SPb HSE, МОАД ВБИБ, осень 2024/25
Практика по алгоритмам #15

Treap

27 января

Собрано 27 января 2025 г. в 15:22

Содержание

Treap

1. **Модификации отрезка**
   a) Запросы: `insert(i,x)`, `del(i)`, `get_sum(l,r)`
   b) Отложенные операции: `set(l,r,value)`.
   c) Добавим запросы `reverse(l,r)` и `rotate(k)`.
   d) `(*)` То же самое, но `add(l,r,value)` по модулю 5, а `get_sum` по-прежнему без модуля.

2. **Excel**

   Есть excel-табличка. Научитесь за $\mathcal{O}(\log n)$ обрабатывать запросы

   a) Столбец j подвинуть влево-вправо на d.
   b) Строку i подвинуть вверх-вниз на d.
   c) Поменять/прочитать ячейку $[i, j]$.

3. **Улучшаем и изучаем декартово дерево**
   a) `insert` через один спуск и один `split`.
   b) `del` через один спуск и один `merge`.
   c) Дан отсортированный массив x_i. Постройте Treap за $\mathcal{O}(n)$ (есть простое решение).

4. **Время работы Treap**

   Пусть дан массив $x_1, x_2, \ldots, x_n$, по нему был построен Treap. На лекции мы показали, что для каждого i глубина узла с ключом x_i имеет матожидание $\mathcal{O}(\log n)$.

   Докажите, что матожидание времени работы `split` и `merge` есть $\mathcal{O}(\log n)$.

5. **Копирование памяти**

   Соделайте массив, который за $\mathcal{O}(\log n)$ умеет $read(i)$, $write(i, x)$, $copy(l, r, i)$.

6. **Детская персистентность**

   Придумайте персистентный массив, который умеет делать

   a) Обращение за $\mathcal{O}(1)$, модификацию за $\mathcal{O}(n)$.
   b) Обращение за $\mathcal{O}(m)$, модификацию за $\mathcal{O}(1)$.
   c) Частичная персистентность (модифицировать можно только последнюю версию). Обращение за $\mathcal{O}(\log m)$, модификацию за $\mathcal{O}(1)$.

7. **Персистентность для взрослых**

   Придумайте персистентный массив, который умеет делать обращение за $\mathcal{O}(\log n)$, модификацию за $\mathcal{O}(\log n)$.

8. **Персистентное удаление**

   Напишите явно код персистентного *ленивого* удаления для BST.

9. `(*)` Персистентный CHM

10. `(*)` Найдите матожидание максимальной глубины вершины в случайном дереве.

Разбор задач практики

8. **Персистентное ленивое удаление**

```
pnode Del(pnode v, int x):
    if (v->x == x)
        return new node {v->x, v->l, v->r, 1};
    if (x < v->x)
        return new node {v->x, Del(v->l, x), v->r, v->isdeleted};
    else
        return new node {v->x, v->l, Del(v->r, x), v->isdeleted};
```