Компьютерная математика II

Дисциплина для студентов 1-го курса специальности «Компьютерная математика и системный анализ» ММФ БГУ

Тема 9. Бинарное дерево поиска. Реализация на основе ООП

доц. Лаврова О.А., кафедра дифференциальных уравнений и системного анализа (ауд. 329)

апрель, 2024

9.1 Определения

Вершина — это базовая сущность для описания графа.

Граф — это пара (V, E) , состоящая из непустого множества вершин V и множества пар вершин E .

Пара вершин из множества E может быть упорядоченной и неупорядоченной. **Ребро** — это неупорядоченная пара вершин.

Граф называется *неориентированным*, если множество E состоит только из ребер.

Граф называется *связным*, если для любых двух вершин из множества V существует маршрут из последовательности ребер графа, соединяющий эти вершины.

Цепь -- это маршрут, состоящий из различных ребер. **Цикл** -- это циклическая цепь.

Граф называется *ациклическим*, если он не содержит циклов.

Дерево — это неориентированный связный ациклический граф.

Корневое дерево — это дерево с одной выделенной вершиной, которая называется *корнем* дерева.

Бинарное дерево — это корневое дерево, вершины которого, исключая корень, содержатся в двух непересекающихся множествах: *левое поддерев* и *правое поддерев*. При этом каждое поддерев либо является пустым множеством, либо бинарным деревом.

Любая вершина называется *родительской вершиной* для своих поддеревьев.

Каждая родительская вершина бинарного дерева может иметь не более двух *вершин-потомков*, левую и правую.

Вершина, которая не имеет потомков, называется *листом* дерева.

Бинарное дерево поиска (binary search tree) — это бинарное дерево, каждая вершина которого имеет значения одного типа, для которых определены операции сравнения (например, значениями вершин являются объекты типа `int` или `float`). При этом значения вершин непустого левого поддерева меньше ($<$), чем значение соответствующей родительской вершины, а значения вершин непустого правого поддерева больше либо равны ($\geq$) значению соответствующей родительской вершины.

Бинарное дерево поиска — это упорядоченная динамическая структура данных, которая позволяет быстро производить поиск элементов, а также удалять и добавлять элементы.

9.2 Представление бинарного дерева поиска

Анализируя структуру бинарного дерева поиска, нетрудно видеть, что его построение можно свести к умению строить **элемент дерева** (вершина) – сущность, имеющую значение и связанную с левым и правым поддеревом, если они существуют.

Тогда уместно рекурсивное определение: **бинарное дерево** – это структура данных, которая либо пуста, либо содержит элемент дерева (корневую вершину), связанный с двумя различными бинарными деревьями, называемыми левым и правым поддеревьями.

В силу этого, представление бинарного дерева сводится к представлению элементов дерева и установлению/описанию связей между элементами дерева.

Элемент дерева будет представлен экземпляром класса `BinaryNode` .

Бинарное дерево поиска будет представлено экземпляром класса `BinaryTree` .

Пустое бинарное дерево поиска будет представлено экземпляром класса `EmptyNode` .

Экземпляр класса `BinaryTree` будет содержать ссылку на корень дерева. Корень дерева будет представлен экземпляром класса `EmptyNode` , если дерево пустое, или экземпляром класса `BinaryNode` , если дерево не пусто.

Для связывания элемента дерева со своими поддеревьями экземпляр класса `BinaryNode` будет содержать ссылки на корни левого и правого поддеревьев. Корень поддерева представляется экземпляром класса `EmptyNode`, если поддерево пустое, или экземпляром класса `BinaryNode`, если поддерево не пусто.

В частности, лист дерева будет представлен экземпляром класса `BinaryNode`, правое и левое поддерева которого являются экземплярами класса `EmptyNode`.

9.3 Проектирование классов

Бинарное дерево поиска будем описывать с помощью трех классов: `BinaryTree`, `BinaryNode`, `EmptyNode`.

Класс `BinaryTree` является основным. Он определяет шаблон для создания объекта, представляющего бинарное дерево поиска.

Класс `BinaryNode` является вспомогательным. Он определяет шаблон для создания объекта, представляющего элемент (вершину) бинарного дерева поиска.

Класс `EmptyNode` является вспомогательным. Он определяет шаблон для создания объекта, представляющего "пустой" элемент (вершину) бинарного дерева поиска без значения и без поддеревьев. Экземпляры класса `EmptyNode` будут использоваться для представления пустого дерева и пустых поддеревьев.

Классы `BinaryTree`, `BinaryNode`, `EmptyNode` наследуются только от `object`.

Экземпляр класса `BinaryTree` содержит атрибут `root`, который представляет корень дерева. Атрибут `root` хранит ссылку на экземпляр класса `EmptyNode`, если дерево пустое, или экземпляр класса `BinaryNode`, если дерево не пусто.

При таком связывании классов говорят о реализации концепции ООП **композиция**. Экземпляр класса `BinaryTree` называют *объектом-контейнером* для экземпляра класса `EmptyNode` или `BinaryNode`.

Экземпляр класса `BinaryNode` содержит атрибуты `left` и `right`, которые хранят ссылки на экземпляры классов `BinaryNode` или `EmptyNode`. Экземпляр класса `BinaryNode` является *объектом-контейнером* для двух экземпляров класса `BinaryNode` или `EmptyNode`, которые ссылаются на корневые вершины левого и правого поддеревьев.

Проектирование класса `BinaryTree`

Класс `BinaryTree` определяет шаблон для создания объекта, представляющего бинарное дерево поиска.

Определение класса `BinaryTree` содержит ТРИ метода: `__init__`, `__repr__` и `insert`.

