پروژه ی پایانی درس ساختمان داده

ورودی اول

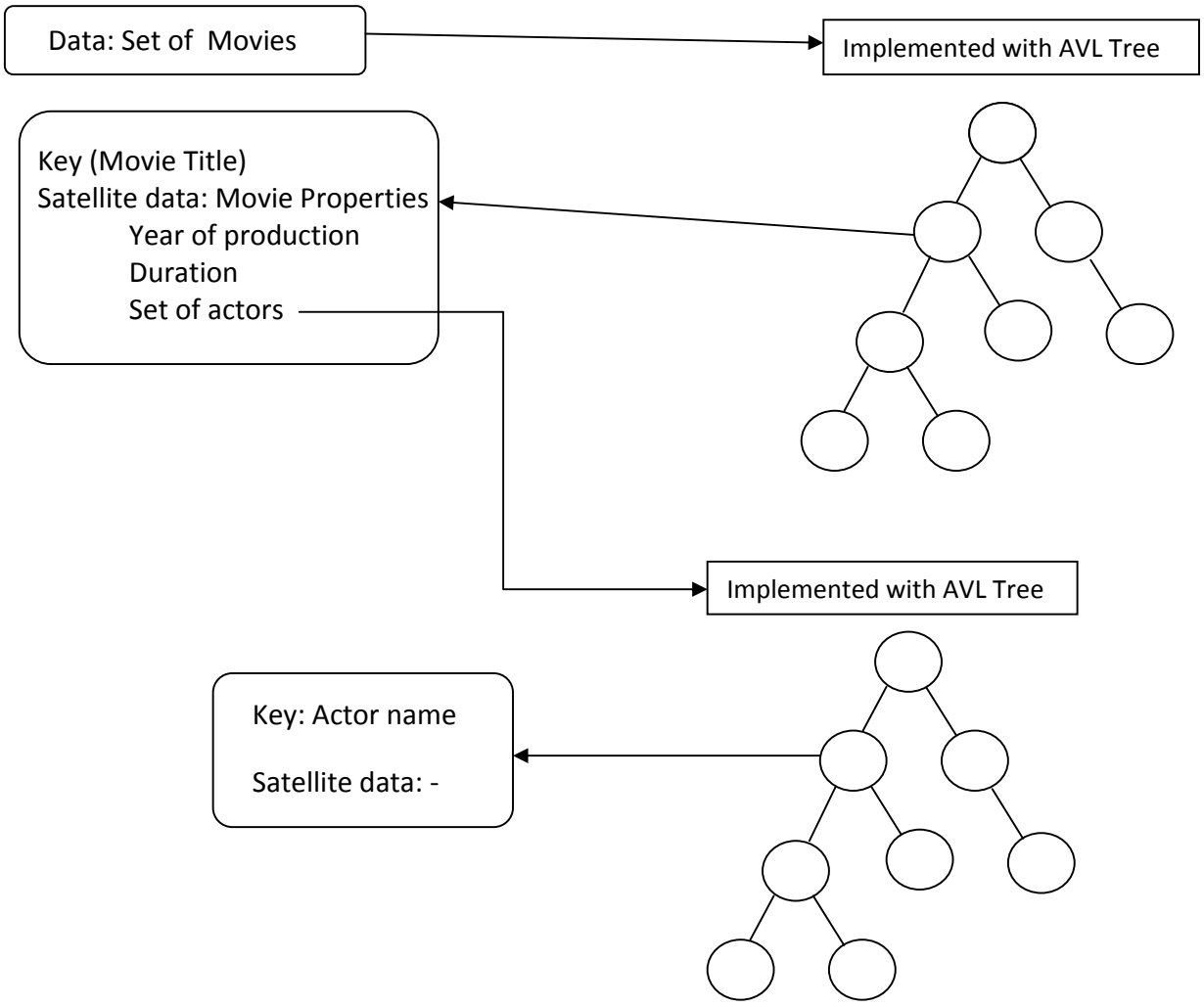
سینا قادرمرزی

86213153

طرح کلی ساختمان داده

بانک اطلاعاتی مورد نظر به صورت درخت درخت AVL از داده های فیلم طراحی شده است که کلید آن ها نام فیلم ها و داده های جانبی آن شامل مجموعه بازیگران ، سال تولید و مدت زمان فیلم است که که مجموعه ی بازیگران خود یک درخت AVL با کلید نام بازیگران است با توجه به ساختمان داده ی درخت AVL تنها فضای اضافی به کار رفته در ساختمان داده همان فضای مربوط به اشاره گر های موجود و نیز ضریب تعادل است که به ازای هر بازیگر مقدار ثابت کوچک و به ازای هر فیلم هم مقدار ثابت کوچکی فضا توسط ساختمان داده مصرف می شود

