درس ساختمان داده‌ها
دانشکده ریاضی. دانشگاه صنعتی خواجه نصیرالدین طوسی — بهار ۹۹
تدریس توسط: حسین جوهری

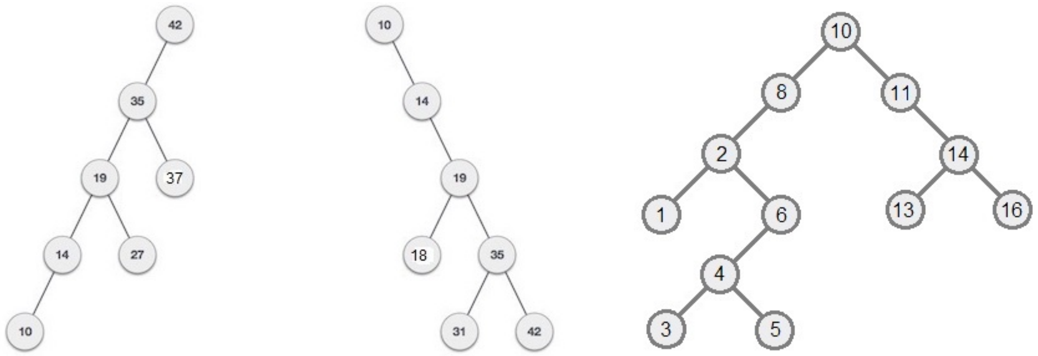
درخت AVL

درخت AVL یک درخت باینری است که بر اساس متوازن سازی درخت BST کار می‌کند. در واقع درخت AVL یک نوع درخت BST است که حالت متوازن داشته و ارتفاع آن در حد $O(\log n)$ است. همانطور که قبلا دیدیم، زمان اجرای اعمال اصلی در ساختار داده BST (مانند جستجو، درج و حذف) متناسب با ارتفاع درخت است. هر چه ارتفاع درخت کمتر باشد، اعمال اصلی در BST سریعتر اجرا می‌شوند. درخت AVL سعی می‌کند با متوازن سازی ارتفاع درخت را در حد $O(\log n)$ نگه دارد.

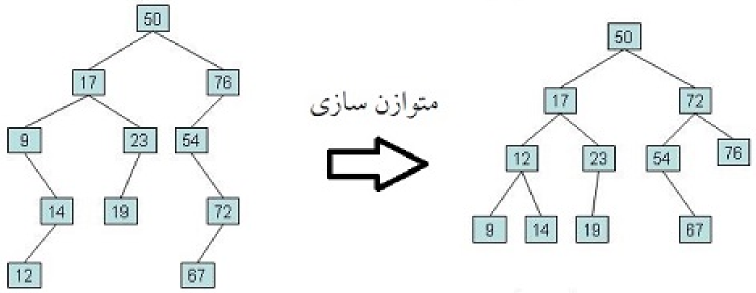
در این جلسه ابتدا به معرفی دو عملگر ساده (دوران rotation و بازسازی سه‌گانه trinode reconstruction) می‌پردازیم که با انجام آنها درخت BST متوازن تر می‌شود و ارتفاع آن می‌تواند کاهش پیدا کند. سپس درخت AVL را معرفی خواهیم کرد و نشان خواهیم داد که ارتفاع یک درخت AVL همواره $O(\log n)$ است.

۱ متوازن سازی BST

پس از یک انجام یک سری عمل حذف و اضافه روی یک BST ، ممکن است درخت بصورت کاملا نامتوازن دربیاید و حالت باریک و مورب پیدا کند. شکل پایین چند نمونه از درختهای BST نامتوازن را نشان می‌دهد.

