مریم علیپور | 96920371 — تمرین سری اول

سوال 1)

a) $T(n) = 2T(n/2) + \frac{n}{\log n}$

با روش master theorem قابل حل نیست چون $n\log n$ به طور چند جمله‌ای از n بزرگتر نیست.

$f(n) = \frac{n}{\log n}$ $\quad n^{\log_b^a} = n^{\log_2^2} = n$

فرض کنیم $f(n) = \Theta(n^{\log_b^a}) = \Theta(n) \Rightarrow \frac{n}{\log n} \leqslant cn \Rightarrow n \leqslant c\, n\log n$

که نعضم باید به صورت Polynomial کوچکتر باشد که نیست.

b) $T(n) = \frac{1}{2}T(\frac{n}{2}) + n$

با master theorem حل نمی‌شود، چون از شرایط این روش $a \geqslant 1$ بود که در اینجا $a = \frac{1}{2}$ است.

c) $T(n) = T(\frac{3n}{4}) + 1$

$a = 1,\ b = 4/3,\ f(n) = 1,\ n^{\log_b^a} = n^{\log_{4/3}^1} = n^0 = 1$

$\xrightarrow{\text{شرط 2}} f(n) = \Theta(n^{\log_b^a}) \Rightarrow 1 = \Theta(1) \Rightarrow 1 \leqslant c(1)$

$\Rightarrow T(n) = \Theta(\cancel{n^{\log_b^a}}\, \lg n) = \Theta(\lg n)$

d) $T(n) = 7T(\frac{n}{3}) + n^2$

$a = 7,\ b = 3,\ f(n) = n^2$

$\xrightarrow{\text{شرط 3}} f(n) = \Omega(n^{\log_b^{a+\varepsilon}}) \Rightarrow T(n) = \Theta(f(n))$

$n^2 \geqslant c\,n^{\log_3^{7+\varepsilon}}$ and $7(\frac{n}{3})^2 \leqslant cn^2$ که هر دو شرط برقرار هستند.

$\Rightarrow T(n) = \Theta(n^2)$

e) $T(n) = 2T(\frac{n}{2}) + n\log(n) \longrightarrow$ با master theorem حل نمی‌شود.

در ابتدا ممکن است به نظر برسد که با حالت 3 حل می‌شود چون $f(n)$ از $n^{\log_b^a}$ بزرگتر است

اما مشکل این است که به صورت polynomial بزرگتر نیست.

پس $T(n)$ در حالتی بین حالت 2 و حالت 3 است.

f) $T(n) = T(\frac{n}{2}) + n(2 - \cos(n))$

$a = 1 ,\ b = 2 ,\ f(n) = 2n\cos(n) ,\ n^{\log_b^a} = n^{\log_2^1} = n^0 = 1$

شرط 3 $\longrightarrow 2n\cos(n) \geq c\, n^{\log_2^{1+\varepsilon}} \xrightarrow[n=2\pi]{\varepsilon=1} 0 \geq c\, 2\pi^{\log_2^{1+\varepsilon}}$ ✗

پس با روش master theorem حل نمی‌شود.

(سوال 2

a) 4 مسئله دارد :

هر 4 مسئله در 1 روز	$\to$	1 1 1 1
3 تا در 1 روز و یکی در 2 روز		1 1 1 2
2 تا در 2 روز و 2 تا در 1 روز		1 1 2 2
3 تا در 2 روز و یکی در 1 روز		1 2 2 2
هر 4 مسئله در 2 روز		2 2 2 2

$$P(A=4) = \frac{3}{4} \times \frac{3}{4} \times \frac{3}{4} \times \frac{3}{4} = \frac{81}{256}$$

$$P(A=5) = \binom{4}{1} \times \frac{3}{4} \times \frac{3}{4} \times \frac{3}{4} \times \frac{1}{4} = \frac{108}{256}$$

$$P(A=6) = \binom{4}{2} \times \frac{3}{4} \times \frac{3}{4} \times \frac{1}{4} \times \frac{1}{4} = \frac{54}{256}$$

$$P(A=7) = \binom{4}{3} \times \frac{3}{4} \times \frac{1}{4} \times \frac{1}{4} \times \frac{1}{4} = \frac{12}{256}$$

$$P(A=8) = \frac{1}{4} \times \frac{1}{4} \times \frac{1}{4} \times \frac{1}{4} = \frac{1}{256}$$

$$\Rightarrow E(A) = 4 \times P(A=4) + 5 \times P(A=5) + 6 \times P(A=6) + 7 \times P(A=7) + 8 \times P(A=8) = \boxed{5}$$

b) $P(B=0) = 1/6$

$P(B=1) = 5/6 \times 1/6$

$P(B=2) = 5/6 \times 5/6 \times 1/6$

$\vdots$

$P(B=n) = (\frac{5}{6})^n \times 1/6$

$$\Rightarrow E(B) = \sum_{n=0}^{\infty} n \times (5/6)^n \times 1/6 = \boxed{5}$$