رشته های به کاررفته جهت ذخیره ی نام بازیگران و فیلم ها هم همانگونه که گفته شده بود به صورت لیست پیوندی از کاراکتر ها پیاده سازی شده است .



الگوریتم های جستجو و درج و حذف در درخت AVL همان گونه که انتظار می رود دارای زمان مصرفی $\Theta(\log(n))$ هستند.

مصرف حافظه در ساختمان داده

به این ترتیب مصرف حافظه به این صورت است که اگر m فیلم داشته باشیم که فیلم i ام n_i بازیگر داشته باشد . مصرف حافظه ی مربوط به ساختمان داده بدون احتساب فضای مربوط به نام های داده ها که از قبل ساختمان آن ها تعیین شده است. و دیگر داده ها ، به این صورت خواهد بود

$$\Theta(\sum_{1}^{m} n_i)$$

و در حالت ساده اگر m فیلم د9اشته باشیم که هر کدام از آن ها n بازیگر داشته باشند حاصل به صورت زیر در می آید

$$\Theta(m.n)$$

نحوه ی پیاده سازی عملیات موجود در ساختمان داده

- **اضافه کردن یک فیلم جدید به مجموعه**

این عمل به این صورت انجام می شود که 1- یک گره برای فیلم جدید ایجاد می شود و نام وسال وزمان فیلم از ورودی گرفته می شود و در گره جدید ذخیره می شود که این عمل Θ(1) است 2- یک AVL جدید برای مجموعه ی بازیگران ایجاد می شود و نام بازیگران با تعداد N که در ورودی نوشته شده است از ورودی گرفته شده و به درخت AVL بازیگران اضافه می شود که Θ(log(n!)) است 3- در نهایت گره ایجاد شده به درخت AVL فیلم ها اضافه می شود . که Θ(log(m)) است که m تعداد فیلم های مجموعه است .

- **حذف یک فیلم از مجموعه**

نام فیلم مورد نظر از ورودی گرفته می شود و کلید مورد نظر از درخت فیلم ها حذف می شود. اگر چنین کلیدی در درخت مورد نظر وجود نداشته باشد این عملیات تنها شامل جستجو برای کلید مورد نظر است که مرتبه ی زمانی آن برابر با Θ (log(m)) است که m تعداد فیلم های مجموعه است و اگر این کلید موجود باشد باید تمام منابع اختصاص یافته به داده های مربوط به این فیلم نیز آزاد شوندکه این عمل شامل آزاد کردن همه ی گره های بازیگران آن است که این عمل از مرتبه ی Θ(n) می باشد . که n تعداد بازیگران فیلم است .

- **اضافه کردن یک بازیگر جدید به یک فیلم**

نام فیلم و بازیگر مورد نظر از ورودی گرفته می شود و در مجموعه ی فیلم ها برای نام آن فیلم جستجو می شود $\Theta(\log(m))$ 2- درصورت وجود، نام جدید به مجموعه ی بازیگران آن اضافه می شود $\Theta(\log(n))$

- **حذف یک بازیگر از یک فیلم**

1- نام فیلم و بازیگر از ورودی گرفته می شود برای نام فیلم در مجموعه ی فیلم ها جستجو می شود $\Theta(\log(m))$

2- در صورت وجود ، کلید مربوط به نام بازیگر وارد شده از مجموعه ی بازیگران آن فیلم حذف می شود $\Theta(\log(n))$

- **جستجو برای یک فیلم**

1- نام فیلم از ورودی گرفته می شود و در درخت فیلم ها برای کلید مورد نظر جستجو می شود $\Theta(\log(m))$

2- در صورت پیدا شدن، نام فیلم سال تولید و مدت زمان فیلم درخروجی چاپ می شود و سپس درخت بازیگران پیمایش می شود و همه ی کلید ها در خروجی چاپ می شود .

