درختها- قسمت سوم

سید مهدی وحیدی پور

با تشکر از دکتر جواد سلیمی

دانشگاه کاشان دانشکده برق و کامپیوتر

درختها

- مقدمه
  - بازنمایی درخت ها
- درخت های دودویی
- پیمایش درختهای دودویی
- عملیات دیگر روی درختهای دودویی
- درختهای دودویی نخ کشی شده
- Heap ها
- درختان جستجوی دودویی
- درختهای انتخاب
- جنگل ها
- نمایش مجموعه ها
- شمارش درخت های دودویی متمایز

عملیات دیگر روی درخت های دودویی

- کپی درخت های دودویی
- می توانیم الگوریتم پیمایش پس ترتیب درخت دودویی را اندکی تغییر دهیم تا برای کپی درخت به کار رود.

```
tree_pointer copy(tree_pointer original)
/* this function returns a tree_pointer to an exact copy
of the original tree */
{
   tree_pointer temp;
   if (original) {
      temp = (tree_pointer) malloc(sizeof(node));
      if (IS_FULL(temp)) {
         fprintf(stderr, "The memory is full\n");
         exit(1);
      }
      temp->left_child = copy(original->left_child);
      temp->right_child = copy(original->right_child);
      temp->data = original->data;
      return temp;
   }
   return NULL;
}
```

شبیه پیمایش پس ترتیب

عملیات دیگر روی درخت های دودویی

ازمایش برابری درخت های دودویی

- دو درخت دودویی را برابر (هم ارز) گویند اگر ساختار (شکل) یکسان داشته و اطلاعات داخل گره های متناظرشان نیز یکسان باشد.

V L R

```
int equal(tree_pointer first, tree_pointer second)
{
/* function returns FALSE if the binary trees first and
second are not equal, Otherwise it returns TRUE */
   return ((!first && !second) || (first && second &&
           (first->data == second->data) &&
           equal(first->left_child,second->left_child) &&
           equal(first->right_child, second->right_child))
}
```

شکل و ساختار شبیه پیمایش پیش ترتیب

تمرین

- الگوریتمی به نام swaptree بنویسید که یک درخت دودویی را دریافت کند و جای بچه های چپ و راست هر گره را عوض کند.
- الگوریتمی برای شمارش تعداد گره های برگ /گره های غیر برگ /کل گره ها در یک درخت دودویی بنویسید.
- الگوریتمی برای حذف تمام گره ها در یک درخت دودویی بنویسید.

