دانشکده علوم ریاضی و آمار

مدرس: دکتر مجتبی رفیعی نیمسال اول ۱۴۰۰-۱۴۰۱

ساختمان داده‌ها و الگوریتم‌ها

جلسه ۳۴

نگارنده: راضیه نظری

۲۵ آذر ۱۴۰۰

فهرست مطالب

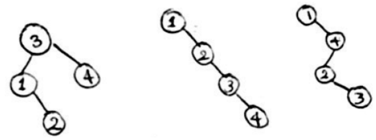
۱ درخت دودویی جست‌وجو (Binary Search tree)

یک درخت دودویی ریشه‌دار است که نودهای داخلی آن دارای خصیصه زیر هستند:

1) از تمامی نودهای زیر درخت سمت چپ خود بزرگتر یا مساوی هستند،

2) از تمامی نودهای زیردرخت سمت راست خود کوچکتر هستند.

مثال ۱



کاربرد اصلی BTS: جستجوی یک مقدار در مجموعه اعداد موجود می‌باشد. الگوریتم جستجو در چنین داده ساختاری به‌صورت زیر می‌باشد:

BST-Search (X) key

```
if (T=Null or X. key=key) then
      return X;
if (key<x.key) then
      return BST-Search(x. left, key)
else
      return BST-Search(x.right, key)
```

نکته: یک راه حل ساده برای جستجوی یک مقدار در یک مجموعه، پوشش کل مجموعه و تست برابری است که در این حالت برای یک مجموعه n-عضوی، پیچیدگی زمانی خطی $O(n)$ را خواهیم داشت. پس در استفاده از داده ساختار BTS دنبال آن هستیم که تا حد امکان پیچیدگی زمانی کمتری داشته باشیم.

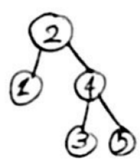
۲ بررسی برخی ویژگی‌های داده ساختار BTS

1) حداکثر ارتفاع درخت برای n عنصر در BST، $n-1$ است که بیانگر جستجویی با پیچیدگی خطی $O(n)$ دارد.

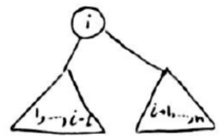
2) حداقل ارتفاع درخت برای n عنصر در BTS، از مرتبه $O(\log n)$است که نوید از جست‌وجوی بهینه با پیچیدگی $O(\log n)$ دارد.

3) اگر اسکلت یک درخت دودویی داده شده باشد و یک توالی از اعداد متمایز در اختیار داشته یاشیم، می‌توانیم به‌صورت یکتا این توالی را در درخت قرار دهیم به نحویکه درخت BST حاصل شود.

مثال ۲ *توالی* $1, 2, 3, 4, 5$ *را در اسکلت درختی زیر جای‌دهی کنید به نحو یکه درخت* BST *حاصل می‌شود.*



4) با توجه به بند 3 از پیش مشخص باشد پس می‌توان درخت BST بهینه‌ای (با ارتفاع $O(\log n)$) تشکیل داد و همواره جستجو در آن از مرتبه $O(\log n)$ خواهد بود. امّا از آنجاییکه ما با مجموعه‌های پویا طرف هستیم و عناصر حذف و اضافه می‌شود، پیشنهاد مناسبی نیست.



سوال۱: تعداد اسکلت های متمایز مربوز به درخت‌های جستجوی دودویی با n گره چه میزان است.
فرض کنید توالی $1,\ldots,n$ عنصر متمایز داریم، هرکدام از این عناصر می‌تواند ریشه باشد، فرض کنید i ریشه باشد.

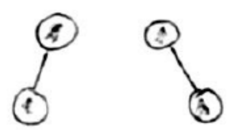
$$\begin{aligned} T(n) &= \sum_{i=1}^{n} T(i-1) + T(n-i) \\ T(n) &= 1 \end{aligned}$$

که همان رابطه بازگشتی مربوط به عددد کاتالان است و پاسخ:

$$T(n) = \frac{1}{n+1}\binom{2n}{n}$$

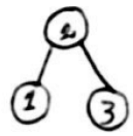
مثال ۳ *برای* $n = 2$ *داریم:*

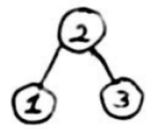
$$\frac{1}{n+1}\binom{2n}{n} = \frac{1}{3}\binom{4}{2} = \frac{1}{\cancel{3}} \times \frac{4\times \cancel{3}}{2} = 2$$



سوال ۲: اگر ما n عنصر متمایز a_1 تا a_n داشته باشیم، ما $n!$ درخت BST با ارتفاع‌های مختلف خواهیم داشت؟ بله، چرا که $n!$ جایگشت از توالی می‌توان داشت و با درج آنها به‌ترتیب برای ساخت درخت BST، $n!$ درخت برچسپ‌دار متمایز می‌شود.
جمع‌بندی: تعدا درخت‌های BST حاصل جایگشت توالی به‌وضوح بیشتر از تعداد اسکلت‌های درخت BTS است چرا که برخیاز درخت‌های حاصل از جایگشت دارای اسکلت یکسان هستند.

