ساختمان داده‌ها

پیمایش‌های درخت:

هدف از پیمایش درخت ارائه یک نمایش خطی از ساختار غیر خطی درخت است به طوری که تمام عناصر درخت دقیقاً یک بار در آن حضور داشته باشند.

۱- روش پیش ترتیب (Preorder)

این روش به صورت بازگشتی تعریف می‌شود: اگر یک گره داشته باشیم پیمایش Preorder آن برابر همان گره است. اگر T_{kp}, ..., T_{3p},T_{2p},T_{1p} پیمایش‌های Preorder مربوط به زیر درخت‌های T_1, T_2 تا T_k باشند، پیمایش Preorder به صورت زیر است:

r T_{1p} T_{2p} T_{3p} ... $T_{k\,p}$

۲- روش پس ترتیب (postorder)

این روش نیز به صورت بازگشتی تعریف می‌شود: اگر یک گره داشته باشیم پیمایش postorder آن برابر همان گره است. اگر T_{kpo},...,T_{2po},T_{1po} پیمایش‌های مربوط به زیر درخت‌های T_1, T_2, ..., T_k باشند پیمایش postorder درخت T به صورت روبرو است:

T_{1po} T_{2po} .... T_{kpo} r

۳- روش بین ترتیب (inorder)

مشابه روش‌های قبل این روش نیز به صورت بازگشتی تعریف می‌شود: اگر یک گره داشته باشیم پیمایش inorder آن برابر همان گره است. اگر T_{kin}, ..., T_{2in}, T_{1in} پیمایش‌های inorder مربوط به زیر درخت‌های T_1, T_2,..., T_k باشند پیمایش inorder درخت به صورت روبرو است: T_{1in}r T_{2in} ... T_{kin}

درخت عبارت یک درخت دودویی است که برگ‌های آن شامل عملوندها و گره‌های غیر برگ آن شامل عملگرهای یک عبارت است. پیمایش Preorder، postorder و inorder یک درخت عبارت به ترتیب معادل prefix، postfix و infix آن عبارت است.

نکته: در پیمایش inorder آخرین گره سمت چپ در اول و آخرین گره سمت راست در آخر ظاهر می‌شود.
در پیمایش preorder ریشه درخت در ابتدا و آخرین گره سمت راست در آخر ظاهر می‌شود.
در پیمایش postorder ریشه در آخر و آخرین گره سمت چپ در اول ظاهر می‌شود.

۴- روش سطر ترتیب (level order)

در این روش عناصر هم عمق درخت در یک سطر قرار می‌گیرند و از پایین‌ترین سطح (سطح با شماره کمتر) به ترتیب عناصر از چپ به راست نوشته می‌شوند.

۵- روش DFS (اول عمق)

این روش مبتنی‌بر stack است. پیمایش در این روش همواره از ریشه شروع می‌شود و سپس با توجه به معیار اولویتی تعریف شده یک عنصر مشاهده نشده متصل به گره جاری را انتخاب می‌کنیم. اگر این عنصر وجود نداشت به آخرین عنصر مشاهده شده بازگشت می‌کنیم و الگوریتم را از این گره دنبال می‌کنیم.

نکته: preorder همان روش DFS است در صورتی که اولویت عناصر با چپ‌ترین عنصر باشد.

۶- BFS (اول پهنا)

در این روش پیمایش همواره از ریشه شروع می‌شود و در هر مرحله تمام گره‌های متصل به گره جاری به ترتیب اولویت مشاهده می‌شوند. گره جاری بعدی براساس اولویت بین گره‌ها انتخاب می‌شود. این روش مبتنی‌بر ساختمان داده صف است.

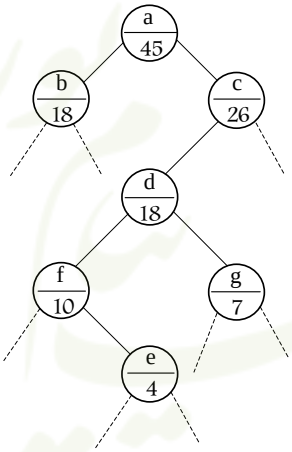
ترسیم درخت به کمک پیمایش‌های آن:

برای درخت‌های دودویی (عام) با داشتن تنها یک پیمایش از درخت بازسازی درخت به صورت یکتا امکان‌پذیر نیست.