- **یافتن تمام فیلم های یک بازیگر**

درخت فیلم ها پیمایش می شود و به از ای هر گره برای بازیگر وارد شده در درخت بازیگران جستجو می شود و در صورت یافتن کلید فیلم در خروجی چاپ می شود .$\Theta(m.\log n)$ در صورتی که m فیلم در مجموعه باشند که هر کدام n بازیگر دارند.

- **یافتن تمام فیلم های یک سال مشخص**

سال مورد نظر از ورودی گرفته می شود درخت فیلم ها پیمایش می شود و به ازای هر گره اگر سال تولید برابر سال مورد نظر باشد کلید آن فیلم در خروجی چاپ می شود $\Theta(m)$

- **یافتن تمام فیلم های ساخته شده در یک بازه ی زمانی معین**

سال ابتدایی و انتهایی از ورودی گرفته می شود . درخت فیلم ها پیمایش می شود و هر کدام از گره ها که دارای سال تولید بین دو سال داده شده بباشد کلید آن فیلم در خروجی چاپ می شود $\Theta(m)$

- **چاپ همه ی فیلم های موجود در مجموعه**

درخت فیلم ها پیمایش می شود و به ازای هر گره کلید آن در خروجی چاپ می شود $\Theta(m)$

الگوریتم های درخت AVL

- **الگوریتم درج**

اولین کار در الگوریتم درج در درخت AVL یافتن جای مناسب برای گره جدید به طوری که خاصیت BST بودن درخت حفظ شود پس از یافتن جای مناسب گره جدید به صورت یک برگ به درخت اضافه می شود. همان طور که که می دانیم در درخت AVL همه ی گره ها متعادل اند یعنی اختلاف ارتفاع زیر درخت راست و چپ آن ها حداکثر 1 است . درج یک گره در درخت AVL می تواند تعادل برخی گره ها را بر هم بزند و در نتیجه درخت از حالت AVL خارج شود پس بعد از درج کردن گره جدید یک سری جابه جایی ها–که به آن ها چرخش می گوییم– لازم است تا مجددا همه ی گره ها متعادل شوند و در نتیجه درخت AVL شود .

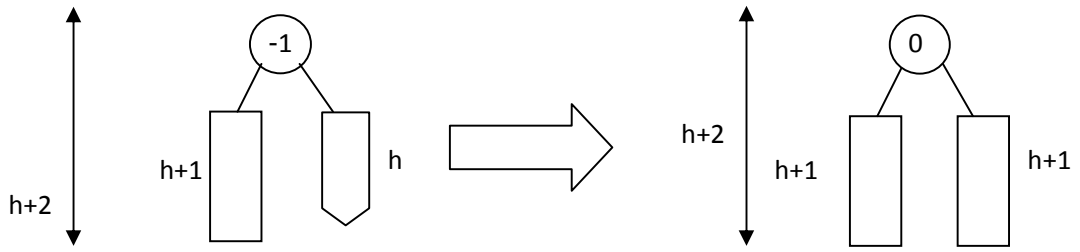
درتوضیحاتی که در ادامه می آید از اصطلاح ضریب تعادل یک گره به معنی ارتفاع زیر درخت راست منهای ارتفاع زیر درخت چپ آن گره استفاده می شود و آن را روی گره ها می نویسیم.

هنگامی که گرهی به درخت AVL اضافه می شود ارتفاع یکی از زیردرخت های پدر آن از 0 به 1 می رسد .

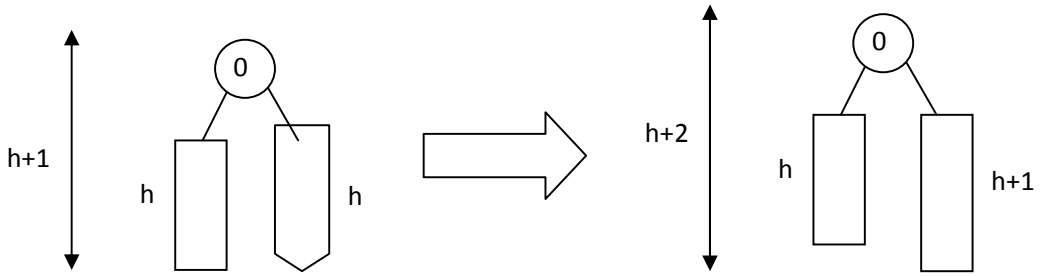
برای هرگرهی که که ارتفاع یکی از زیر درخت های آن یک واحد افزایش یابد با فرض این که زیر درخت های آن AVL باشند بسته به شرایط ممکن است یکی از حالات زیر پیش بیاید