درختهای دودویی نخ کشی شده

- نخ کشی کردن
- عیب درختهای دودویی بالا: تعداد زیادی از اشاره گرها NULL است

  تعداد گره n

  تعداد اشاره گر غیر صفر n-1

  تعداد کل اشاره گر ها 2n

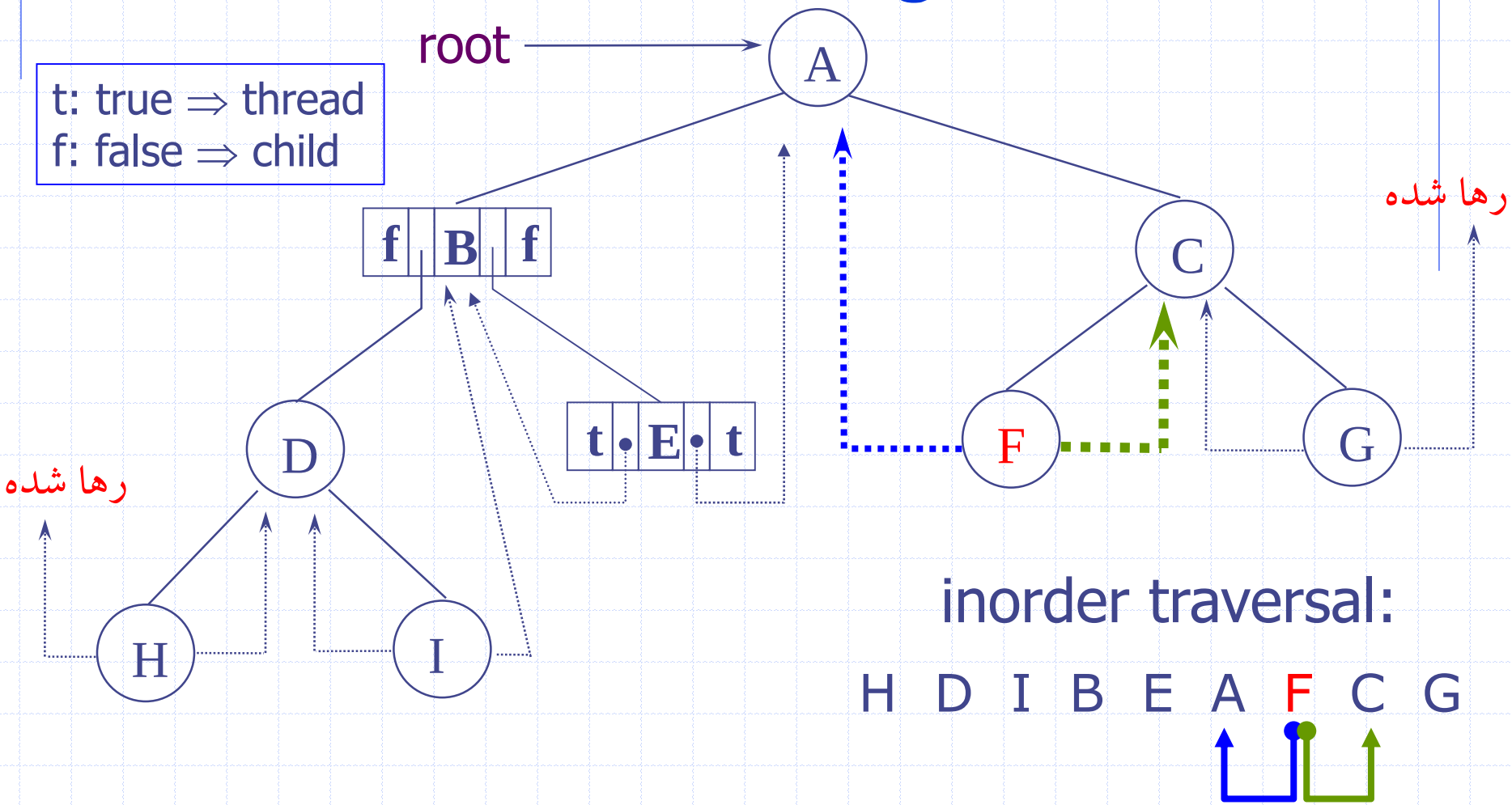
  تعداد اشاره گر صفر n+1
- راهکار : به جای اشاره گرهای صفر از این اشاره گرها برای اشاره به گره های دیگر درخت استفاده کنیم. به چنین اشاره گرهایی نخ گویند.

درختهای دودویی نخ کشی شده

- درختهای نخ کشی شده با استفاده از قوانین زیر ساخته می شوند:
  - اگر ptr->left_child صفر باشد تبدیل به اشاره گر به گره ای می شود که در پیمایش میان ترتیب بلافاصله قبل از ptr ملاقات می شود.
  - اگر ptr->right_child صفر باشد تبدیل به اشاره گر به گره ای می شود که در پیمایش میان ترتیب بلافاصله بعد از ptr ملاقات می شود.

درختهای دودویی نخ کشی شده

یک درخت دودویی نخ کشی شده



درختهای دودویی نخ کشی شده

- اضافه کردن دو فیلد boolean به نامهای left-thread و right-thread
  - اگر ptr->left-thread= *TRUE*
  - انگاه ptr->left-child حاوی نخ کشی است در غیر اینصورت حاوی یک اشاره گر به بچه چپ است.
  - همین ترتیب در مورد بچه راست هم صادق است

```
typedef struct threaded_tree *threaded_pointer;
typedef struct threaded_tree {
        short int left_thread;
        threaded_pointer left_child;
        char data;
        threaded_pointer right_child;
        short int right_thread;
        };
```

درختهای دودویی نخ کشی شده

- پیمایش میان ترتیب درخت دودویی نخ کشی شده
  - با استفاده از نخ کشیها می توان پیمایش میان ترتیب را بدون استفاده از پشته انجام داد.

