Министерство образования и науки Российской Федерации
Санкт-Петербургский государственный электротехнический университет «ЛЭТИ»
Факультет компьютерных технологий и информатики
Кафедра вычислительной техники

Отчёт по лабораторной работе «Деревья двоичного поиска» по дисциплине «Алгоритмы и структуры данных»

Вариант 18

Выполнили студенты
факультета КТИ группы №3305
Лоуцкер Алексей и Григорьева Анна

Проверил старший преподаватель
Колинько Павел Георгиевич

Санкт-Петербург
2015

Оглавление

1. Задание.....3
2. Тестовый пример.....4
3. Оценки временной сложности операций.....5
4. Выводы.....5
5. Список использованной литературы.....5
6. Приложение: исходные тексты.....6

1. Задание

Переделать программу, составленную по теме «Хеш-таблицы», под использование деревьев двоичного поиска с автобалансировкой. Дополнить операциями над последовательностями

Вариант 18

Дерево: АВЛ-дерево с автобалансировкой

Мощность множества: 32

Формула: E = A \/ (B /\ C) \ D

Операции над последовательностями

1. Слияние (merge)

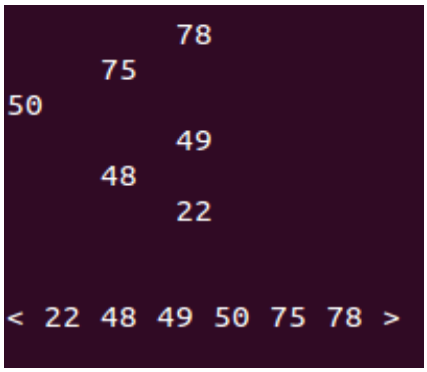
2. Включение (subst)

3. Замена (change)

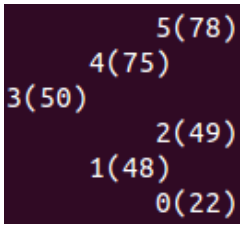
2. Описание структуры данных

Отличительная особенность нашей структуры данных в том, что она может работать в двух режимах и переключаться между ними:

1) Режим value — аналог std::set. Данные хранятся в виде бинарного дерева, без повторов, поддерживается быстрый поиск элемента по значению, относительно быстрые вставка/удаление. В этом режиме доступны 'логические' операции: объединение, пересечение, вычитание. Вывод на экран в виде дерева или упрядоченной по возрастанию последовательности



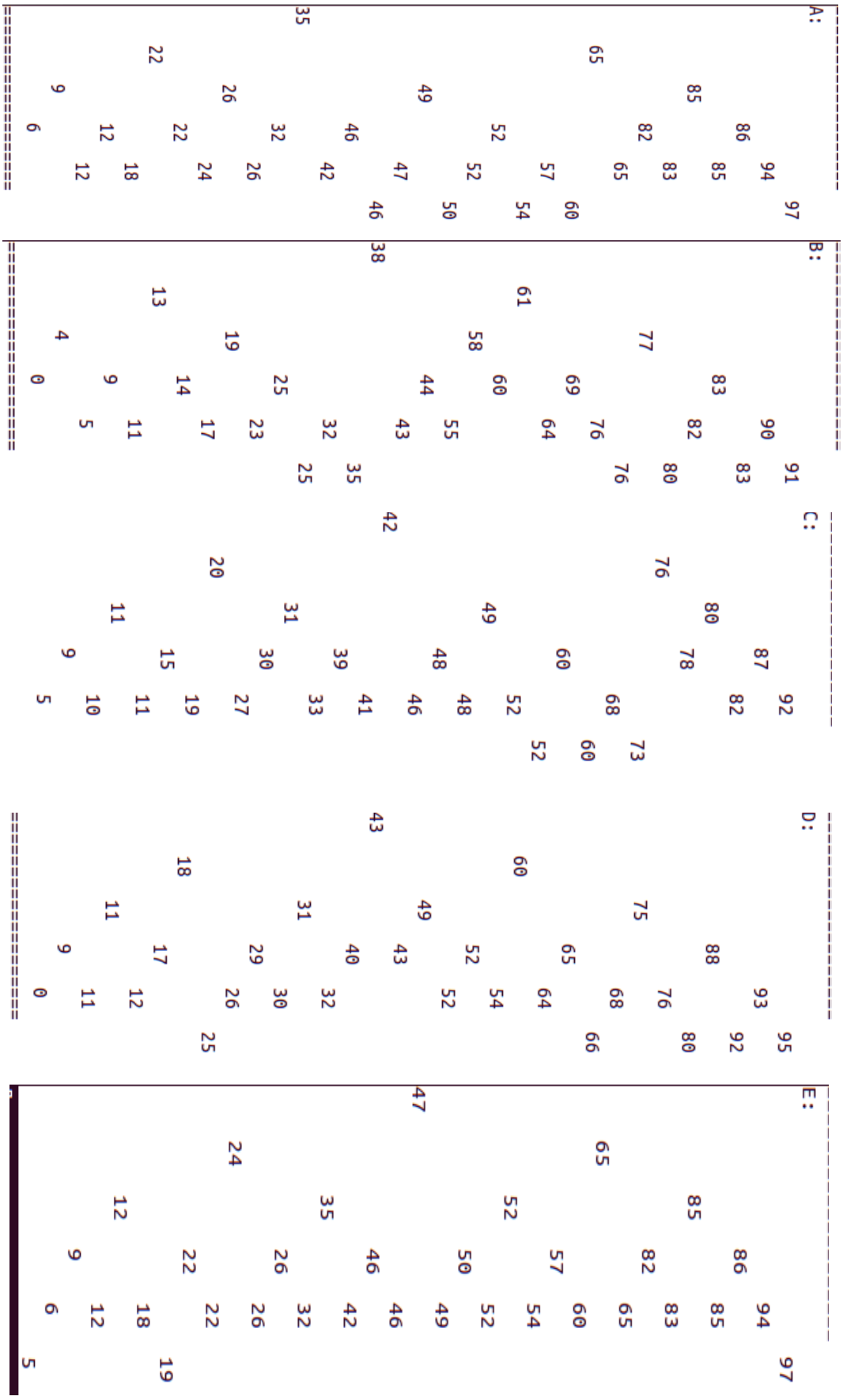
2) Режим sequence. Дерево строится не по значениям элементов, а по их позициям в последовательности.



