Типы и Структуры Данных

Лабораторная работа №6

*Хеш-таблица, дерево.*

Выполнил студент группы: ИУ7-36

Гасанзаде Мухаммедали

2018

# Задание:

Построить хеш-таблицу для слов. Осуществить поиск указанного слова в двоичном дереве поиска (ДДП) и в хеш-таблице, если его нет, то добавить его (по желанию пользователя) в дерево и, соответственно, в таблицу. Сравнить время поиска и количество сравнений при использовании ДДП, сбалансированных деревьев и хеш-таблиц.

# Техническое задание:

Цель работы - освоить навыки работы с хеш-таблицами. Закрепить навыки работы с двоичными деревьями.

# Входные данные.

На вход программа получает пункт меню, выбранный пользователем (Show tree/balanced tree/ open hash table/closed hash table/compare time/search/exit (1/2/3/4/5/6/0): ). Слова программа считывает из файла.

# Выходные данные.

В зависимости от выбора пользователя, программа выводит соответствующие результаты

(например, выводит дерево).

# СД:

В программе были использованы созданные структуры. #Деревья typedef struct tree { struct tree \*left; struct tree \*right; char \*data; } tree;

typedef struct Btree { char \*data; int height; struct Btree \*left; struct Btree \*right; } Btree;

# Результат работы:

**Время работы и сравнения:**

|  |  |  |
| --- | --- | --- |
| **Алгоритм:** | **Время:** | **Сравнения**  **:** |
| Дерево | 0.00066  7 | 6 |
| Сбалансированное дерево | 0.00051  5 | 6 |
| Открытая хеш-таблица | 0.00012  3 | 3 |
| Закрытая хеш-таблица | 0.00008  1 | 1 |

1. **Выводы.**

Хеш-таблицы позволяют ускорить процесс поиска, но они всегда занимают больше памяти (за исключением 100% заполнения). Их использование имеет смысл если есть много данных или если их количество можно определить заранее с высокой точностью. В обратном случае эффективнее использовать сбалансированное дерево, которое эффективнее обычного.

# Контрольные вопросы.

1. Чем отличается идеально сбалансированное дерево от АВЛ дерева?
   * Если при добавлении узлов в дерево мы будем их равномерно располагать слева и справа, то получится дерево, у которого число вершин в левом и правом поддеревьях отличается не более, чем на единицу. Такое дерево называется идеально сбалансированным. Адельсон-Вельский и Ландис сформулировали менее жесткий критерий сбалансированности таким образом: двоичное дерево называется сбалансированным, если у каждого узла дерева высота двух поддеревьев отличается не более чем на единицу. Такое дерево называется АВЛ-деревом.
2. Чем отличается поиск в АВЛ-дереве от поиска в дереве двоичного поиска?
   * Отличается эффективностью (трудоемкость – O(log2n)).
3. Что такое хеш-таблица, каков принцип ее построения?
   * Хеш-таблица - массив, заполненный в порядке, определенным хеш-функцией. По значению ключа определяется индекс элемента массива, в котором хранится информация. Хеш-функция ставит в соответствие каждому ключу ki индекс ячейки j, где расположен элемент с этим ключом, таким образом: h(ki) = j, если j=(1,m), где j принадлежит множеству от 1 до m, а m. – размерность массива.
4. Что такое коллизии? Каковы методы их устранения?
   * Может возникнуть ситуация, когда разным ключам соответствует одно значение хеш- функции, то есть, когда h(K1)=h(K2), в то время как K1 ≠ K2. Такая ситуация называется коллизией. Существует несколько возможных вариантов разрешения коллизий, которые имеют свои достоинства и недостатки. Первый метод – внешнее (открытое) хеширование (метод цепочек) Другой путь решения проблемы, связанной с коллизиями – внутреннее (закрытое) хеширование (открытая адресация).

5)В каком случае поиск в хеш-таблицах становится неэффективен?

* + При увеличении кол-ва сравнений, коллизий. В случаях, когда нельзя предсказать примерный размер таблиц.

1. Эффективность поиска в АВЛ деревьях, в дереве двоичного поиска и в хеш-таблицах.
   * Трудоемкость: Дерево двоичного поиска: O(log2n)) .. O(n)). АВЛ дерево: O(log2n)).

Хеш-таблица: минимальная - О(1), может увеличиться из-за коллизий.