اگر پیمایش‌های inorder و preorder یک درخت دودویی را داشته باشیم، در حالتی که برچسب گره‌های درخت یکتا باشد (درخت عنصر تکراری نداشته باشد) ترسیم درخت به صورت یکتا امکان‌پذیر است.

هم چنین با فرض یکتا بودن برچسب گره‌های درخت و داشتن پیمایش‌های inorder و postorder از یک درخت دودویی می‌توان درخت را به صورت یکتا ترسیم نمود.

مثال) در شکل زیر قسمتی از یک درخت دودویی نشان داده شده است. در زیر برچسب هر گره عددی نوشته شده است که تعداد کل گره‌های موجود در زیر درخت آن گره را نشان می‌دهد (با احتساب خود آن گره). اگر درخت مفروض را به طور inorder پیمایش کنیم f چندمین خروجی خواهد بود؟ (علوم کامپیوتر ۸۳)



۱) ۲۵ امین
۲) ۲۶ امین
۳) ۲۹ امین
۴) ۳۰ امین

حل گزینه ۱. ابتدا زیر درخت سمت چپ ریشه که دارای ۱۸ گره است پیمایش می‌شود. سپس ریشه پیمایش می‌شود. بعد زیر درخت سمت چپ f که دارای ۵ گره است (۵ =۱-۴-۱۰) پس تا اینجا ۲۴=۵+۱+۱۸ گره پیمایش شده و f گره ۲۵ام است.

زیر درخت راست f
خود f

مثال) اگر acdfbeg پیمایش preorder یک درخت دودویی باشد، کدام یک از دنباله‌های زیر نمی‌تواند پیمایش inorder آن درخت باشد؟ (علوم کامپیوتر ۸۱)

۱) cdbfage
۲) cabfged
۳) fdecbag
۴) fdbcage

حل گزینه ۳.

مثال) شرط اصلی برای این که بتوان از روی دو پیمایش داده شده یک درخت باینری، درخت اصلی را ساخت چیست؟ (علوم کامپیوتر ۸٤)

۱) تمام داده‌ها در درخت متمایز باشند.

۲) فقط ریشه از بقیه متمایز باشد

۳) داده‌های زیر درخت چپ ریشه از داده‌های زیر درخت راست متمایز باشند ولی در خود زیر درخت‌های چپ و راست می‌تواند داده تکراری وجود داشته باشد.

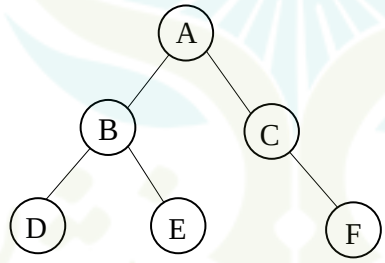
٤) داده‌های موجود در برگ‌های درخت متمایز باشند.

حل گزینه ۱

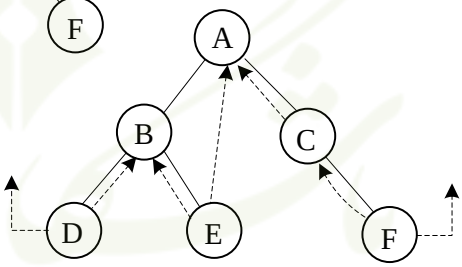
نخ‌کشی درخت:

در یک درخت دودویی با n گره، ۲n اشاره گر وجود دارد که n+1 تای آنها تهی است. با استفاده از این اشاره‌گرها می‌توان پیمایش‌های مختلف را سریع‌تر انجام داد. مثلاً در صورت نخ‌کشی preorder درخت، اشاره‌گرهای null مشخص کننده عناصر بعدی یا قبلی در پیمایش preorder درخت هستند. برای تشخیص نخ بودن اشاره‌گرها از دو بیت (چپ و راست) استفاده می‌کنیم.

