GSourceCode

GSOURCECODE

INTRODUCCIÓN A LOS ÁRBOLES DEL SEGMENTO (INTERVALO DE CONSULTA MÍNIMA)

12 de marzo 2014 · *por* MGhareeb · *en Algoritmos , Estructura de Datos · Deja un comentario*

1 0 Valora este

Rango de consultas mínimo

Dado un vector Ade longitud N, responder $getMin(i, j)$qeuries. $getMin(i, j)$debe devolver el índice m, donde $A[m] = min_{i<k<j}\{A[k]\}$.

Hay muchas maneras de hacer esto. En el caso en que Ase les permite 's valores de los elementos para cambiar, y sin embargo, tenemos que mantener $getMin(i, j)$para que funcione correctamente, un árbol segmento es nuestra mejor opción. Que sigue es una breve explicación de los árboles del segmento para este ejemplo concreto. Voy a poner una explicación más generalizada en otro post.

Árboles del segmento

La idea detrás de un árbol segmento es construir un **árbol binario** donde cada nodo representa un segmento de A. Para la gama de consultas mínimo, cada nodo del árbol contendrá el valor de $getMin(segment)$(el índice del valor mínimo en el segmento del nodo de la matriz).

Ahora, ¿cómo están los nodos y segmentos asignados? Para simplificar, supongamos que A'longitud s es una potencia de dos:

- Comenzando en el nivel más bajo del árbol, tendremos Nnodos hoja cada uno que representa un elemento de A. Como se muestra en la siguiente figura, si etiquetamos nodos hoja de 0a $N - 1$de izquierda a derecha, cada n

INTRODUCE TU EMAIL PARA SUSCRIBIRTE

Enter your email address

Siga!

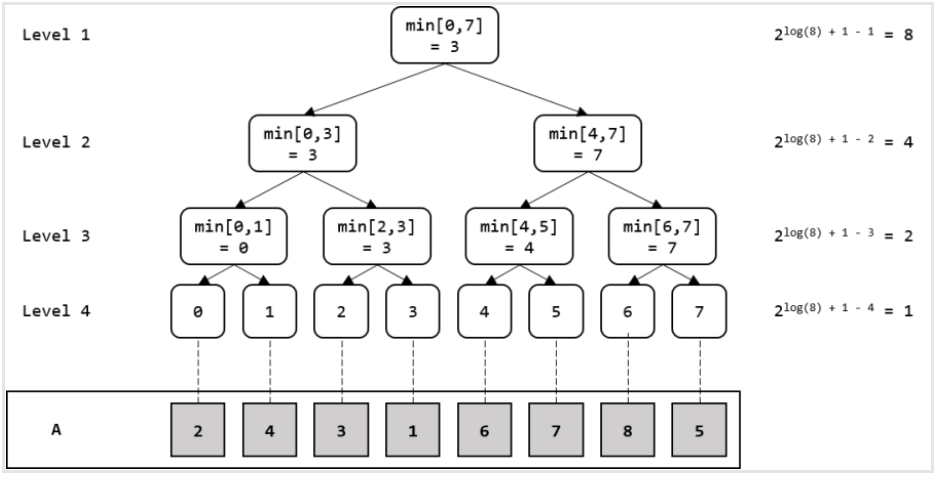
BÚSQUEDA

Search this site...

EXPLORAR UNA CATEGORÍA

- Para cada nodo interno $node$ a partir
  calcula $node.value$ como sigue:
  $node.value = m, A[m] = min\{A[node.l$
  . Esto quiere decir que un nodo padre
  uno de sus hijos tales que $A[node.v$
- Dado que este es un árbol binario con
  N los nodos, el árbol tendrá la altura
  tendrá un nodo, la raíz, que sostiene $getMin(0, N-1)$. Esto significa que cada nodo en un nivel L representará un segmento de longitud $2^{log(N)+1-L}$.

