به نام خدا

ساختمان داده‌ها و الگوریتم‌ها (۴۰۲۵۴)

دانشگاه صنعتی شریف — تمرین چهارم

مدرس: دکتر مهدی صفرنژاد — انتشار: ۱۸ اردیبهشت ۱۴۰۱

درهم‌سازی

سؤالات را با دقت بخوانید و روی همه آن‌ها وقت بگذارید. تمرین‌های تئوری تحویل گرفته نمی‌شوند اما از آن‌ها سؤالات کوییز مشخص می‌شود. بنابراین روی سؤالات به خوبی فکر کنید و در کلاس‌های حل تمرین مربوطه شرکت کنید. در سوالات ارائه الگوریتم از روش درهم‌سازی استفاده کنید.

سؤال ۱. اگر در یک جدول هش که از روش زنجیره برای رفع برخورد استفاده می‌کند به جای زنجیره مرتب نشده از لیست‌های مرتب شده استفاده کنیم، پیچیدگی زمانی انجام هر یک از عملیات زیر در بهترین و بدترین حالت چگونه تغییر می‌کند؟

۱. درج یک عنصر جدید در آرایه

۲. پیدا کردن یک عنصر در آرایه

پاسخ:

۱. در بهترین حالت برای درج هیچ برخوردی رخ نمی‌دهد و پیچیدگی زمانی درج در هر دو روش $\theta(1)$ خواهد بود.
در بدترین حالت برای درج، تمام عناصر درج شده تا کنون در یک خانه قرار گرفته‌اند. در این حالت درج در لیست مرتب نشده پیچیدگی زمانی $\theta(1)$ و در لیست مرتب شده پیچیدگی زمانی $\theta(n)$ خواهد داشت.

۲. در بهترین حالت برای پیدا کردن یک عنصر، خانه حاوی آن عنصر فقط یک عضو دارد و در هر دو روش پیچیدگی زمانی پیدا کردن آن $\theta(1)$ خواهد بود.
در بدترین حالت برای پیدا کردن یک عنصر، تمامی عناصر در یک خانه قرار گرفته‌اند و پیدا کردن آن، در لیست مرتب نشده پیچیدگی زمانی $\theta(n)$ و در لیست مرتب شده (به روش جستجوی دودویی) پیچیدگی زمانی $\theta(log(n))$ خواهد داشت.

سؤال ۲. الگوریتمی با پیچیدگی زمانی $O(n)$ ارائه کنید که با گرفتن آرایه‌ای به طول n، طول بلندترین زیر مجموعه این آرایه که متشکل از اعداد متوالی است را محاسبه کند. (ترتیب قرارگیری این اعداد متوالی در آرایه اهمیتی ندارد.)

پاسخ: ابتدا تمامی عناصر آرایه را وارد یک جدول درهم‌سازی می‌کنیم. سپس برای هر عنصر آرایه ($arr[i]$) اعمال زیر را اجرا می‌کنیم:

۱. چک می‌کنیم که ایا این عنصر می‌تواند نقطه شروعی یک دنباله متوالی باشد یا نه. برای این کار کافی است که وجود یا عدد وجود $arr[i] - 1$ در جدول در هم سازی را چک کنیم.

۲. اگر جواب مرحله قبل مثبت بود، طول دنباله اعداد متوالی شروع شده از $arr[i]$ را محاسبه می‌کنیم.

در پایان هر مرحله طول دنباله محاسبه شده را با max_length مقایسه می‌کنیم و اگر از آن بزرگ‌تر بود، max_length را برابر با این طول جدید قرار می‌دهیم. (مقدار اولیه max_length را 0 در نظر می‌گیریم)

سؤال ۳. تعدادی شیرینی از انواع مختلف داریم و می‌خواهیم آن‌ها را بین تعدادی مشتری تقسیم کنیم، اما هر مشتری بیش از سه شیرینی از یک نوع را نمی‌گیرد. تعداد شیرینی (n) و تعداد مشتری (k) و آرایه n تایی نشان دهنده نوع هر شیرینی را داریم و می‌خواهیم تشخیص دهیم که می‌توان تمام شیرینی‌ها را بین مشتری‌ها تقسیم کرد یا نه. حداقل پیچیدگی زمانی انجام این کار را محاسبه کنید.