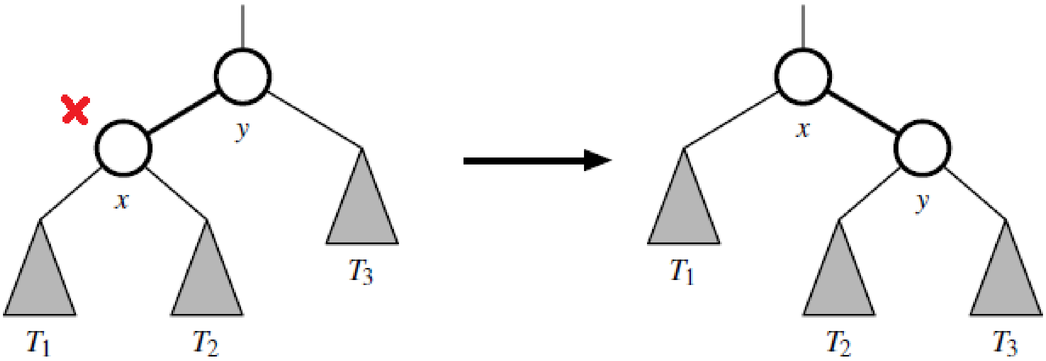


برای کاهش ارتفاع یک BST نامتوازن می‌توان با انجام یک سری جابجایی درختی بدست آورد که همان کلیدها را در خود ذخیره کرده است و نظم و قاعده BST هم در آن برقرار است و در عین حال ارتفاعش کاهش پیدا کرده است. در شکل زیر یک درخت BST در سمت چپ نشان داده شده است که بعد از انجام یک سری جابجایی به حالت متوازن در آمده است (درخت سمت راست.)



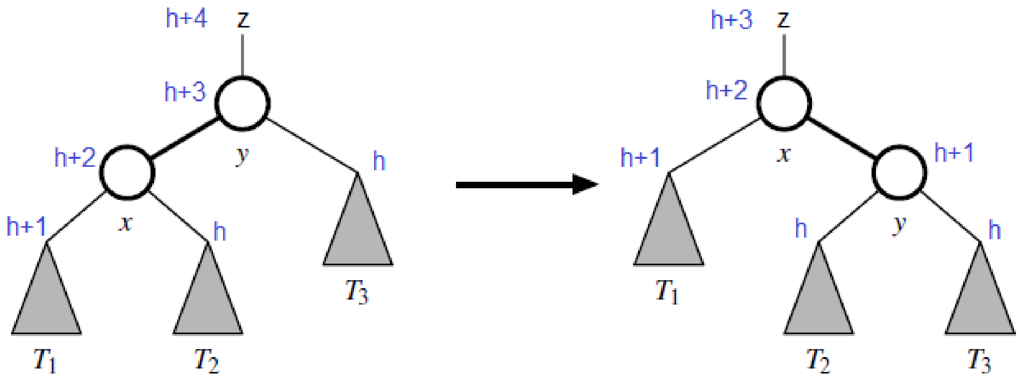
۱.۱ دَوَران rotation

دَوَران یک عمل ساده است که با انجام دو جابجایی زیردرختهای یک BST سعی می‌کند ارتفاع یک زیردرخت را کاهش دهد. به شکل زیر توجه کنید.

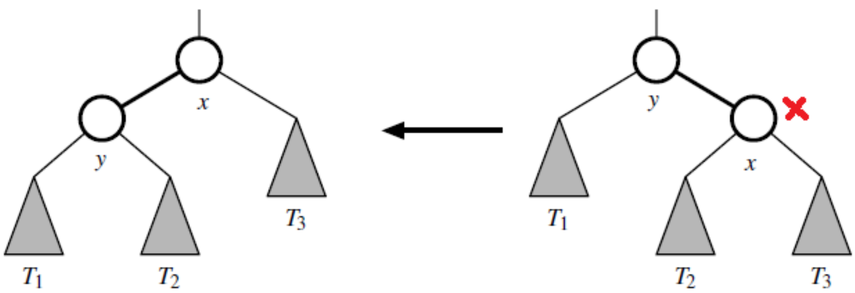


دوران روی راس x انجام می‌شود. راسی که دوران روی آن انجام می‌شود باید پدر داشته باشد (راس y). دقت کنید بعد از دوران راس y که پدر بوده حالا فرزند x می‌شود و زیردرخت T_2 به عنوان فرزند چپ به آن داده می‌شود. از طرف دیگر جابجاییها نظم و قانون حاکم بر BST را تغییر نمی‌دهند.