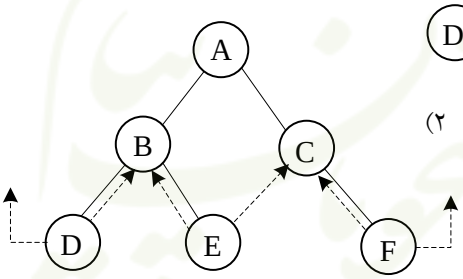
مثال) درخت زیر را در نظر بگیرید. می‌خواهیم این درخت را برای پیمایش inorder نخی (threaded) کنیم. گزینه صحیح را انتخاب کنید (علوم کامپیوتر ۷۹)



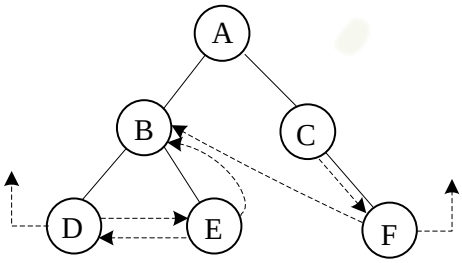
۱)



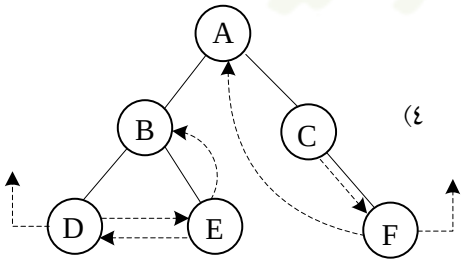
۲)



۳)



٤)



حل گزینه ۱. پیمایش inorder درخت به صورت DBEACF است.

فرهنگ داده‌ای جستجو:

فرهنگ داده‌ای جستجو ساختمان داده‌ای است که اعمال جستجو، درج، و حذف را بر روی مجموعه‌ای از داده‌ها به صورت بهینه انجام می‌دهد. برای پیاده‌سازی آن می‌توان از درخت دودویی جستجو (Binary Search Tree)، درخت AVL، درخت ۳-۲، درخت B و جداول درهم (hash) استفاده کرد.

درخت دودویی جستجو: از نظر ساختمان داده‌ها هیچ تفاوتی با درخت دودویی معمولی ندارد. (البته BST می‌تواند تهی باشد) ویژگی این درخت این است که در آن مقدار هر گره بزرگ‌تر از هر مقدار در زیر درخت چپ و کوچک‌تر از هر مقدار در زیر درخت راست آن است. هر گره دارای یک کلید است و دو گره نباید دارای کلیدهای یکسان باشند. (کلیدها منحصر بفرد هستند)

نکته مهم: پیمایش inorder درخت BST، عناصر درخت را به صورت مرتب شده صعودی به دست می‌دهد و برعکس.

تعریف BST بازگشتی است ⇐ هر زیر درخت BST خود یک BST است.

تعداد BST هایی که می‌توان با n کلید ساخت برابر است با عدد کاتالان: $\frac{\binom{2n}{n}}{n+1}$

جستجو در BST در بدترین حالت (درخت به صورت اریب باشد) O(n) و در بهترین حالت و حالت متوسط O(lgn) است.

$$T(n)=\begin{cases}1+T(n-1) & \text{بدترین حالت}\\ 1+T(n/2) & \text{بهترین حالت و حالت متوسط}\end{cases}$$

نکته: چنانچه مجموعه‌ای از عناصر موجود باشد برای ساختن بهترین BST، باید عنصر میانه را به صورت بازگشتی به عنوان ریشه قرار دهیم.

الگوریتم درج در BST مشابه جستجو است. در BST همواره عنصر تازه وارد را به عنوان برگ به ساختار اضافه می‌کنیم.

مرتبه الگورتیم درج و حذف نیز مشابه جستجو است. هر سه الگوریتم مرتبه O(h) دارند که h ارتفاع درخت است.