پاسخ: یک جدول درهم‌سازی از تعداد هرنوع شیرینی ایجاد می‌کنیم. این کار با یک بار پیمایش آرایه انجام می‌شود و پیچیدگی زمانی آن $O(n)$ می‌باشد. سپس یک دور روی عناصر جدول درهم‌سازی پیمایش می‌کنیم. اگر هیچ عضوی از $3k$ بزرگ‌تر نبود، تقسیم شیرینی‌ها ممکن و در غیر این صورت غیرممکن است.
پیچیدگی زمانی این الگوریتم $O(n)$ می‌باشد.

سؤال ۴. آرایه‌ای از اعداد داریم که عنصر ماکسیمم آن را با A و عنصر مینیمم آن را با B نشان می‌دهیم. می‌خواهیم تعداد اعدادی که باید به آرایه اضافه شوند تا تمام اعداد B تا A در آن موجود باشند را محاسبه کنیم. الگوریتمی با پیچیدگی زمانی $O(A-B)$ ارائه کنید که این کار را انجام دهد.

پاسخ: تمام عناصر آرایه را وارد یک جدول درهم‌سازی می‌کنیم. روی اعداد B تا A پیمایش می‌کنیم و وجود یا عدم وجود آن‌ها در جدول را چک می‌کنیم. تعداد اعدادی که در جدول موجود نبودند، برابر با پاسخ مسئله است.

سؤال ۵. آرایه‌ای n عضوی و عدد k را داریم. می‌خواهیم آرایه را به گروه‌های دوتایی تقسیم کنیم، به طوری که جمع اعضا هر گروه بر k بخش پذیر باشد. حداقل پیچیدگی زمانی برای تشخیص امکان انجام این کار را محاسبه کنید.

پاسخ: روی عناصر آرایه پیمایش می‌کنیم و تعداد اعداد با هر باقی‌مانده به k را در یک جدول درهم‌سازی ذخیره می‌کنیم. به این صورت که با فرمول $r = ((arr[i]\%k) + k)\%k$ باقی‌مانده عنصر مورد نظر به k را محاسبه کرده و سپس مقدار عنصر ایندکس r در جدول را یکی اضافه می‌کنیم. در نهایت دوباره آرایه را پیمایش می‌کنیم و شروط زیر را برای هر باقی‌مانده چک می‌کنیم:

۱. اگر باقی‌مانده برابر با $\frac{k}{2}$ یا 0 باشد، باید تعداد آن در جدول درهم‌سازی زوج باشد.

۲. اگر باقی‌مانده هر عدد دیگری باشد، باید تعداد آن با تعداد باقی‌مانده $k-r$ برابر باشد.

اگر شروط بالا برای تمام باقی‌مانده‌ها برقرار باشند، امکان تقسیم آرایه ممکن و در غیر این صورت غیر ممکن است.
پیچیدگی زمانی این الگوریتم $O(n)$ می‌باشد.

سؤال ۶. یک چهارتایی فیثاغورثی شامل ۴ عدد صحیح (a, b, c, d) است به صورتی که $a^2 + b^2 + c^2 = d^2$. آرایه‌ای شامل n عضو داریم، الگوریتمی با پیچیدگی زمانی $O(n^2)$ ارائه دهید که وجود یا عدم وجود چهارتایی فیثاغورثی در آرایه را تشخیص دهد.