فرض می کنیم ارتفاع زیر راست افزایش می یابد

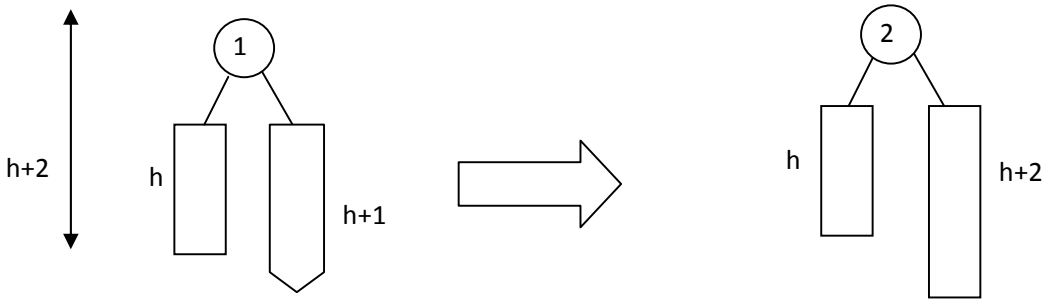
1– ضریب تعادل گره برابر 1– باشد، در این حالت راس متعادل باقی می ماند و ارتفاع آن نیز تغییری نمی کند .



2- ضریب تعادل گره برابر 0 باشد ، تعادل این گره به هم نمی خورد اما ارتفاع زیر درخت کلی یک واحد افزایش می یابد

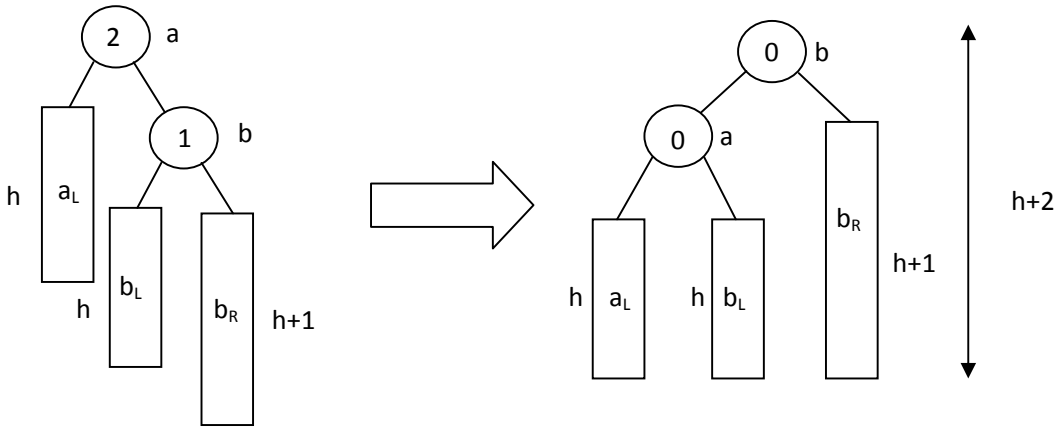


3- ضریب تعادل گره برابر 1 باشد. در این حالت تعادل گره به هم می خورد و نیاز به تغییراتی در ساختمان زیر درخت هست



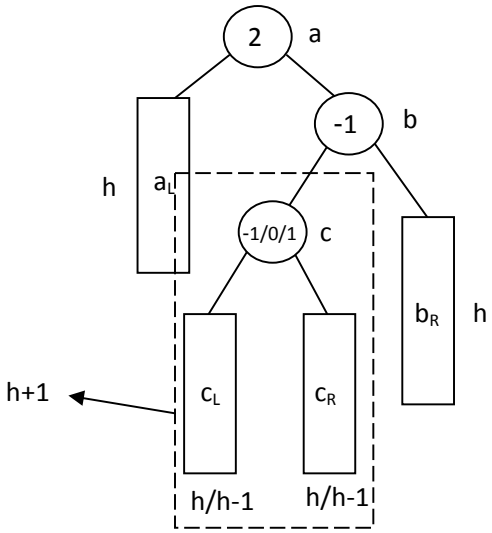
بسته به شرایط یکی از دوچرخش موجود روی گره انجام می شود که حاصل هر دو نوع چرخش این است که ارتفاع زیر درخت تحت تاثیر این تغییرات تغییری نمی کند و با ارتفاع آن قبل از افزایش ارتفاع زیر درخت یکی می ماند.

