جزوه جلسه دوازدهم داده ساختارها و الگوریتم

۱۱ آبان ۱۴۰۰

فهرست مطالب

۱ مرتب سازی سریع تصادفی

در این نسخه، عنصر لولا در هر مرحله، به صورت تصادفی و با احتمال مساوی بین عناصر آرایه(A) انتخاب میشود.
به دیگر بیان، عدد r بین ۰ و $\mathrm{len(A)}$ - 1 انتخاب شده و $\mathrm{A[r]}$ به عنوان عنصر لولا انتخاب میگردد. این کار در عین سادگی بسیار کارامد نیز هست. همچنین فرض میشود انتخاب عدد رندوم در $O(1)$ انجام میگیرد.

۱.۱ مرتبه زمانی مرتب سازی سریع تصادفی

ادعا میکنیم امید ریاضی (میانگین) زمان اجرا برای ورودی های مختلف برابر $O(nlogn)$ است.
حالت خاص این الگوریتم یعنی مرتب سازی سریع تصادفی وسواسی را در نظر گرفته و ادعای خود را برای آن اثبات میکنیم. در این الگوریتم، بعد از انتخاب عنصر لولا و تشکلی L و G، اگر اندازه هرکدام از آنها حداقل $\mathrm{len(A)}/4$ نبود، لولای دیگری را انتخاب میکنیم.
پس اگر لولا در آرایه مرتب شده A در ۲ قسمت میانی باشد (بعد از تقسیم آرایه به ۴ قسمت) الگوریتم ادامه پیدا میکند و درغیر اینصورت لولای دیگری انتخاب میشود. با این تفاسیر، احتمال انخاب لولای مناسب برابر $1/2$ است و در نتیجه، امید ریاضی تعداد تکرار انتخاب لولا با احتمال $1/2$ درستی لولا، طبق توزیع هندسی برابر ۲ است.
طبق توضیحات فوق داریم:

$$\mathrm{T(n)} = \#\mathrm{iterations}.\Theta(n) + \mathrm{T(n/4)} + \mathrm{T(3n/4)}$$

با اعمال امید ریاضی نیز داریم:

$$\mathrm{E(T(n))} = \mathrm{E}(\#\mathrm{iterations}).\Theta(n) + \mathrm{E(T(n/4))} + \mathrm{E(T(3n/4))}$$

میدانیم:

$$\mathrm{E}(\#\mathrm{iteration}) = 2$$

پس در نهایت مرتبه زمانی در بدترین حالت برابر است با:

$$\mathrm{E(T(n))} = \Theta(nlogn)$$

۲ کران پایین الگوریتم های مختلف مرتب سازی

برای پیدا کردن کران پایین مرتب سازی، نیاز به یک مدل محاسباتی داریم؛ زیرا از جلسات اول میدانیم هر الگوریتم در یک مدل محاسبات اجرا میشود.
مدل محاسبه خود را، مدل مقایسه انتخاب میکنیم. در این مدل، عناصر واسط هایی

هستند که تنها میتوانیم آنها را باهم مقایسه کنیم و دسترسی مستقیم به مقادیر آنها نداریم.
زمان اجرای الگوریتم در این مدل، تعداد کل مقایسه هاست. همچنین از عملیات دیگر به جز مقایسه برای محاسبه کران پایین صرف نظر میکنیم؛ هرچند این صرف نظر برای اثبات ما مشکلی ایجاد نمیکند.
تمام مرتب سازی هایی که تا به حال دیدیم به جز الگوریتم Binary Insertion Sort با این مدل محاسباتی سازگار هستند (تنها از مقایسه عناصر برای مرتب سازی استفاده میشود)

۱.۲ درخت تصمیم

این درخت، با درخت هایی قبلی که خواندیم متفاوت است و یک مفهوم انتزاعی است و برای اثبات های ریاضیاتی کاربرد دارد. در عمل، درختی رسم نمیکنیم. همچنین درخت تصمیم منحصر به مدل مقایسه نیست و در دیگر مدل های محاسباتی نیز وجود دارد.
برای هر الگوریتم مبتی بر مقایسه با اندازه ورودی فیکس n، یک درخت تصمیم معادل آن نسبت داده میشود. در این درخت تمام مقایسه های ممکن برای عناصر مختلف و نتایج ممکن برای هر مقایسه بصورت یکجا به تصویر کشیده میشود. رئوس داخلی مقایسه ها و برگ ها نتایج ممکن هستند.