پاسخ:

کافی است که چهارتایی (a, b, c, d) را پیدا کنیم به صورتی که رابطه $a^2 + b^2 = d^2 - c^2$ برقرار باشد. برای این کار ابتدا تمام n^2 جفت مرتب از اعضای آرایه را در نظر گرفته و برای هر جفت (a, b) مقدار $a^2 + b^2$ را محاسبه کرده و در یک جدول درهم‌سازی ذخیره می‌کنیم. سپس بار دیگر روی تمام جفت‌ها پیمایش می‌کنیم و برای هر جفت (c, d) مقدار $d^2 - c^2$ را محاسبه کرده و وجود یا عدم وجود آن در جدول درهم‌سازی را چک می‌کنیم. اگر این مقدار برای حداقل یکی از این جفت‌ها در جدول وجود داشته باشد، این یعنی جفت (a, b) وجود داشته به صورتی که $a^2 + b^2 = d^2 - c^2$ و چهارتایی فیثاغورثی در آرایه موجود می‌باشد.

سؤال ۷. دو لیست از اعداد صحیح داریم. اولی به طول n و دیگری به طول m. الگوریتمی از $O(n + m)$ ارائه دهید که بررسی کند آیا دو عدد یافت می‌شود که یکی در لیست اول و دیگری در لیست دوم باشد و حاصل جمعشان مقدار ذکر شده باشد.

پاسخ:

اگر فرض کنیم جمع مورد نظر، s است، اعضای مجموعه اول را پیمایش می‌کنیم و به ازای هر عضو این مجموعه مانند a کلید $s - a$ را وارد جدول درهم‌سازی کرده سپس روی اعضای مجموعه دیگر پیمایش می‌کنیم و به ازای هر عضو بررسی می‌کنیم که ایا در جدول درهم‌سازی وجود دارد که اگر برای حداقل یک عضو این اتفاق بیوفتند جواب مسئله مثبت و در غیر این صورت منفی می‌باشد.

سؤال ۸. الگوریتمی از $O(n)$ ارائه دهید که در یک رشته به طول n اولین کاراکتر تکراری را پیدا کند.

پاسخ:

ابتدا یک جدول در هم سازی تشکیل می‌هیم. روی رشته پیمایش انجام می‌دهیم به این صورت که برای هر کاراکتر ابتدا بررسی می‌کنیم که ایا در جدول درهم‌سازی وجود دارد اگر وجود داشته باشد جواب همان کاراکتر خواهد بود و در غیر این صورت آن کاراکتر را به جدول درهم‌سازی اضافه می‌کنیم.

سؤال ۹. در جدول درهم سازی با استفاده از روش وارسی خطی، تابع درهم سازی برای جدولی با اندازه ۹ به صورت زیر است:

I	H	G	F	E	D	C	B	A	key
۷	۱	۱	۵	۴	۰	۴	۱	۱	hash

اگر جدول درهم سازی در ابتدا تهی باشد، به چند حالت می‌توان این عناصر را در جدول درج کرد تا در نهایت جدول زیر تولید شود؟

۸	۷	۶	۵	۴	۳	۲	۱	۰	i
F	I	G	E	C	H	B	A	D	T(i)

پاسخ:

$$hash(A) = 1, hash(B) = 1, hash(H) = 1$$
$$T(1) = A, T(2) = B, T(3) = H$$

با توجه به روابط بالا ترتیب این سه حرف مشخص می‌شود.

$$hash(C) = 4, hash(E) = 4, hash(G) = 1$$
$$T(4) = C, T(5) = E, T(6) = G$$

با توجه به روابط بالا، G بعد از ۵ حرف H, E, C, B, A باید باشد و C قبل از E قرار می‌گیرد. در نتیجه جایگشت های متفاوت ۶ حرف ذکر شده، به صورت $C(2, 5)$ زیرا ۲ مکان از ۵ مکان ممکن انتخاب شده و با توجه به ترتیب E, C قرار می‌گیرند. با توجه به روابط ذکر شده، H, B, A در ۳ مکان باقیمانده جایشان مشخص خواهد شد. در نتیجه ۱۰ حالت متفاوت برای قرار گیری این ۶ حرف وجود دارد. حرف I در بین هر یک از ۶ حرف قبل می تواند قرار بگیرد. در نتیجه، ۷ حالت متفاوت برای این حرف داریم. جایگاه حرف F در بعد از ۷ حرف قبلی مشخص است. در نهایت حرف D در بین هر یک از ۸ حرف قبلی می تواند قرار بگیرد و ۹ حالت متفاوت شکل خواهد گرفت. در نتیجه به $9 \times 7 \times 10 = 630$ حالت ممکن عناصر را می توان در جدول درج کرد.