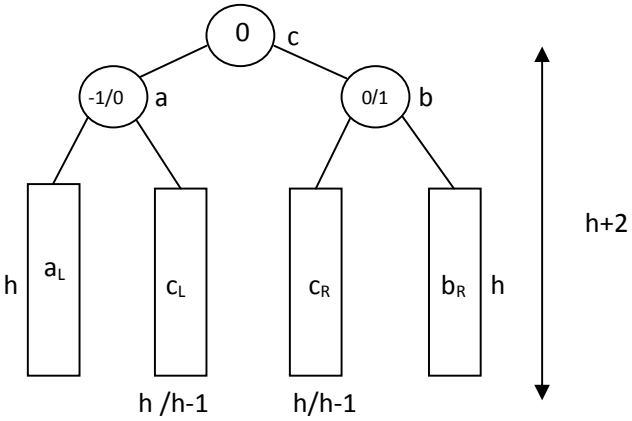
3-1- ضریب تعادل فرزند راست برابر با 1 باشد



که همانگونه که دیده می شود ارتفاع زیر درخت ثبت می ماند .

2-3- ضریب تعادل فرزند راست برابر با 1- باشد .





که در این حالت هم ارتفاع زیر درخت ثابت می ماند

3-3 ضریب تعادل فرزند راست برابر صفر باشد ؛ در این حالت افزایش ارتفاع زیر درخت نمی تواند به علت افزایش ارتفاع یکی از زیر درخت های خود آن باشد زیرا در این صورت ضریب تعادل آن صفر نخواهد بود ونمی تواندبه علت تبدیل این زیر درخت از تهی به یک تک گره باشد چون این با 1 بودن ضریب تعادل گره اصلی تناقض دارد پس چنین حالتی نمی تواند اتفاق بیفتد

با توجه به مطالب گفته شده هنگامی که یک گره به صورت برگ به درخت اضافه می شود ارتفاع زیر درخت پدر آن افزایش می یابد . اگر گره پدر در شرط 2 صدق کند ارتفاع زیر درخت کل نیز افزایش می یابد و عیناً این اتفاق برای پدر پدر نیز می افتد و این سلسله ادامه پیدا می کند تازمانی که یابه ریشه ی درخت برسیم یا به یکی از اجداد برسیم که در شرایط 1 یا 3 صدق کند که در صورت لزوم چرخش را می دهیم و چون دیگر ارتفاع زیر درخت تغییری نکرده است در حالت اجداد بالاتر تغییری داده نمی شود و بنابراین متعادل باقی می مانند و ضرایب تعادل آن ها هم تغییری نمی کند .

بنابراین الگوریتم درج را به این صورت می توان بیان کرد.

- راس جدید را درجای مناسب خود قرار می دهیم اگر این مکان ریشه ی درخت است الگوریتم به پایان می رسد (زیرا یک تک گره ، یک درخت AVL است)و در غیر این صورت پدرگره را نگه می داریم
- گره فعلی رابرابر پدر قرار می دهیم .

- اگر گره فعلی در حالت 1 یا 3 قرار دارد اعمال تغییرات به صورت گفته شده و پایان الگوریتم
- اگر گره فعلی در حالت 2 قرار دارد تغییرات را در ضریب تعادل گره فعلی ایجاد می کنیم و
- گره فعلی را برابر پدر گره فعلی قرار می دهیم و اگرپدر نداشت پایان الگوریتم