مثال ۴ *برای دو حالت* $2,1,3$ *و* $2,3,1$ *مشاهده می‌کنید.*



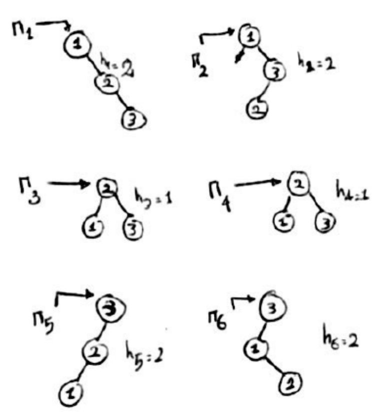


قضیه ۱ *اگر یک توالی از عناصر متمایز داشته باشیم و یک جایگشت تصادفی از آنها را در نظر بگیریم به‌طور متوسط انتظار می‌رود که درخت* BST *حاصل دارای ارتفاع* $O(\log n)$ *باشد. به‌عبارت دیگر:*

$$\begin{aligned}
&\pi_1 \longrightarrow h_1 \\
&\pi_2 \longrightarrow h_2 \qquad X_n = \frac{1}{n!}\sum_{i=1}^{n!} h_i = O(\log n) \\
&\vdots \\
&\pi_{n!} \longrightarrow h_{n!}
\end{aligned}$$

مثال ۵ *برای* $1,2,3$ *داریم:*

$$\pi_1 = 1,2,3 \quad \pi_2 = 1,3,2 \quad \pi_3 = 2,1,3 \quad \pi_4 = 2,3,1 \quad \pi_5 = 3,2,1 \quad \pi_6 = 3,1,2$$



$$X_3 = \frac{1}{6}\sum_{i=1}^{3} h_i = \frac{10}{6}$$

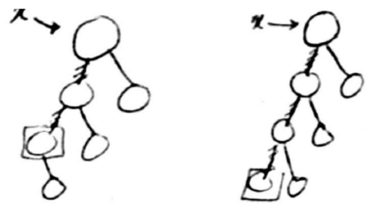
نکته: اگر یک توالی از قبل داشته باشیم و بخواهیم درخت BST را بسازیم، می‌توان از ایده‌های زیر استفاده کرد:

1) عناصر را مرتب کنیم و پس براساس آن یک درخت BST بسازیم که درخت متعادل حاصل شود، د راین حالت پیچیدگی زمانی $O(n \log n)$ خواهد بود (به سبب مرتب‌سازی).

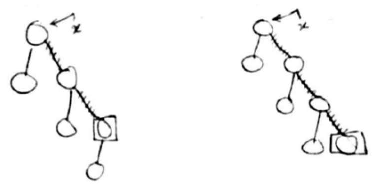
2) یک جایگشت تصادفی روی توالی اعمال کنیم و سپس درخت را بسازیم که طبق قضیه قبل به احتمال خوبی متعادل است و اگر نبود مجدداً تکرار می‌کنیم. از آنجاییکه درج هر نود در درخت در بدترین حالت نیاز به جستجویی از مرتبه ($O(\log n)$) دارد، و n برای درج و ساخت درخت دودویی جستجو نیاز داریم، پیچیدگی زمانی از مرتبه $O(n \log n)$ خواهد بود.

نکته: در عمل ورودی‌ها به تدریج می‌آیند و توالی را از پیش نداریم و ضمانت بر توالی رندم داشتن هم از ورودی فرض قوی است. بنابراین نیازمند داده ساختارهای پیچیده‌تری برای این منظور هستیم که در جلسات آتی آنها را شرح می‌دهیم (مثل $\mathrm{Rad} - \mathrm{Block\ tree}$ و Arl tree).

تمرین۱. فرض کنید یک نود خاص در درخت BST مثل x داده شده است، کوچکترین عدد در زیردرخت‌های x را چگونه پیدا می‌کنیم.
همیشه حرکت به چپ داریم.



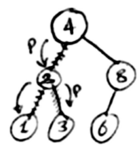
تمرین۲. فرض کنید یک نود خاص مثل x در درخت BST داده شده است، بزرگترین عدد در زیردرخت‌های x را چگونه پیدا می‌کنیم.
همیشه حرکت به راست داریم.



تمرین ۳. فرض کنید یک نود خاص مثل x در درخت BST داده شده است، عنصر بعدی عنصر x در توالی مرتب (Successor) را چگونه پیدا می‌کنیم.

$$\mathrm{Successor}(4) = 6 \qquad \mathrm{Successor}(2) = 3$$

تمرین ۴. فرض کنید یک نود خاص مثل x در درخت BST داده شده است، عنصر قبلی عنصر x در توالی مرتب (Predecessor) را چگونه پیدا می‌کنیم.



$$\text{Predecessor}(4) = 3 \qquad \text{Predecessor}(2) = 3$$

تمرین ۵. نحوه پیدا کردن عنصر کمینه و بیشینه در درخت BST به چه نحو است.
تمرین ۶. پیمایش میانوندی درخت BST چه توالی را تولید می‌کند؟
تمرین ۷. قیاس بین جستوجوی در یک آرایه مرتب و یک درخت BST انجام شود.
و دیگر عملیات تشریح شده