Метод инициализации экземпляра класса `__init__` вызывается без аргументов и инициализирует единственный атрибут экземпляра класса `root` экземпляром класса `EmptyNode`:

```
def __init__(self):  
    self.root = EmptyNode()
```

Назначение атрибута `root` -- это хранение ссылки на корень дерева.

Метод строкового представления экземпляра класса `__repr__` возвращает строковое представление для корня дерева `root`:

```
def __repr__(self):  
    return repr(self.root)
```

Метод `insert(value)` является методом вставки в дерево НОВОГО элемента. Новый элемент всегда является листом со значением `value`.

Метод вставки элемента в дерево реализуется через метод вставки элемента в корневую вершину дерева `root`. При этом атрибут `root` переопределяется ссылкой на новый объект для представления корня

```
def insert(self, value):  
    self.root = self.root.insert(value)
```

Определим класс `BinaryTree`:

```
In [1]: class BinaryTree:  
        def __init__(self):  
            self.root = EmptyNode()  
  
        def __repr__(self):  
            return repr(self.root)  
  
        def insert(self, value):  
            self.root = self.root.insert(value)
```

Создать экземпляр класса `BinaryTree` пока нельзя, так как не определен класс `EmptyNode`

```
In [2]: tree = BinaryTree()
```

```

-----
NameError                                Traceback (most recent call last)
Cell In[2], line 1
----> 1 tree = BinaryTree()

Cell In[1], line 3, in BinaryTree.__init__(self)
      2 def __init__(self):
----> 3     self.root = EmptyNode()

NameError: name 'EmptyNode' is not defined

```

Проектирование класса BinaryNode

Класс `BinaryNode` определяет шаблон для создания объекта, представляющего элемент бинарного дерева поиска.

Определение класса `BinaryNode` содержит ТРИ метода: `__init__`, `__repr__` и `insert`.

Метод инициализации экземпляра класса вызывается с тремя аргументами `left`, `value` и `right` для инициализации одноименных атрибутов экземпляра класса `left`, `value` и `right` соответствующими значениями

```

def __init__(self, left, value, right):
    self.left = left
    self.value = value
    self.right = right

```

Назначение атрибута `left` -- это хранение ссылки на корень левого поддерева.

Назначение атрибута `right` -- это хранение ссылки на корень правого поддерева.

Назначение атрибута `value` -- это хранение значения вершины.

Метод строкового представления экземпляра класса `__repr__` возвращает кортеж, состоящий из строкового представления корня левого поддерева `left`, из строкового представления значения `value` вершины и из строкового представления корня правого поддерева `right`

```

def __repr__(self):
    return f'({self.left}, {self.value}, {self.right})'

```

Метод `insert(value)` является методом вставки в текущую вершину НОВОЙ вершины со значением `value`. Метод `insert` содержит основной алгоритм построения бинарного дерева поиска и будет определен позже

```

def insert(self, value):
    ...

```

Определим класс `BinaryNode` :

```
In [3]: class BinaryNode:
        def __init__(self, left, value, right):
            self.left = left
            self.value = value
            self.right = right

        def __repr__(self):
            return f'({self.left}, {self.value}, {self.right})'

        def insert(self, value):
            ...
```

Создадим экземпляр класса BinaryNode :

```
In [4]: node = BinaryNode(None, 5, None)
```

Сделаем вывод строкового представления для экземпляра класса BinaryNode :

```
In [5]: print(node)
```

```
(None, 5, None)
```

Проектирование класса EmptyNode

Класс EmptyNode определяет шаблон для создания объекта, представляющего "пустой" элемент (вершину) бинарного дерева поиска без значения и без поддеревьев. Экземпляры класса EmptyNode будут использоваться для представления пустого дерева и пустых поддеревьев.

Класс EmptyNode определяется ДВУМЯ методами: __repr__ и insert .

Метод инициализации экземпляра класса отсутствует, так как экземпляры класса EmptyNode не хранят никаких данных.

Метод строкового представления экземпляра класса __repr__ возвращает один символ *

```
def __repr__(self):
    return '*'
```

Метод insert(value) является методом вставки в пустую вершину НОВОЙ вершины со значением value .Метод insert возвращает новый экземпляр класса BinaryNode со значением value .Левое и правое поддерево определяются ссылками на пустую вершину

```
def insert(self, value):
    return BinaryNode(self, value, self)
```

При вставке НОВОГО элемента в бинарное дерево поиска ссылка на экземпляр класса `EmptyNode` будет заменяться на ссылку на новый экземпляр класса `BinaryNode`.

Экземпляр класса `EmptyNode` создается только один раз при создании экземпляра класса `BinaryTree`. Каждое пустое поддерево для экземпляров класса `BinaryNode` будет ссылаться на ЕДИНСТВЕННЫЙ экземпляр класса `EmptyNode`.

Определим класс `EmptyNode`:

```
In [6]: class EmptyNode:
        def __repr__(self):
            return '*'

        def insert(self, value):
            return BinaryNode(self, value, self)
```

Создадим экземпляр класса `EmptyNode`:

```
In [7]: empty_node = EmptyNode()
```

Сделаем вывод строкового представления для экземпляра класса `EmptyNode`:

```
In [8]: print(empty_node)
```

*

Создадим экземпляр класса `BinaryTree`:

```
In [9]: tree = BinaryTree()
```

Бинарное дерево поиска пусто, корень представлен экземпляром класса `EmptyNode`.
Пустое дерево выводится с помощью символа *

```
In [10]: print(tree)
```

*

Вставим в дерево новый элемент со значением 10:

```
In [11]: tree.insert(10)
```

Сделаем вывод дерева:

```
In [12]: print(tree)
```

(* , 10, *)

Дальнейшая вставка элементов в дерево невозможна, так как не определен метод `insert` класса `BinaryNode`.