- **الگوریتم حذف**

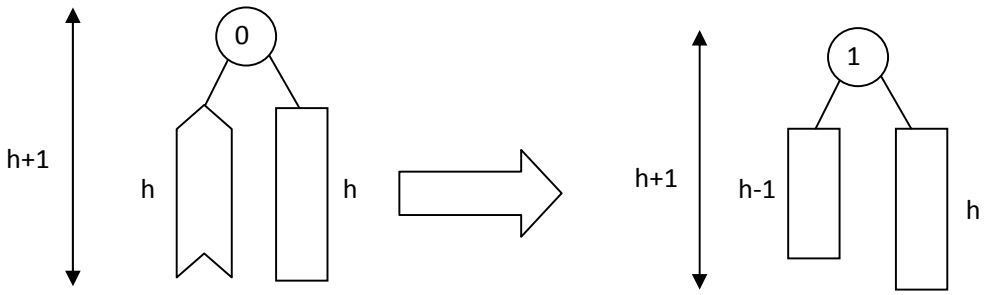
درالگوریتم حذف گره از درخت AVL که ورودی آن یک کلید است اولین کاری که انجام می شود این است که برای کلید مورد نظر در درخت جستجو می شود و در صور ت یافت نشدن الگوریتم بااعلام عدم یافت پایان می یابد.اگر در درخت چنین کلیدی وجود داشته باشد مانند درخت BST این کلید درسه حالت از درخت حذف می شود اما این حذف در نهایت به حذف یک گره برگ یا تک فرزندی از درخت می انجامد . که این می تواند ارتفاع زیر درخت ها و ضرایب تعادل برخی از اجداد آن گره را تحت تاثیر قرار دهد . و بنابراین ممکن است به بعضی تغییرات در ساختمان درخت نیاز باشد .

وقتی یک گره از درخت حذف می شود ارتفاع یکی از زیر درخت های پدر آن کم می شود .

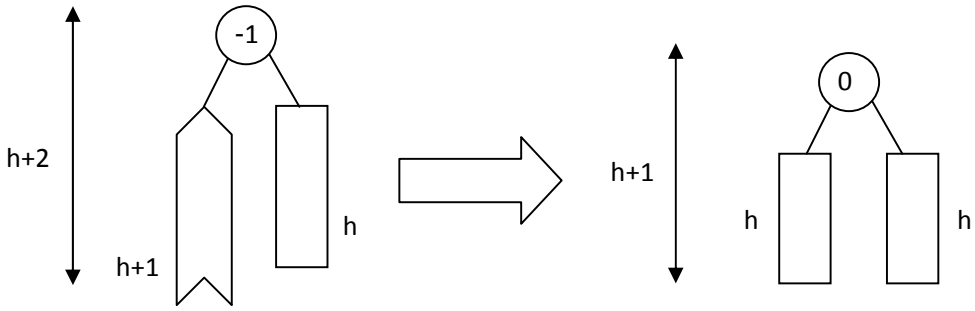
برای یک گره که ارتفاع یکی از زیر درخت های آن یک واحد کم می شود برحسب شرایط می تواند سه حالت روی دهد

فرض می کنیم که ارتفاع زیر درخت چپ یک واحد کم می شود

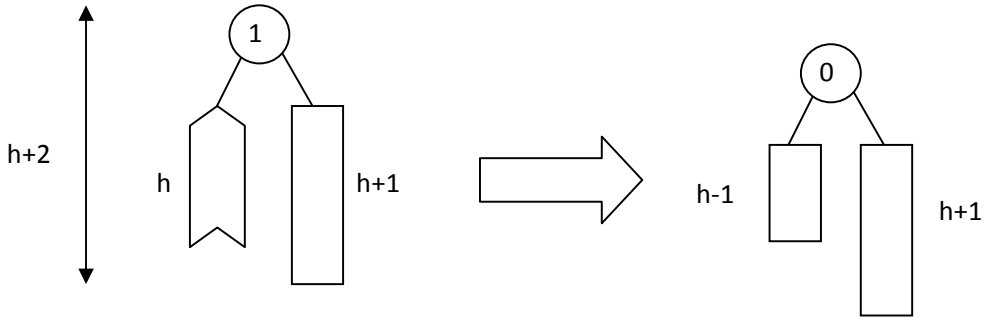
1- اگر ضریب تعادل گره برابر 0 باشد ، تعادل گره باقی مانده و ارتفاع زیر درخت کلی نیز تغییری نمی کند .



2- اگر ضریب تعادل گره برابر 1- باشد ، گره متعادل باقی می ماند ولی ارتفاع آن 1 و احد کم می شود.