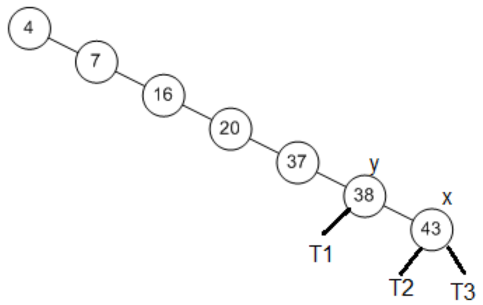
حال این کار چه نفعی دارد؟ فرض کنید قبل از دوران، پدر y راس z باشد. اگر زیردرخت T_1 بیشترین ارتفاع را داشته باشد، بعد از انجام دوران، یک واحد از ارتفاع z کم می‌شود. دقت کنید انجام دوران هیچ وقت باعث افزایش ارتفاع z نمی‌شود. در شکل زیر یک نمونه از کاهش ارتفاع نشان داده شده است. ارتفاع ها را به رنگ آبی در محل زیردرخت مربوطه نوشته‌ایم.

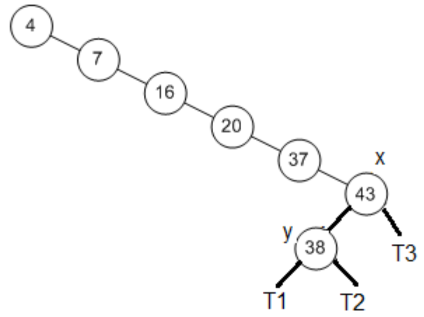


دوران می‌تواند در جهت دیگر (از سمت راست به چپ) نیز انجام شود. به شکل زیر توجه کنید. دوران روی راس x انجام شده است.



یک مثال برای دوران شکل زیر یک درخت کاملا مورب را نشان می‌دهد. عمل دوران روی راس x انجام شده است. دقت کنید زیردرختان T_1 و T_2 و T_3 در این مثال تهی هستند.