درخت AVL: یک BST است که در آن حداکثر اختلاف ارتفاع دو زیر درخت چپ و راست هر گره یک باشد. جستجو در این درخت همواره O(lgn) است و الگوریتم جستجوی آن مشابه الگوریتم جستجوی BST است. اما الگوریتم‌های حذف و درج آن کاملاً متفاوت هستند چرا که ممکن است حذف یا درج یک گره باعث تغییر کلی درخت یا حتی تعویض ریشه شود. در این درخت کمترین تعداد عناصر از رابطه بازگشتی T(h)= T(h-1)+ T(h-2)+ 1 و بیشترین تعداد عناصر از رابطه بازگشتی T(h)= 2T(h-1)+ 1 به دست می‌آید.

درخت ۳-۲: درختی است که هر گره آن ۲ یا ۳ فرزند دارد. حالت‌های مختلف آن بررسی شود.

جداول درهم: هدف از این جداول ارائه ساختمان داده‌ای است که بتواند جستجو O(1) انجام دهد.

منظور از درهم سازی تبدیل کلید به آدرس (key to Address Transformation) توسط تابع درهم ساز (hashing function) است. تابع درهم‌ساز برای مقدار x جای آن را مشخص می‌کند. (h(x)) معمولاً تعداد خانه‌های جدول کمتر از تعداد عناصر است و به همین دلیل تابع h یک به یک نیست و پدیده تصادم رخ می‌دهد.

در open hashing هر سلول از جدول hash در حقیقت یک لیست پیوندی است. جستجو در بهترین حالت و بدترین حالت به ترتیب $O\left(\frac{n}{B}\right)$ (توزیع یکنواخت تابع) و O(n) (نگاشت تمام عناصر به یک خانه) است که در آن B نشان دهنده تعداد سلول‌های جدول است.

در close hashing تعداد داده‌ها محدود است.

ساختمان داده heap: heap یک درخت دودویی نیمه مرتب متوازن است.

در max heap مقدار کلید هر گره بزرگ‌تر یا مساوی کلیدهای فرزندانش است و در min heap مقدار کلید هر گره کوچک‌تر یا مساوی کلیدهای فرزندانش است.

واضح است که در max heap ماکزیمم در ریشه و مینیمم در یک برگ قرار دارد و در min heap مینیمم در ریشه و ماکزیمم در یک برگ قرار دارد.

در پیاده‌سازی heap با آرایه، ریشه در خانه اول قرار می‌گیرد، اگر عنصری در خانه i قرار داشته باشد فرزندانش در 2i (فرزند چپ) و 2i+1

(فرزند راست) قرار می‌گیرند چنانچه عنصری در خانه j قرار داشته باشد پدر آن در $\left\lfloor \frac{j}{2} \right\rfloor$ قرار می‌گیرد.

نتیجه: برگ‌های یک heap همواره در اندیس‌های $\left\lceil \frac{n}{2} \right\rceil$ به بعد قرار می‌گیرند.

در الگوریتم درج در heap عنصر جدید همواره به عنوان آخرین برگ اضافه می‌شود و سپس با bubble up ساختار heap بازسازی مجدد می‌شود bubble up از O(lgn) است.

به طور مشابه برای حذف از الگوریتم bubble down از O(lgn) استفاده می‌شود.

تبدیل یک آرایه به heap با O(n) امکان‌پذیر است.

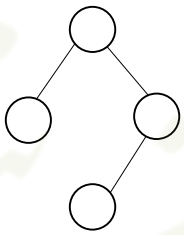
با استفاده از heap می‌توان عناصر را با مرتبه nlgn مرتب نمود. (الگوریتمی از O(nlgn) برای مرتب سازی با استفاده از heap وجود دارد).

مثال) برای یک درخت جستجوی باینری که در گره‌های آن تعدادی عدد ذخیره شده است، با داشتن پیمایش preorder آن کدام جمله صحیح است؟ (علوم کامپیوتر ۸۴)

۱) فقط با یک پیمایش نمی‌توان درخت ساخت.
۲) می‌توان درخت را در $O(n^2)$ ساخت.
۳) می‌توان درخت را در O(nlgn) ساخت.
۴) می‌توان درخت را در O(n) ساخت.