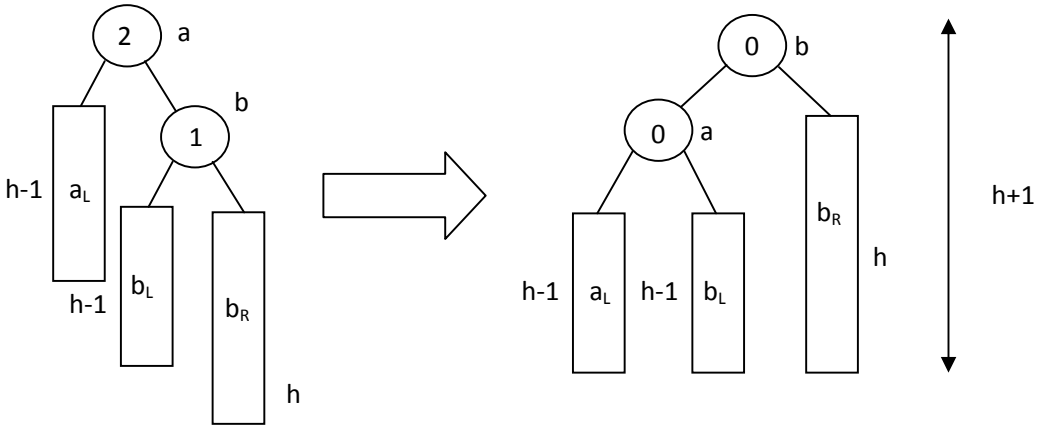


3- اگر ضریب تعادل آن برابر1 باشد ، تعادل گره به هم می خورد و نیاز به تغییراتی در ساختمان زیر درخت هست.

