 1, 1, 2, 4, 8, 16, 31, 61, \ldots$.

При использовании шести вспомогательных файлов идеальными распределениями серий являются следующие:

1	0	0	0	0
1	1	1	1	1
2	2	2	2	1
4	4	4	3	2
8	8	7	6	4
16	15	14	12	8
…	…	…	…	…

Понятно, что если распределение основано на числе Фибоначчи f_i, то минимальное число серий во вспомогательных файлах будет равно f_i, а максимальное – f_{i+1}. Поэтому после выполнения слияния мы получим максимальное число серий – f_i, а минимальное – f_{i-1}. На каждом этапе будет выполняться максимально возможное число слияний, и процесс сойдется к наличию всего одной серии.

Поскольку число серий в исходном файле может не обеспечивать возможность такого распределения серий, применяется метод добавления пустых серий, которые в дальнейшем как можно более равномерного распределяются между промежуточными файлами и опознаются при последующих слияниях. Понятно, что чем меньше таких пустых серий, т.е. чем ближе число начальных серий к требованиям Фибоначчи, тем более эффективно работает алгоритм.

Задание к лабораторной работе

Вариант	**Задание**
1	Выполнить сортировку целых чисел алгоритмом прямого слияния.
2	Выполнить сортировку целых чисел алгоритмом естественного слияния.
3	Выполнить сортировку целых чисел алгоритмом сбалансированного многопутевого слияния.
4	Выполнить сортировку целых чисел многофазной сортировкой (4-6 файлов).