حل ✓ گزینه ۴. همان طور که گفته شد پیمایش inorder یک BST همواره در دسترس است (مرتب شده عناصر به ترتیب صعودی)

مثال) چند حالت عناصر با کلیدهای a< b< c< d را می‌توان وارد یک درخت دودویی جستجوی تهی کرد که درختی به شکل زیر به دست آید؟ (سراسری ۸۴)



۱) ۴ ۲) ۳ ۳) ۲ ۴) ۱

حل ✓ گزینه ۲. سه حالت وجود دارد.

مثال) در یک درخت جستجوی دودویی با n گره، تعداد مقایسه برای جستجوی یک، عنصر حداکثر برابر است با: (علوم کامپیوتر ۸۴)

۱) $O\left(\frac{n}{2}\right)$ ۲) O(n) ۳) $O\left(\sqrt{2}\right)$ ۴) O(lgn)

حل ✓ گزینه ۲. در بدترین حالت ارتفاع درخت n است.

مثال) سطح- ترتیب (level- order) یک درخت، عناصر درخت را به ترتیب سطح (عمق) آن عناصر و در هر سطح از چپ به راست بررسی می‌کند. فرض کنید که دنباله‌های سطح- ترتیب و میان ترتیب (inorder) یک درخت دودویی جستجو داده شده‌اند. کدام یک از گزینه‌های زیر صحیح است؟ (سراسری ۸۳)

۱) تنها با دنباله سطح- ترتیب می‌توان درخت را ساخت، ولی درخت یکتانیست.
۲) تنها با دنباله سطح- ترتیب می‌توان درخت را به صورت یکتا ساخت.
۳) با سطح- ترتیب و میان ترتیب می‌توان درخت را ساخت، ولی درخت یکتانیست.
۴) با سطح- ترتیب و میان ترتیب می‌توان درخت را ساخت و درخت یکتا است.

حل ✓ گزینه ۲.

مثال) کدام یک از گزینه‌های زیر در ارتباط با هر درخت دودویی جستجو با n عنصر غلط است؟ (سراسری ۷٤)

۱) در حذف تعدادی از عناصر درخت، ترتیب حذف تأثیری در درخت حاصل ندارد.

۲) متوسط ارتفاع کلیه درخت‌های دودویی جستجو با n المان متناسب است با $\log_2^n$.

۳) اگر n عنصر را از قبل داشته باشیم می توان یک درخت دودویی با ارتفاع متناسب با $\log_2^n$ ایجاد کرد.

٤) می‌توان کوچک‌ترین عنصر را در این درخت با مرتبه O(lgn) حذف کرد.

حل گزینه ۲.

مثال) ماکزیمم تعداد مقایسه برای min- heap کردن یک max- heap با n گره برابر است با: (علوم کامپیوتر ۸٤)

۱) O(n+ lgn)　　۲) O(lgn)　　۳) O(n lgn)　　٤) O(2n)

حل گزینه ٤. الگوریتم آن چیست؟

مثال) تابع زیر چک می‌کند که آیا یک درخت دودویی با عناصر صحیح (int) و متمایز و باریشه root یک درخت جستجوی دودویی است یا خیر. کدام گزینه در جای خالی D, C, B, A باید قرار داده شود؟ (مهندسی IT ۸٤)

```
Bool IsBST (tree *t)
{
Return IsBST(t, MININT, MAX INT) ;
}
Int IsBST(tree *t, int min, int max)
{
if (t= = Null) return True;
if (t→ data<min || t→ data>max) return False;
Return IsBST(t→ left child, ... A..., ...B...) && IsBST(t→ Right child,..C.., ...D...);
}
```

۱) $A = t \rightarrow data,\ B = max,\ C = min,\ D = t \rightarrow data + ۱$

۲) $A = max,\ B = t \rightarrow data,\ C = t \rightarrow data + ۱,\ D = min$

۳) $A = t \rightarrow data,\ B = min,\ C = max,\ D = t \rightarrow data + ۱$

۴) $A = min,\ B = t \rightarrow data,\ C = t \rightarrow data + ۱,\ D = max$

حل گزینه ٤.