В этом режиме доступны вставка/удаление элемента на любую позицию, операции: слияние, вставка и замена, а также вывод на экран в виде дерева или последовательности элементов в соответствии с их позицией.

Также возможно переключение между режимами с потерей позиций элементов или повторов

2. Тестовый пример



```
==========================================
< 5 9 14 18 20 24 26 28 30 32 33 35 36 37 38 43 44 51 52 53 57 58 62 64 71 78 79
 85 92 95 >
change:
< 4 3 2 1 >
==> < 5 9 14 18 20 24 26 28 30 32 4 3 2 1 >
==========================================
merge:
< 0 0 0 1 1 1 97 98 99 >
==> < 0 0 0 1 1 1 1 2 3 4 5 9 14 18 20 24 26 28 30 32 97 98 99 >
==========================================
subst:
< 1 2 3 4 5 6 7 8 9 >
==> < 0 0 0 1 1 1 1 2 3 4 5 6 7 8 9 1 2 3 4 5 9 14 18 20 24 26 28 30 32 97 98 99
 >
```

3. Оценки временной сложности операций

1) Режим value

Функции поворотов и балансировки не содержат ни циклов, ни рекурсии, а значит выполняются за постоянное время, не зависящее от размера АВЛ-дерева.

Операции вставки и удаления (а также поиска) выполняются за время, пропорциональное высоте дерева, т.к. в процессе выполнения этих операций производится спуск из корня к заданному узлу, и на каждом уровне выполняется некоторое фиксированное число действий. А в силу того, что АВЛ-дерево является сбалансированным, его высота зависит логарифмически от числа узлов. Таким образом, время выполнения всех трех базовых операций логарифмически зависит от числа узлов дерева.

Двухместные операции получают результат в виде вектора значений, а затем последовательно вставляют элементы в новое дерево

Операция	Сложность
балансировка, повороты	O(1)
вставка, удаление, поиск	O(log(n))
пересечение, объединение, вычитание	O(n*log(n))

2) Режим sequence

При вставке/удалении элемента последовательности необходимо не только удалить его из дерева, но и обновить номера остальных элементов, поэтому сложнсть — линейная

Операции над последовательностями создают из двух последовательностей одну с помощью вставок и удалений, поэтому их сложность пропорциональна квадрату числа элементов.

Преимущество выбранного способа хранения последовательности - быстрый вывод и доступ к элементу по его позиции при любом количестве повторов; не требует дополнительной памяти

Операция	Сложность
балансировка, повороты	O(1)
доступ по номеру	O(log(n))
вставка, удаление, вывод на экран	O(n)
слияние, подстановка, замена	O(n^2)

4. Выводы

Реализованная структура данных, также как и хеш-таблица, поддерживает простейшие и двухместные операции над множеством ключей. За счет строгой внутренней организации АВЛ-дерево позволяет получить выигрыш в скорости при выполнении двухместных операций, но из-за необходимости регулярно выполнять балансировку, вставка, поиск и удаление требуют больше времени.
Режим последовательности легко справляется с большим количеством повторяющихся элементов, однако, если таких не слишком много, то вариант хранения позиций элементов другим способом будет иметь лучшую среднюю оценку.

5. Список использованной литературы

1) Алгоритмы и структуры данных: методические указания к лабораторным работам, практическим занятиям и курсовому проектированию. Часть 2 / сост. П. Г. Колинько. - СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2015

2) http://stackoverflow.com

3) http://cplusplus.com

4) http://habrahabr.ru/post/150732/

6. Приложение: исходные тексты

Исходный код доступен в репозитарии по адресу:

github.com/alout1/avl-tree/sequences