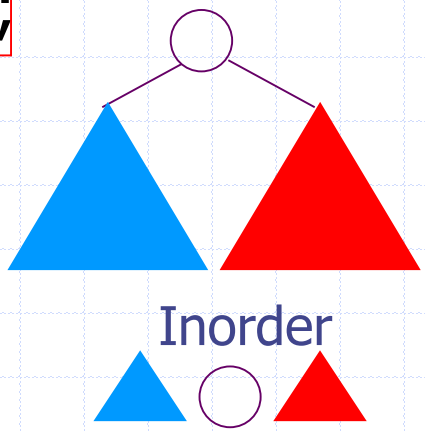
۱- اگر *ptr->right_thread = TRUE* باشد انگاه عضو بعد از *ptr* در روش میان ترتیب با توجه به تعریف نخ کشیها ptr->right_child است.

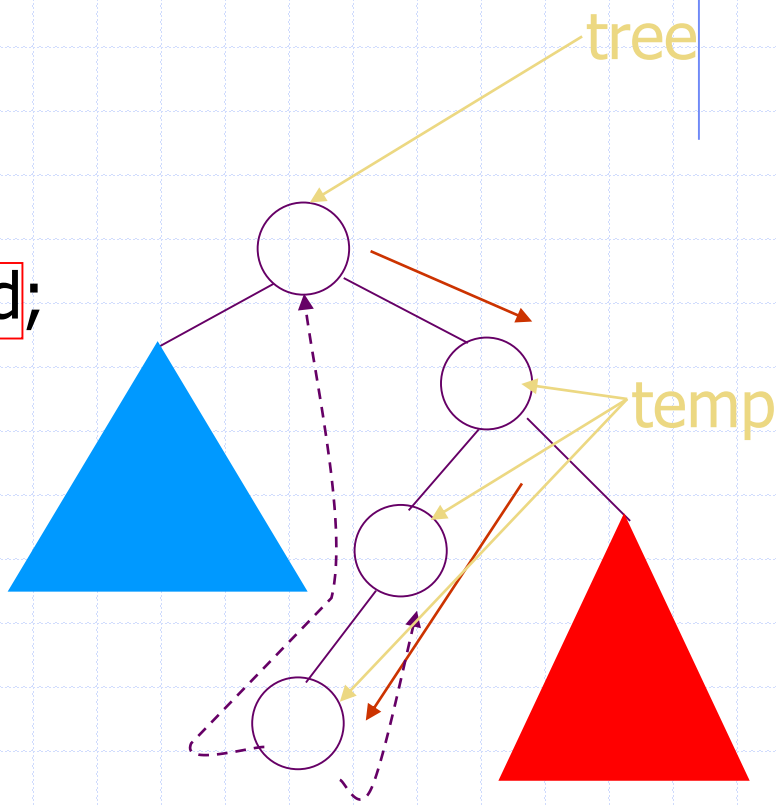
۲- در غیر اینصورت عضو بعد از *ptr* در روش میان ترتیب با دنبال کردن مسیری از اشاره گرهای بچه چپ از بچه راست ptr به دست می اید تا به گره ای با *left_thread = TRUE* برسیم.

درختهای دودویی نخ کشی شده

پیدا کردن عضو بعدی (Successor) در یک درخت دودویی نخ کشی شده

```
threaded_pointer insucc(threaded_pointer tree){
    threaded_pointer temp;
    temp = tree->right_child;
    if (!tree->right_thread)
      while (!temp->left_thread)
            temp = temp->left_child;
    return temp;
}
```



