RMQ и LCA

Range Minimum Query

≡ Вход: массив длины n

Запрос: $(i, j) \to \min$ на отрезке $[i, j]$

„Прямое вычисление" $\Rightarrow O(n)$

Динамическая RMQ

Запрос:

- $\mathrm{Min}(i, j)$
- $\mathrm{Change}(i, x)$

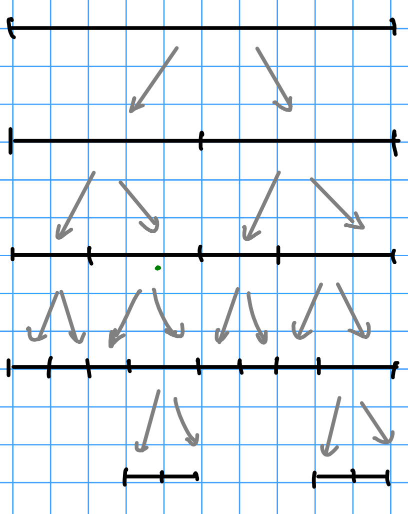
≡ Дерево отрезков

Бинарное дерево ассоц. с массивом $[1, n]$

С корнем ассоц. отрезок $[1, n]$.

С $\forall$ внутр листом ассоц. некс. отрезок $[i, j]$, а с его потомками — $[i, m]$, $[m+1, j]$, где $m = \lfloor (i+j)/2 \rfloor$

Листья — это одноэлементные массивы



Утв: Глубина дерева — $O(\log n)$

Идея: Давайте в $\forall$ вершине v сохраним $\min(\Delta(v))$, где $\Delta(v)$ - ассоц. с v отрез.

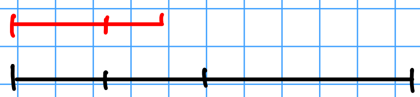
Утв: Можно построить за $O(n)$

Утв: $\forall$ отрезок Δ можно разбить на $O(\log n)$ "канонических" отрезков за $O(\log n)$

= отрезки из дерева

```
▷ Decompose(v, Δ):        // Δ ⊆ Δ(v)
      if Δ = ∅:
          return ∅
      if Δ = Δ(v):
          return Δ(v)
      return Decompose(v.left, Δ ∩ Δ(v.left))
             ∪ Decompose(v.right, Δ ∩ Δ(v.right))
```

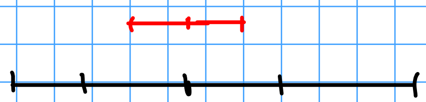
1. Пусть у $\Delta(v)$ и Δ совпадает один из концов



Подобно бинарному поиску правого конца $\Rightarrow$ $O(\log n)$ итераций

2. Пусть у $\Delta(v)$ и Δ нет совпад. концов.

Найдём первую вершину с нетривиальным разбиением отрезка $\to$ два случая № 1.



◁

NB: отрезки получились непересекающиеся

Следствие: $\mathrm{Min}(i, j)$ можно реализовать за $O(\log n)$

$\mathrm{Change}(i, x)$?

Замечание: $\forall$ эл-т массива входит в $O(\log n)$ канонических отрезков.

Все такие отрезки лежат на пути из листа в корень $\Rightarrow$ $\mathrm{Change}(i, x)$ за $O(\log n)$

Можно ли быстрее? Обе операции не ускорить, т.к. иначе получим сортировку за $O(n\log n)$

Замечание 1: можно реализовать на массиве

Замечание 2: работает для $\forall$ ассоц. операции

Например, RSQ (range sum query).

Статическая постановка задачи RMQ

Массив не меняется $\Rightarrow$ можно предобработать

- „Полное предвычисление"

  Заведём таблицу $n \times n$ и запишем туда ответы на все запросы.