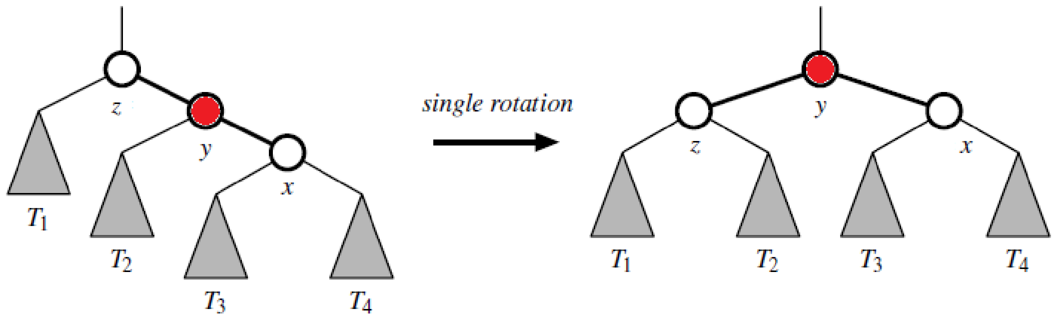


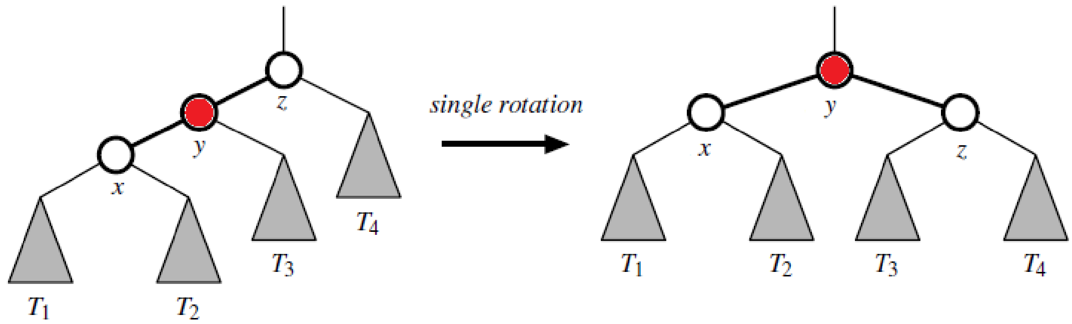


۲.۱ بازسازی سه‌گانه

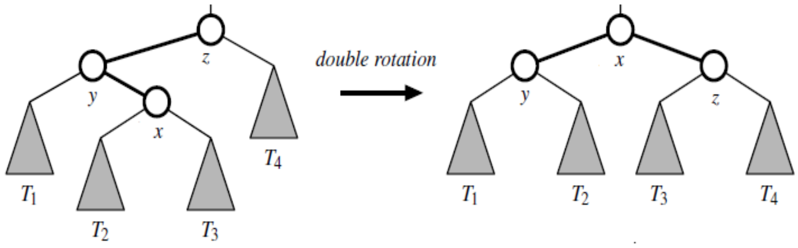
عمل بازسازی سه‌گانه trinode reconstruction روی سه راس درخت BST انجام می‌شود و شامل یک دَوَران و یا یک ترکیبی از دو دَوَران است. اینکه یک دوران انجام می‌شود یا اینکه به دو دوران نیاز است بستگی به موقعیت سه راس مورد نظر دارد.

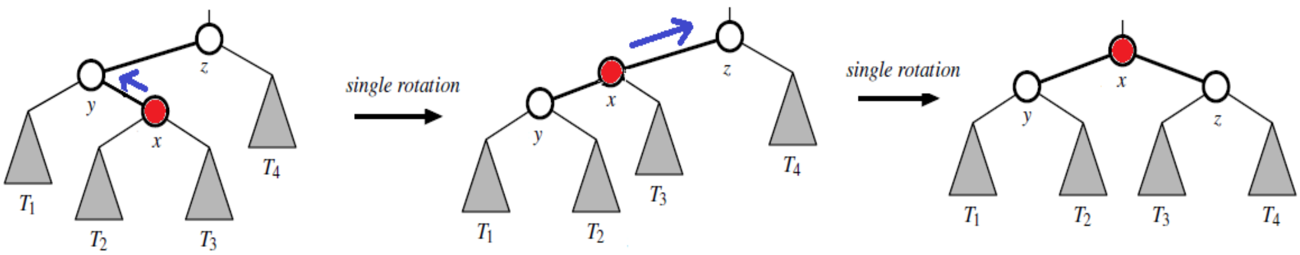
بازسازی سه‌گانه با یک دَوَران به شکلهای زیر توجه کنید. عمل بازسازی سه‌گانه روی راس x انجام می‌شود. راس x باید پدر (y) و پدربزرگ (z) داشته باشد. در هر دو حالت زیر یک عمل دوران صورت گرفته است. عمل دوران روی راس y انجام می‌شود.



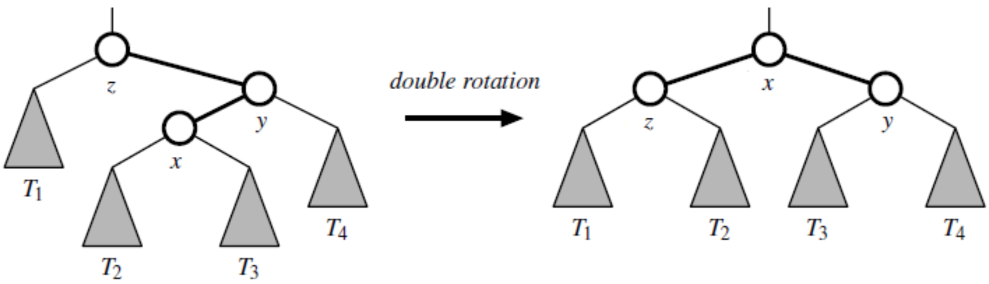


بازسازی سه‌گانه با دو دَوَران شکل زیر یک بازسازی سه‌گانه با دو دوران را نشان می‌دهد. دقت کنید در این حالت هر بار دوران روی راس x انجام شده است. جهت دوران با فلش مشخص شده است.





شکل زیر یک حالت دیگر از بازسازی سه‌گانه (با دو دوران) را نشان می‌دهد. مانند حالت بالا هر بار عمل دوران روی راس x انجام می‌شود.



نکته: در همه حالات بازسازی سه‌گانه، راسی که از لحاظ مقدار وسط قرار گرفته است در انتها در ریشه قرار می‌گیرد.