سؤال ۱۰. الگوریتمی از $O(n \log n)$ ارائه دهید که در رشته‌ای به طول n تعداد زیررشته‌های آینه‌ای را پیدا کند.

پاسخ:

اول از همه بین همه ی کاراکترها و همین طور ابتدا و انتهای رشته یک کاراکتر دلخواه درج می‌کنیم تا طول رشته حتما فرد شود. علت این کار ساده شدن فرایند جستجوی دودویی در ادامه مسئله است. حالا به هر کاراکتر یک عدد یکتا نسبت می‌دهیم و تابع هش را برای رشته ی $s = s_0 s_1 ... s_{n-1} s_n$ به شکل زیر تعریف می‌کنیم.

$$h(s) = p^{n-1} s_0 + p^{n-2} s_1 + ... + p s_{n-1} + s_n$$

آرایه T را طبق رابطه بازگشتی $T[0] = s_0, T[i] = s_i + pT[i-1]$ پر می‌کنیم با این کار برای مقدار تابع هش هر زیرآرایه $s_{i,j}$ خواهیم داشت: $h(s_{i,j}) = T[j]T[i-1]p^{(j-i)+1}$ همین کار را برای معکوس رشته s و آرایه Q انجام می‌دهیم. سپس سراغ گام نهایی می‌رویم. می‌دانیم هر زیررشته آینه‌ای یک کاراکتر میانی خواهد داشت. بنابراین روی تمام رشته لوپ می‌زنیم. و هر بار یک کاراکتر را کاراکتر میانی در نظر می‌گیریم. و تمام زیررشته های آینه ای که کاراکتر انتخاب شده کاراکتر میانی آن است را پیدا می‌کنیم. برای این کار از آرایه هایی که قبلا حساب کردیم و همچنین جستجوی باینری استفاده می‌کنیم. روش کلی به این صورت است که که اگر s و t دو زیرشته با کاراکتر میانی مشترک باشند و همچنین t زیررشته s باشد، تنها در صورتی s آینه‌ای است که t هم آینه‌ای باشد. همچنین تمام زیر رشته های t نیز آینه‌ای هستند بنابراین با در نظر گرفتن یک کاراکتر میانی و استفاده از جست‌وجوی دودویی می توان ابتدا زیررشته ای را بررسی کرد که ابتدای آن، وسط بازه ی ابتدای رشته تا کاراکتر میانی در نظر گرفته شده است و مکان انتهای آن نیز متناظر با ابتدای آن در سمت دیگر کاراکتر میانی است. حالا اگر هش این زیررشته با هش وارون آن برابر باشد متقارن است و می‌توان سراغ رشته های بزرگ‌تر که این رشته زیر رشته آنها است برویم و در غیر این صورت جستجوی دودویی را به زیررشته های کوچک‌تر محدود کنیم. همچنین دیدیم که با استفاده از آرایه هایی که قبلا حساب کردیم، می‌توانیم هش یک زیررشته یا وارون آن را در $O(1)$ حساب کنیم. برای پیچیدگی زمانی نیز می‌دانیم جستجوی دودویی در زمان $O(\log n)$ چون n بار از جستجوی دودویی استفاده می‌کنیم پس پیچیدگی زمانی برابر با $O(n \log n)$ خواهد بود.