  $O(n^2)$ на предобработку
  $O(1)$ на запрос
  $(O(n^2), O(1))$

- $(O(n), O(1))$ алгоритм для RSQ

  Считаем частичные суммы $S[i] = \sum_{j=1}^{i} M[j]$

  Тогда $RSQ(i, j) = S[j] - S[i-1]$

  Не подходит для RMQ, т.к. у min нет обратной

- $(O(n\log n), O(1))$ алгоритм для RMQ

  „Метод разреженной таблицы".

  Сохраним ответы для всех отрезков Δ:
  $|\Delta| = 2^k$ для некоторого k

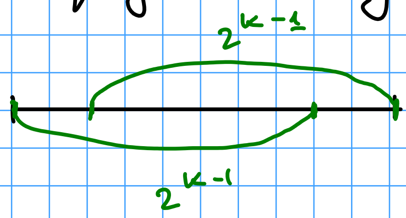
  Всего таких „канонических" отрезков $O(n\log n)$

  Упр: заполнить таблицу за $O(n\log n)$

  Утв: $\forall$ отрезок разбивается на ≤ 2 канонических

▷ 1. $|\Delta| = 2^k$

2. $2^{k-1} < |\Delta| < 2^k \Rightarrow$ покрываете двумя отрезками длины 2^{k-1}



◁

$\Rightarrow$ Запрос за $O(1)$

Замечание: пользуемся идемпотентностью min

$\min(a, a) = a$

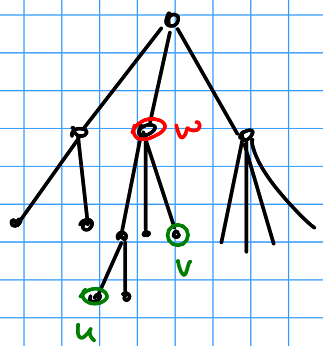
Замечание: нужно быстро находить ближайшую степень двойки (таблица или битовые операции)

LCA (least common ancestor)

„ближайший общий предок"

Вход: дерево (корневое)

Запрос: $LCA(v, u)$ — ближайший общий предок v, u — некоторая w: $u, v \in$ поддереву w и w — самая глубокая из таких вершин



$LCA(u, v) = LCA(v, u)$

Если u в поддереве $v \Rightarrow$ $LCA(u, v) = v$

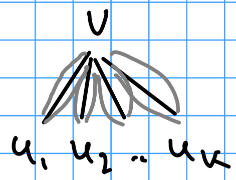
„Прямое вычисление" $O(n)$

„Полное предвычисление" $(O(n^2), O(1))$

Сведение LCA к RMQ

$\equiv$ $ET(T)$ – Эйлеров обход дерева

$ET(v)$



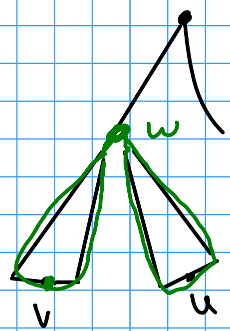
$$ET(v) = v\,ET(u_1)\,v\,ET(u_2)\,v \ldots ET(u_k)\,v$$

$ET(\ell) = \ell$, где ℓ – лист

$|ET(v)| = 2n - 1$

$ET(T)$ ~ „протокол" обхода в глубину.

$LCA(v, u)$ – вершина с наименьшей глубиной на отрезке ET от первого вхождения v до первого вхождения u



] $first[u]$ – индекс первого вхождения u в $ET(T)$

$$LCA(v, u) = RMQ_{ET(T)}(first[v], first[u])$$

$RMQ_{ET(T)}$ – min глубина вершина на отрезке

Следствие: $(O(n \log n), O(1))$ алгоритм для LCA

Замечание: LCA сводим ± 1 - RMQ

Замечание: можно свести RMQ к LCA (при помощи декартова дерева)

Факт: $\exists$ $(O(n), O(1))$ алгоритм для ± 1-RMQ

Следствие: $\exists$ $(O(n), O(1))$ алгоритм для LCA

Следствие: $\exists$ $(O(n), O(1))$ алгоритм для RMQ