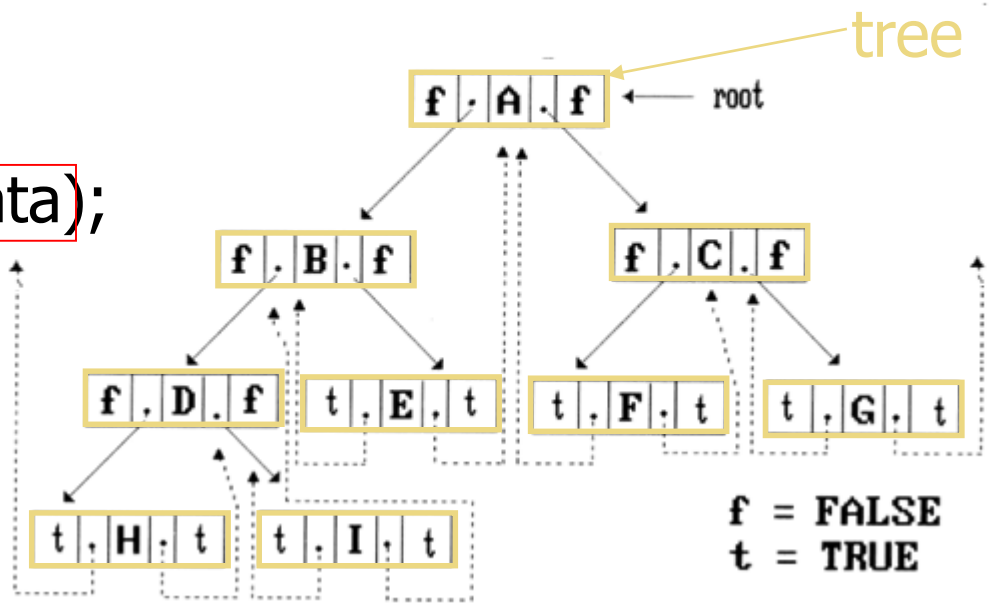


درختهای دودویی نخ کشی شده

- پیمایش میان ترتیب درخت دودویی نخ کشی شده

```
void tinorder(threaded_pointer tree){
threaded_pointer temp = tree;
   for (; !temp->left_thread; temp=temp->left-child);
   for (;;) {
      temp = insucc(temp);
      if (temp==null)
            break;
      printf("%3c",temp->data);
   }
}
```

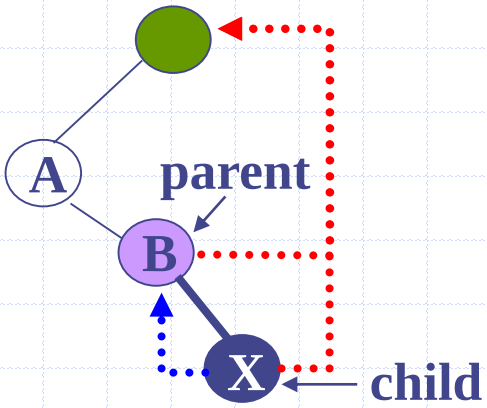
output: H D I B E A F C G



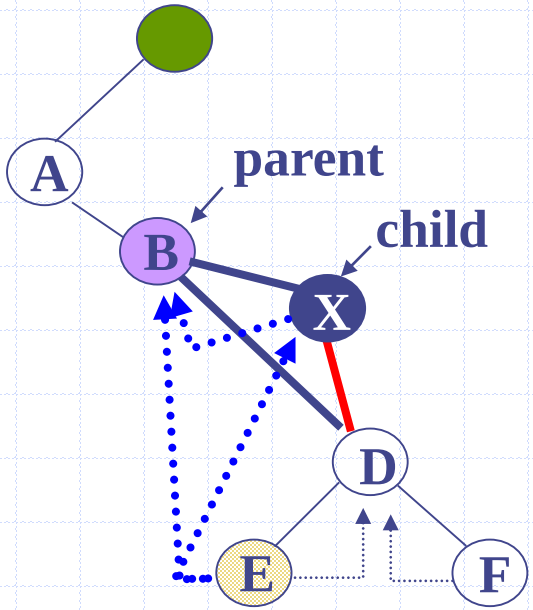
Time Complexity: O(*n*)

درختهای دودویی نخ کشی شده

- اضافه کردن گره به درخت دودویی نخ کشی شده
  - اضافه کردن یک گره به عنوان بچه راست یک گره والد



حالت اول: زیر درخت راست B خالی باشد



حالت دوم:زیر درخت راست Bخالی نباشد

در حالت دوم: عنصر بعدی x در پیمایش میان ترتیب را پیدا کرده و عنصر قبل از آن را x قرار می دهیم

درختهای دودویی نخ کشی شده

- اضافه کردن گره به صورت بچه راست والد در یک درخت دودویی نخ کشی شده

```
void insert_right(thread_pointer parent, threaded_pointer child){
/* insert child as the right child of
parent in a threaded binary tree */
    threaded_pointer temp;
    child->right_child = parent->right_child;
    child->right_thread = parent->right_thread;
    child->left_child = parent;
    child->left_thread = TRUE;
    parent->right_child = child;
    parent->right_thread = FALSE;
    If(!child->right_thread){
      temp = insucc(child);
      temp->left_child = child;
    }
}
```

حالحتلدولول