۱.۱.۲ مقایسه درخت تصمیم و الگوریتم در مدل مقایسه

۱. هر راس داخلی در درخت ≡ یک تصمیم دودویی در الگوریتم
۲. برگ های درخت ≡ یک جواب پیدا شده برای الگوریتم
۳. هر مسیر از ریشه به برگ در درخت ≡ یک بار اجرای الگوریتم
۴. طول مسیر از ریشه به برگ در درخت ≡ زمان یک بار اجرای الگوریتم (به ازای یک ورودی خاص)
۵. ارتفاع درخت ≡ طول بزرگترین مسیر از ریشه به برگ ≡ زمان اجرای الگوریتم در بدترین حالت

۲.۱.۲ مسئله کران پایین جست و جو

با درخت تصمیم، میخواهیم اثبات کنیم برای n عنصر پیش پردازش شده، پیدا کردن یک عنصر در مدل مقایسه، حداکثر به زمان $O(nlogn)$ نیاز دارد. دو مورد را در نظر داریم:
۱. درخت تصمیم، یک درخت باینری است
۲. درخت تصمیم متناظر با مسئله حداقل n برگ دارد. این تعداد ممکن است بیشتر هم بشود زیرا ممکن است عنصری که دنبال آن میگردیم در آرایه تکرار شده باشد.
با توجه به دو بند بالا، ارتفاع درخت برابر $\Omega(logn)$ است که معادل زمان اجرای الگوریتم

در بدترین حالت است.
در نتیجه، جست و جوی دودویی روی آرایه و درخت های دودویی جست و جوی متوازن (AVL و R-B) بهینه هستند.

۲.۲ کران پایین مرتب سازی

به مانند مسئله قبل، دو بند زیر را داریم:
۱. درخت تصمیم تشکیل شده، دودویی است.
۲. حداقل تعداد برگ ها برابر $n!$ است. زیرا برای هر آرایه ورودی، باید تمام حالت های ممکن را درنظر بگیریم.
پس ارتفاع درخت حداکثر $logn!$ ($\Omega(logn!)$) میباشد.
در ادامه ثابت میکنیم:

$$\Theta(logn!) = \Theta(nlogn)$$

این کار را به دو روش انجام میدهیم:

۱. تبدیل ضرب به جمع در لگاریتم

$$log(n!) = \sum_{i=1}^{n} logi > (n/2)log(n/2) = \Omega(nlogn)$$

۲. تقریب استرلینگ

$$log(n!) = log(\sqrt{2\pi n}(n/e)^n) = nlogn - nlog_e n + O(logn) = \Theta(nlogn)$$

در نهایت میتوان ادعا کرد مقادیر $nlogn$ و $log(n!)$ بصورت اردری باهم برابرند و اختلاف اندکی دارند.
از تقریب استرلینگ نیز میتوان نتیجه گرفت: $O(n!) = O(n^2)$
همچنین طبق درخت تصمیم میتوان ادعا کرد با تعداد تقریبی $log(n!)$ مقایسه میتوان آرایه را مرتب کرد.
درنهایت با توجه به نتیجه ای که از کران پایین مرتب سازی گرفتیم، میتوانیم ادعا کنیم الگوریتم های مرتب سازی ادغامی، هرمی، AVL، R-B و سریع با انتخاب هوشمندانه لولا (به صورت مجانبی و اردری در مدل مقایسه) بهینه هستند.