۳.۱ زمان اجرای دوران و بازسازی سه‌گانه

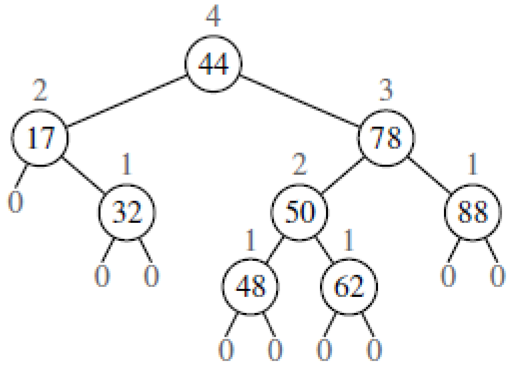
همانطور که در شکلهای بالا مشاهده می‌شود، دوران و بازسازی سه‌گانه با چند جابجایی زیردرختهای BST قابل اجراست. این نشان می‌دهد که زمان اجرای این اعمال $O(1)$ است.

۲ درخت AVL

برای تعریف درخت AVL از مفهوم ارتفاع یک راس استفاده می‌کنیم. در اینجا بر خلاف قاعده گذشته که ارتفاع یک برگ را صفر در نظر گرفتیم، در اینجا ارتفاع یک برگ را 1 در نظر می‌گیریم. فرزندان یک برگ که تهی هستند را به حساب می‌آوریم و ارتفاع آنها را صفر در نظر می‌گیریم. حال با این فرض، درخت AVL بصورت زیر تعریف می‌شود:

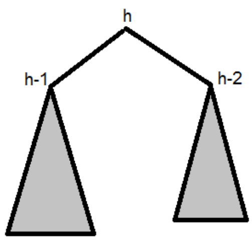
تعریف: درخت AVL یک درخت BST است با این ویژگی که هر راس آن فرزندانش حداکثر یک واحد با هم اختلاف ارتفاع داشته باشند.

شکل زیر یک نمونه از درخت AVL را نشان می‌دهد. ارتفاع هر راس را بالای آن نوشته‌ایم. دقت کنید فرزندان تهی را هم نوشته‌ایم و ارتفاع آنها را صفر در نظر گرفته‌ایم.



لم. یک درخت AVL با n راس ارتفاعش $O(\log n)$ است.

اثبات: فرض کنید $n(h)$ حداقل تعداد رئوس یک درخت AVL با ارتفاع h باشد. ابتدا یک کران پایین برای $n(h)$ بدست آوریم.
$n(h)$ را بصورت استقرائی بررسی می‌کنیم. روشن است که $n(1) = 1$. همچنین $n(2) = 2$. این یعنی اینکه یک درخت AVL با ارتفاع 2 به حداقل 2 راس نیاز دارد. حال یک درخت AVL با ارتفاع h (وقتی که $h \geq 3$) را در نظر بگیرید که کمترین تعداد رئوس را داشته باشد. این درخت دو زیردرخت دارد که هر دو کمترین تعداد رئوس را دارند. یکی ارتفاعش $h - 1$ و دیگری ارتفاعش $h - 2$ است.



(دقت کنید هر دو زیردرخت نمی‌توانند ارتفاع $h - 1$ داشته باشند. چون در این صورت می‌توان از یکی از زیردرختها تعدادی راس کم کرد و ارتفاع آن را به $h - 2$ تقلیل داد. بدین ترتیب یک درخت AVL با ارتفاع h بدست می‌آوریم که تعداد رئوس کمتری دارد و این با فرض ما متناقض است.)

از بحث بالا نتیجه می‌گیریم، برای $h \geq 3$

$$n(h) = 1 + n(h-1) + n(h-2)$$

چون $n(h-1) \geq n(h-2)$ پس می‌توانیم بنویسیم

$$\begin{aligned} n(h) &\geq 2.n(h-2) \\ &\geq 4.n(h-4) \\ &\geq 8.n(h-6) \\ &\vdots \end{aligned}$$

$$\geq 2^i.n(h-2i)$$

اگر قرار بدهیم $i = \lceil \frac{h}{2} \rceil - 1$ بدست می‌آید:

$$n(h) \geq 2^{\lceil \frac{h}{2} \rceil - 1}.n(h - 2\lceil \frac{h}{2} \rceil + 2)$$

نتیجه می‌دهد،

$$n(h) \geq 2^{\lceil \frac{h}{2} \rceil - 1}.n(1) \geq 2^{\lceil \frac{h}{2} \rceil - 1}$$

در نهایت می‌توانیم بگوییم

$$h < 2\log(n(h)) + 2$$

این نشان می‌دهد که ارتفاع درخت AVL با تعداد رئوس آن یک رابطه لگاریتمی دارد. لذا $h = O(\log n)$