دانشکده مهندسی کامپیوتر

جزوه درس

ساختمان‌های داده

استاد درس: سید صالح اعتمادی

پاییز ۱۳۹۸

جلسه ۲۴

درخت AVL

هژار آزیز - ۱۳۹۸/۹/۱۸

جزوه جلسه ۲۴ام مورخ ۱۳۹۸/۹/۱۸درس ساختمان‌های داده تهیه شده توسط هژار آزیز.

۱.۲۴ مروری بر مباحث جلسه گذشته

در جلسه گذشته با مفهوم درخت‌های دودویی و چگونگی کارکرد آن‌ها آشنا شدیم. همچنین یاد گرفتیم که چگونه تابع‌های Insert Search، Next، Find و Delete را پیاده‌سازی کنیم. با وجود اینکه درخت‌های دودویی یکی از بهینه‌ترین ساختمان‌های داده هستند، اگر به همان شیوه‌ی ساده و ابتدایی خود پیاده‌سازی شوند، با مشکلاتی روبه‌رو خواهند شد. یکی از این مشکلات، به‌هم‌خوردن تعادل درخت است که در این جلسه به بحث درمورد و راه‌حل‌های آن می‌پردازیم.

۲.۲۴ مقدمه‌ای بر درخت AVL

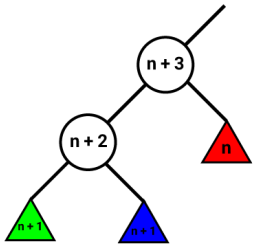
درخت AVL نام خود را از اول نام دو مخترع خود به نام‌های Adelson-Velsky و Landis گرفته است. این درخت در علوم کامپیوتر یک درخت خودمتوازن‌کننده است که اولین نوع از این ساختمان داده است.

۳.۲۴ درخت AVL و پیاده‌سازی آن

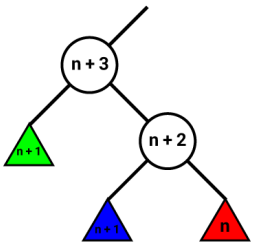
درخت AVL مزیت‌های زیادی دارد که یکی از آن‌ها کاهش ارتفاع درخت برای کاهش پیچیدگی محاسباتی عملیات‌های انجام شده بر روی درخت دودویی است. شرط اصلی برقراری خاصیت درخت AVL این است که تفاوت ارتفاع زیردرخت‌های چپ و راست هر گره در درخت حداکثر برابر یک باشد. اگر این شرط برقرار نباشد، درخت دارای خاصیت AVL نخواهد بود. این خاصیت در نهایت باعث خواهد شد که پیچیدگی محاسباتی همه‌ی عملیات‌های که در درخت دودویی AVL انجام می‌شود از $O(Log(n))$ خواهد بود.

۱.۳.۲۴ درخت AVL با مثال

به درخت‌های دودویی دو شکل زیر نگاه کنید. درخت شکل ۱.۲۴ از قاعده‌ی AVL پیروی نمی‌کند. درخت شکل ۲.۲۴ صحیح شده‌ی درخت اولی است و از چرخش درخت شکل ۱.۲۴ به سمت راست به‌دست آمده‌است.



شکل ۱.۲۴: یک نمونه از درخت دودویی که قاعده‌ی AVL در آن رعایت نشده است.



شکل ۲.۲۴: درخت فوق از چرخش درخت شکل ۱.۲۴ به سمت راست بدست آمده است.

۲.۳.۲۴ پیاده‌سازی و شبه‌کد درخت AVL

همان‌طور که می‌دانید در هربار اضافه‌کردن یک گره جدید به درخت دودویی، درخت تغییر می‌کند. بنابراین هربار با اضافه‌کردن یک گره جدید باید درخت را دوباره متعادل(Rebalance) کنیم.

```
initialization;
AVLInsert(k, R)
 Insert (k, R);
 N = Find (k, R);
 Rebalance (N);
 return;
```

Algorithm 1: AVL Tree Insertion

حالا که مرحله اضافه کردن گره را پیاده‌سازی کردیم، به سراغ حذف یک گره از درخت می‌رویم. در این مرحله نیز درخت تغییر می‌کند و باید دوباره درخت را متعادل کنیم تا خاصیت AVL برقرار باشد.

```
initialization;
AVLDelete(N)
 Delete (N);
 M = Parent of node replacing N;
 Rebalance (M);
 return;
```

Algorithm 2: AVL Tree Deletion

برای پیاده‌سازی دقیق‌تر شبه‌کد بالا، به اسلایدهای اصلی درس مراجعه کنید.

Bibliography