- Utilidades
ndroid
structura de Datos
plicaciones de escritorio
ME
nvestigación de
peraciones



La aplicación va a ser sencillo de aquí en adelante. Vamos a representar el árbol como un conjunto `de árboles` de longitud `2 << ceil (log2 (N + 1))` . Nodo `i` tiene un hijo izquierdo `2 * i` y un hijo derecho `2 * i + 1` . Sobre una `actualización (i)` , vamos a recorrer el árbol de arriba a abajo, cambio de valor de matriz en cuando lleguemos a la hoja `i` , y actualización el árbol, ya que volver a subir. En un `getMin (rango)` , vamos a RECURSE desde la raíz hasta que visitamos todos los subsegmentos máximas de la `gama` y devolver el valor requerido. Aquí está una implementación en `C ++` con este tutorial como referencia:

```
# include <iostream>
# include <vector>
# include <cstring>
# include <cmath>
utilizando  el espacio de nombres  std;

clase  segTree {
    // O (n)
    int  * array, * árbol;
    int  arrayLen, treeLen;

    // O (n)
    void  initialize ( int  nodo, int  b, int  e)
        si  (b == e)
            árbol [nodo] = b;
        otra cosa  {
            // Recurse
            inicializar (2 * nodo, b, (b + e) / 2
```

```
            inicializar (2 * nodo + 1, (b + e) /
            // Valor actualización
            si  (array [árbol [2 * nodo]] <= arra
                Árbol [nodo] = árbol [2 * nodo];
            demás
                árbol [nodo] = árbol [2 * nodo +
        }
    }
pública :
    segTree ( int  * array, int  arrayLen) {
        este -> arrayLen = arrayLen;
        este -> array = array;
        este -> treeLen = 2 << ( int ) ceil (log2
        cout << "treeLen ="  << treeLen << endl;
        este -> árbol = nuevo  int [treeLen];
        memset (árbol, -1, sizeof ( int ) * treeL
        inicializar (1, 0, arrayLen - 1);
    }

    // O (log n)
    void  update ( int  i, int  v, int  nodo = 1,
        e = arrayLen - 1 - e;
        si  (b == e) {
            array [i] = v;
        } otro  {
            int  mid = (b + e) / 2;
            si  (i <= centro)
                actualización (i, v, 2 * nodo, b,
            demás
                actualización (i, v, 2 * nodo + 1
            si  (array [árbol [2 * nodo]] <= arra
                Árbol [nodo] = árbol [2 * nodo];
            demás
                árbol [nodo] = árbol [2 * nodo +
        }
    }

    // O (log n)
    int  consulta ( int  i, int  j, int  nodo = 1
        e = arrayLen - 1 - e;
        // Mala intervalo
        si  (i> e || j <b)
            devolver  -1;
        // Buen intervalo
        si  (b> = i && e <= j)
            regresar  árbol [nodo];
        // Intervalo parcial
        int  izquierda = consulta (i, j, 2 * nodo
        int  = derecho de consulta (i, j, 2 * nod
        si  (a la izquierda == -1)
            regresar  árbol [nodo] = derecha;
        si  (derecha == -1)
            regresar  árbol [nodo] = izquierda;
        si  (array [left] <= array [derecha])
            regresar  árbol [nodo] = izquierda;
        regresar  árbol [nodo] = derecha;
    }
};

int  main () {
    int  A [10] = {2, 4, 3, 1, 6, 7, 8, 9, 1, 7};
    segTree t (A, 10);
    cout << "getMin (0, 4) ="  << t.query (0, 4)
    t.update (1, 0);
    cout << "getMin (0, 4) ="  << t.query (0, 4)
    t.update (0, -1);
    cout << "getMin (0, 4) ="  << t.query (0, 4)
    devolver  0;
}
```

Compartir

Relacionados

TopCoder - SRM570
En "Ad-hoc"

Introducción a los árboles binarios
En "Estructura de Datos"

Heurística Search & AI - 3
En "Algoritmos"

Etiquetas: antepasados , árbol binario , comunes , de datos , dp , dinámica , dinámica programming , lca , lowest , max , maximum , min , minimum , programming , query , range , rmq , segment , segtree , structure , tree

std :: cout <<

← 315 - Red

- 11703 Sqrt Iniciar Sin
- Introducción a la Programación Dinámica - Cortar Rods
- 272 - TEX Cotizaciones
- 10684 - El premio mayor
- gsourcecode

- 31.079 visitas

AUTORES

MGhareeb

Ad-hoc **Algoritmos** Android búsqueda binaria completado la estructura de datos de aplicaciones de escritorio Programación Dinámica Geometría Gráfica Búsqueda Heurística J2ME Matemáticas Operaciones de Investigación Problemas Utilidades

- Registro
- Iniciar sesión
- Entradas RSS
- Comentarios RSS
- Crear un sitio web gratuito o blog en WordPress.com .