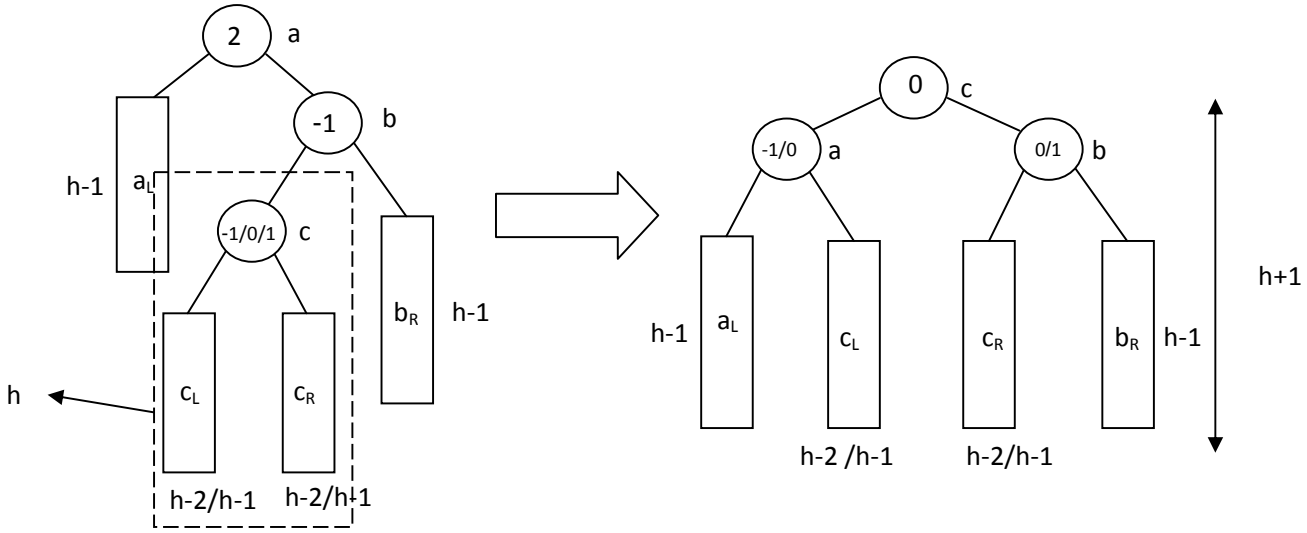


سه حالت ممکن است پیش بیاید

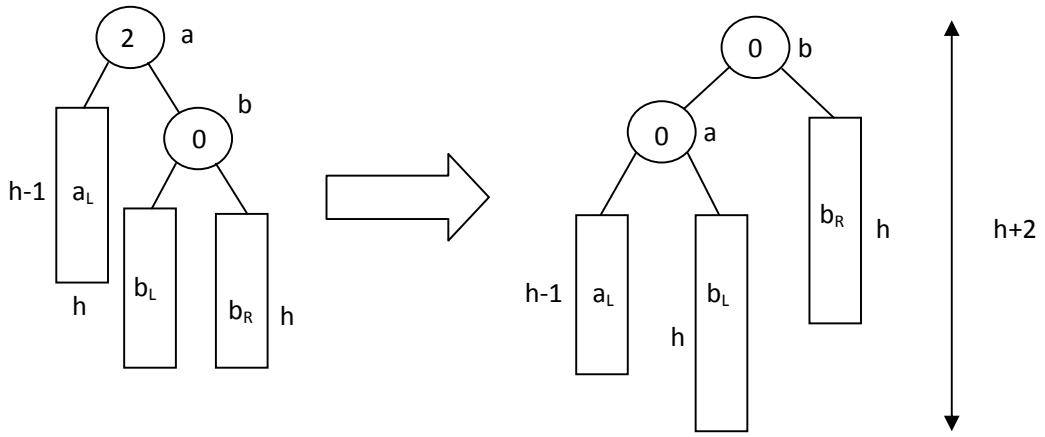
3-1- ضریب تعادل فرزند راست برابر 1 باشد با چرخش زیر، زیردرخت دوباره متعادل می شود و ارتفاع آن نسبت به حالت اولیه یک واحد کمتر می شود



3-2- ضریب تعادل فرزند راست برابر 1- باشد ، در این صورت با چرخش زیر درخت دوباره AVL می شود اما ارتفاع آن یک واحد کم می شد



3-3-اگر ضریب تعادل فرزند راست برابر صفر باشد ، در این صورت با چرخش زیر درخت AVL می گردد . در این حالت ارتفاع زیر درخت تغییری نمی کند



پس در نهایت در دوحالت ارتفاع زیر درخت تغییر می کند که آن ها را حالات A می نامیم ودر دو حالت ارتفاع آن تغییری نمی کند که آن هارا حالات B می نامیم

با توجه به مطالب گفته شده هنگامی که یک گره ازه درخت حذف می شود ارتفاع زیر درخت پدر آن کاهش می یابد . باتوجه به این که پدر درکدام یک از خالات صدق کند یا فقط ضریب تعادل آن تغییر می کند یا عملیات چرخشی روی آن انجام می شود اگر گره پدر در حالت A باشد ارتفاع زیر درخت کل نیز افزایش می یابد و عیناً این اتفاق برای پدر پدر نیز می افتد و این سلسله ادامه پیدا می کند تازمانی که یابه ریشه ی درخت برسیم یا به یکی از اجداد برسیم که درحالت B باشد و چون در این حالت دیگر ارتفاع زیر درخت تغییری نکرده است در حالت اجداد بالاتر تغییری داده نمی شود و بنابراین متعادل باقی می مانند و ضرایب تعادل آن ها هم تغییری نمی کند .

بنابراین الگوریتم حذف را به این صورت می توان بیان کرد.

1- ابتدا عملیات حذف همانند درخت BST انجام می شود اگر گرهی که به طور فیزیکی حذف شده ریشه ی درخت باشد الگوریتم خاتمه می یابد چون درخت شامل دوگره یا یک گره بوده که باحذف ریشه تبدیل به یک درخت تهی یا یک درخت تک گره می شود که AVL است . در غیر این صورت یک گره به عنوان پدر گره حذف شده در اختیار ما قرار دارد

//اکنون باید عملیات AVL سازی مجدد را روی درخت انجام دهیم به این منظور از گره فعلی شروع کرده وبه سمت بالا می رویم و عملیات را روی گره فعلی تا زمانی که به انتها برسیم یا به گرهی از نوع B انجام می دهیم

2- گره فعلی را برابر پدر گره حذف شده قرار می دهیم ،

3- باتوجه به این که گره فعلی در کدام یک از حالات قرار دارد ،عملیات را بر روی آن انجام می دهیم ،اگر گره فعلی از نوع B یا ریشه ی درخت بود الگوریتم خاتمه می یابد و در غیر این صورت گره فعلی را برابر پدر گره فعلی قرار می دهیم و مرحله ی 3 تکرار می شود