ساختمان داده‌ها و الگوریتم‌ها
Data Structures and Algorithms
درخت‌ها

دانشگاه دامغان

مدرس: علی متقی

(نسخه اولیه)

ساختمان داده‌ها- مقدمه

درخت Tree

- گرافی جهت دار، بدون حلقه(چرخه)، متصل، درجه ورودی تمام گرهها غیر ریشه ۱ است

درخت(Tree)

- درخت گرافی است جهت دار، متصل، بدون دور، که درجه ورودی تمام گره بجز یکی(ریشه) برابر با ۱ است.
- مجموعه ای از عناصر که رابطه آنها رابطه پدر فرزندی است.
- اصطلاحات برگ، غیر برگ، اجداد، .....
- درخت متوازن: سطح برگها
- درخت کاملا متوازن: سطح تمام برگها با هم برابر است
- درخت کامل: درختی درجه تمامی گرههای غیر برگ با هم برابر باشد(برابر با k)
- درخت پر: درختی است که هم کامل باشد و هم کاملا متوازن باشد.
- نمونه درختهایی با خواص بالا؟؟؟

- انواع ساختمان داده ها: ایستا، پویا، نیمه پویا
- خطی، غیر خطی
- درخت طبیعی، شجره نامه ها، تاریخچه زبانها

نمایش درختان: روي كاغذ، داخل حافظه

نمایش درخت عمومی در حافظه(پیاده سازی درخت):

۱- با آرایه

۲- با لیست پیوندی با طول ثابت

۳- با لیست پیوندی با طول متغیر

۴- تبدیل درخت عمومی به درخت مرتبه ۲ فرزند چپ-هم نیای راست

۵- تبدیل درخت مرتبه ۲ حاصل از روش ۴ به درخت دودویی

نمایش درختان دودویی: ۱)آرایه ۲)لیست پیوندی

درخت kتایی

- درخت کامل مرتبه کا- رابطه تعداد گرههای برگ و غیر برگ:

$n0= (k-1)n_k + 1$, n_0,

n_k, $n= n_0+ n_k$

B: تعداد یالها

Binary Tree

تعریف، تفاوت با درختان عمومی..

مسأله: رابطه بین گرههای برگ و گرههای درجه ۲:

اثبات با استقراء

- N=1 , n_0=1, n2=0, n0=n2+1
- N=n,
- N=n+1→ , n0=n2+1
- ----------
- e: edges, n, n=e+1

پیاده سازی درخت دودویی:

۱- استفاده از آرایه: یک بعدی، اندیسها

۲- با اشاره گرها(لیست پیوندی):

```
Class btNode{
    btNode  left;
    Object   data;
    btNode  right;
};
Class Btree{
   btNode  root;
```

پیمایش درخت دودویی

- سطحی
- عمقی:
  - پیش ترتیب preorder
  - میان ترتیب inordr
  - پس ترتیب postOrder

در پیمایش درخت :

- ملاقات کلیه گرهها انجام می شود
- از ریشه درخت شروع می شود
- در هر گره باید سه کار انجام دهیم:

ملاقات گره(D, Data)

رفتن به سمت چپ L

رفتن به سمت راست R

DLR, LDR,LRD,

DRL,RDL,RLD

- preorder: a b d g e h c f
- inorder:

Recursive

- خاصیت بازگشتی درختان:
  - تعریف
  - پیمایش عمقی

- از درخت--> پیمایش(خروجی مشخص می شود)
- پیمایش(پیمایشهای) درخت را داریم--> درخت؟؟
  - آیا می توانیم از روی پیمایشهای درخت، درخت اولیه را به‌دست آوریم؟

کاربردها و انواع درخت دودویی

- درخت عبارت
- درخت heap و انواع آن
- (درخت برنده و بازنده

  برای خودتان مطالعه کنید)

- درخت جستجوی دودویی (جستجو و اضافه کردن کامل مطالعه شود
  - حذف از BST به صورت دستی کافی است)
- گراف (تعاریف کامل، نمایش در حافظه با ماتریس مجاورتی کامل، انجام پیمایشهای سطحی و عمقی به صورت دستی، مطالعه شود)

BT: پیاده سازی

```
class btnode{
   mytype  data;
   btnode  left,
                right;
}
class btree{
        btnode   root;
        btree(){
              root=null;}
        btree(mytype d){
          root=new btnode;
          root.data=d;
          root.left=null;   root.right=null;   }
}
```

پیمایش

- ملاقات گره(استفاده): D
- پیمایش چپ L
- پیمایش راست:R

DLR, DRL, LDR, LRD, RLD, RDL,

DLR:preorder, LDR: inorder, LRD:postorder

پیمایش

```
Inorder(btnode  t){
        if(t!=null){
                inorder(t.left);
                print(t.data);
                inorder(t.right);
        }
}
Inorder(root);
```

```
preorder(btnode  t){
        if(t!=null){
                print(t.data);
                preorder(t.left);
                preorder(t.right);
        }
}  NULL   null   nil
```

درخت عبارت
(Expression Tree)

- درختی است است که :
  - گرههای برگ دارای داده ساده هستند(عملوندهای ساده)
  - گرههای غیر برگ، عملگر
  - هر زیردرخت یک گره، در واقع یک عملوند از عبارت عملگر واقع در آن گره است.

درخت عبارت

۱)عبارت به صورت یک درخت نمایش داده می‌شود.

ریشه درخت، عملگر اصلی اصلی است (عملگر با کمترین اولویت ارزیابی)

عملگرهای یک عملوندی: راست

۲)ارزیابی عبارت با درخت؟

۳)پیمایشها:

Preorder→prefix

Postorder→ postfix

Inorder→infix??

Binary Search Tree(BST) درخت جستجوي دودويي

- جایگزینی برای آرایه های مرتب
- جستجو
- درج
- حذف
- مقایسه BST با آرایه (مرتب یا نامرتب)، و با لیست پیوندی

BST

- n
- h
- h~= log n
- O(h)~=O(log n)
- Insert(myData,root)

- Void insert(mytype x, ref bstNode T)

BST, BT

- تعریف
- پیاده سازی: جستجو، درج
- حذف دستی همراه با توضیح کامل روش
- تعریف درخت heap