مثلا رشته $\#a\#f\#c\#a\#c\#f\#b\#$ و دومین کاراکتر a از سمت چپ را بعنوان کاراکتر میانی در نظر بگیرید. برای جستجوی

دودویی به این صورت عمل می‌کنیم که در ابتدا وسط زیررشته $\#a\#f\#c\#$ یعنی f را بعنوان مبدا باینری سرچ قرار می‌دهیم و آینه ای بودن زیر رشته $f\#c\#a\#c\#f$ را بررسی می‌کنیم. چون این زیررشته آینه ای است، پس تمام زیررشته های آن نیز آینه‌ای هستند. و جستجو را برای زیردنباله های بلندتر که کاراکتر میانی آن‌ها a مذکور است انجام می‌دهیم. که در اینجا زیرآرایه ی $a\#f\#c\#a\#c\#f\#b$ خواهد بود. طبق چیزی که گفته شد، هر بار یکی از کاراکترهای آرایه به عنوان کاراکتر میانی در نظر گرفته خواهد شد و جستجوی دودویی با روشی که گفتیم انجام می‌شود.

سؤال ۱۱. الگوریتمی از $O(n)$ ارائه دهید که همه‌ی زیرآرایه‌ها با مجموع عناصر ۰ از آرایه ای با طول n را پیدا کند.

پاسخ:

یک جدول درهم سازی تشکیل می‌دهیم(مانند هش مپ در زبان های برنامه نویسی) به طوری که key آن یک عدد و $value$ آن ارایه از اندیس ها باشد که مجموع عناصر آن آرایه تا آن اندیس همان key بوده است. یک متغیر به اسم sum تعریف میکنیم و ایتدا مقدار آن را برابر ۰ در نظر میگیریم. روی آرایه پیمایش میکنیم و به ازای هر عنصر مقدار آن را با sum جمع می‌کنیم تا مجموع عناصر تا آن عنصر از آرایه به دست بیاید. اگر sum به دست آمده برابر با ۰ بود اندیس ۰ تا آن عنصر را به عنوان یک خروجی در نظر می‌گیریم. سپس بررسی می‌کنیم که sum به دست آمده در جدول درهم‌سازی وجود دارد یا نه اگر وجود داشت $value$ متناظر با آن را که آرایه ای از اندیس های قبلی است به ازای هر عضوش با مقدار فعلی اندیس به عنوان خروجی در نظر می‌گیریم زیرا اگر این نشان می‌دهد که مجموع عناصر از عضو اول تا آن عضو برابر مجموع عناصر از عضو اول تا اندیس فعلی است پس مجموع عناصر بین این ۰ می‌باشد. در آخر در صورت وجود نداشتن آن sum در جدول آن را به همراه اندیس عنصر به جدول اضافه می‌کنیم و در غیر اینصورت تنها آن اندیس را به $value$ متناظر با sum اضافه می‌کنیم. سراغ عنصر بعدی در آرایه اصلی می‌رویم و همین روند را ادامه می‌دهیم.

سؤال ۱۲. الگوریتمی ارائه دهید که اشتراک و اجتماع دو لیست به طول‌های m و n را در $O(m+n)$ پیدا کند.

پاسخ:

اجتماع:

یک آرایه خالی بعنوان اجتماع دو محموعه ایجاد کنید. یک جدول هش خالی ایجاد کنید. هر دو لیست را یکی یکی طی کنید، برای هر عنصری که بازدید می شود، اگر عنصر در جدول هش موجود نیست، عنصر را در لیست اجتماع و جدول هش وارد کنید. اگر عنصر موجود است، آن را نادیده بگیرید.

اشتراک:

یک آرایه خالی بعنوان اشتراک دو مجموعه ایجاد کنید. یک جدول هش خالی ایجاد کنید. آرایه اول را پیمایش کنید. برای هر عنصری که در این آرایه بازدید می شود، عنصر را در جدول هش وارد کنید. اکنون آرایه دوم را پیمایش کنید. برای هر عنصری که در این آرایه بازدید می شود،اگر عنصر در جدول هش موجود است، عنصر را در لیست نتایج قرار دهید. اگر عنصر موجود نیست، آن را نادیده بگیرید.

موفق باشید