Лабораторная работа № 2 по курсу дискретного анализа: Сбаласированные деревья

Выполнил студент группы М8О-208Б-20 МАИ *Попов Матвей*.

Условие

1. Необходимо создать программную библиотеку, реализующую указанную структуру данных, на основе которой разработать программу-словарь. В словаре каждому ключу, представляющему из себя регистронезависимую последовательность букв английского алфавита длиной не более 256 символов, поставлен в соответствие некоторый номер, от 0 до $2^{64} - 1$. Разным словам может быть поставлен в соответствие один и тот же номер.

2. Вариант задания: B-дерево.

3. Операции: вставка, поиск, удаление элемента, сахранение словаря в файл, загрузка словаря из файла.

Метод решения

Было принято решение реализовать отдельный класс для дерева и отдельную структуру для узла дерева, чтобы класс дерева и все методы для работы с ним представляли из себя обёртку для идентичных функций над корневым узлом.

Описание программы

Программа состоит из одного файла, в котором помимо функции main находятся два пространства имён: для узла дерева и самого дерева. В этих пространствах имён содержатся функции для вставки пары в дерево, поиска, удаления, сохранения дерева в файл и загрузки дерева из файла.

Дневник отладки

1. Программа загружена на чекер без функций сохранения и загрузки, чтобы убедиться в работоспособности вставки, поиска и удаления. Выяснилось, что функция удаления написана неверно.

2. Переписана функция удаления, из-за многочисленных Segmentation Fault было принято решение отказаться от разбиения функции удаления на функции для обработки частных случаев.

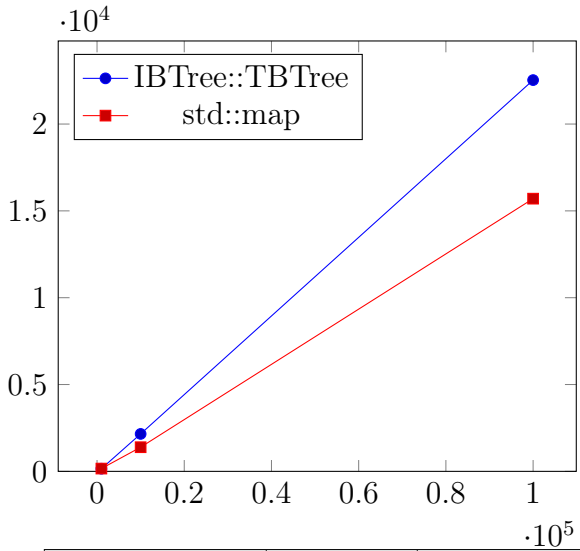
3. Написаны функции сохранения и загрузки, однако программа не проходит тест №7.

4. Функции сохранения и загрузки переписаны дважды, программа проходит все тесты чекера.

Тест производительности

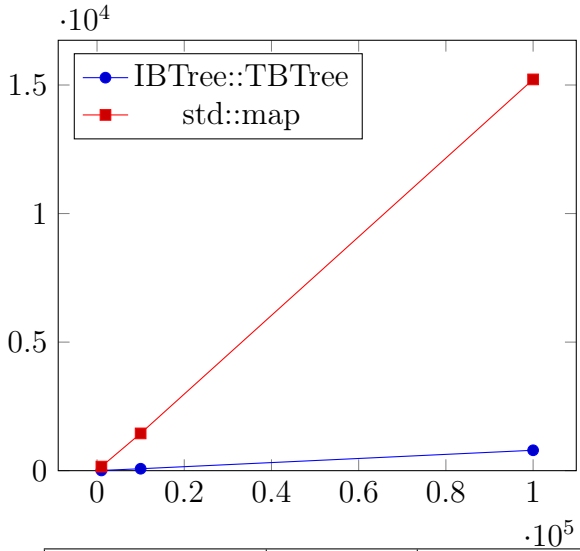
Во всех графиках по оси X отложено время выполнения (в миллисекундах), по оси Y — количество команд.

Вставка



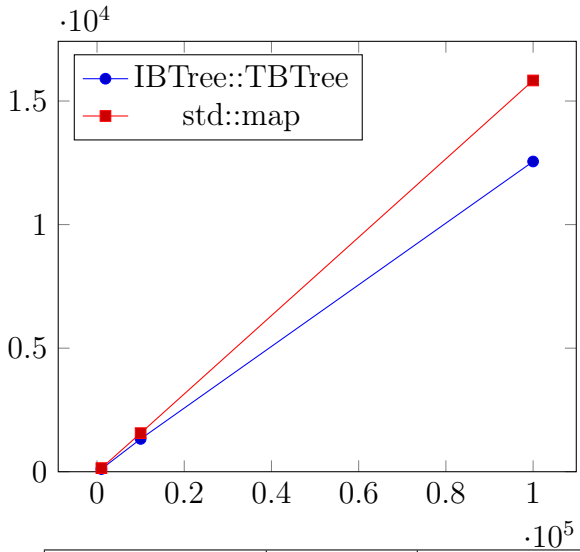
Кол-во строк	std::map	IBTree::TBTree
1000	160	153
10000	1390	2156
100000	15703	22531

Поиск



Кол-во строк	std::map	IBTree::TBTree
1000	158	8
10000	1449	75
100000	15218	793

Удаление



Кол-во строк	std::map	IBTree::TBTree
1000	146	108
10000	1561	1327
100000	15832	12553

Таким образом, операции над B-деревом имеют ту же сложность, что и над std::map, вставка и удаление по времени работают примерно одинаково, но поиск в B-дереве

происходит гораздо быстрее. Это может быть связано с тем, что тестирование проходило над B-деревом со степенью 500, узлы такого B-дерева занимают больше памяти, однако чтобы найти какую-либо пару, в среднем понадобится меньше итераций.

Недочёты

После многочисленных правок код программы получился практически нечитаемым, а размер файла составил 670 строк.

Выводы

B-дерево — очень полезная структура для хранения данных, так как операции вставки, поиска и удаления элементов в среднем работают быстрее, чем в несбалансированном бинарном дереве поиска. Однако недостатком этой структуры является то, что реализации этих функций очень сложны.