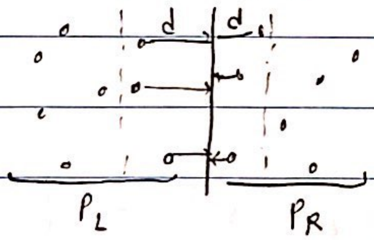
سوال 3)

$$n \cdot 2^n > n^3 \geq 8^{\log n} > n^{1/\log n} > (\log n)! > n^{\log(\log(n))} > \log(\log(n))$$

سوال 4) فاصله بین دو نقطه:

$$\|pq\| = \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2}$$

می‌توانیم با استفاده از D&C زمان اجرا را به $O(n\log n)$ برسانیم. ورودی: یک آرایه با n نقطه P[]
خروجی: کوچکترین فاصله بین نقاط. در ابتدا آرایه را براساس نقطه‌ی x با mergesort مرتب می‌کنیم $\Theta(n\log n)$
حال وسط آرایه‌ی مرتب شده را پیدا می‌کنیم مثلاً نقطه‌ی $P[n/2]$. آرایه را به دو قسمت تقسیم می‌کنیم.
سمت چپ از $P[0]$ تا $P[n/2]$ و آرایه سمت راست از $P[n/2+1]$ تا $P[n-1]$. به صورت بازگشتی کوچکترین فاصله
در این دو زیر آرایه را پیدا می‌کنیم. dl کوچکترین سمت چپ و dr کوچکترین سمت راست و d کوچکترین
بین این دو است. با این کار یک کران بالا برای کمترین فاصله داریم. حال باید کمترین فاصله بین نقاط سمت
چپ و سمت راست را پیدا کنیم. خط عمودی گذرنده از $P[n/2]$ را در نظر بگیرید و همه‌ی نقاطی را که فاصله‌شان نسبت
به x به خط عمودی نزدیکتر است را پیدا کنید و یک آرایه به اسم strip[] شامل این نقاط بسازید.



آرایه strip[] را نسبت به y مرتب کنید، که $O(n\log n)$ زمان می‌برد.
این مرحله می‌تواند به $O(n)$ تبدیل شود اگر به صورت بازگشتی و مرج کردن سورت شود.
کوچکترین فاصله در strip[] را پیدا کنید، که $O(n)$ زمان می‌برد و به صورت هندسی اثبات می‌شود که برای
هر نقطه در strip نیاز است که حداکثر 7 نقطه بعد از آن را چک کنیم. و در نهایت کمترین فاصله پیدا شده
در مرحله‌ی بالا را برمی‌گردانیم.

سوال 5) (غلط است) a) $f(n) = O(g(n))$ implies $2^{f(n)} = O(2^{g(n)})$

غلط است. مثال نقض:

$2n = O(n)$; $2n \leq cn \quad \forall c \geq 2$

$2^{2n} = O(2^n)$; $2^{2n} \leq c2^n$ ✗

b) $f(n) + o(f(n)) = \theta(f(n))$ (درست است)

یک عضو از مجموعه ی $o(f(n))$ در نظر بگیریم مثلا $g(n)$، پس خواهیم داشت:

$f(n) + g(n) = \theta(f(n)) \Rightarrow c_1 f(n) \leq f(n) + g(n) \leq c_2 f(n)$

از طرفی می دانیم:

$\forall c \quad \exists n_0 \text{ s.t } \forall n \geq n_0 \Rightarrow g(n) < cf(n)$

حال اگر $c_1 = 1$ و $c_2 = 2$ انتخاب کنیم قضیه برقرار خواهد شد.

c) $f(n) = \theta(f(\frac{n}{2}))$

درست نمی باشد. اگر $f(n) = 2^n$ قرار دهیم داریم:

$$c_1 2^{n/2} \leq 2^n \leq c_2 2^{n/2}$$

که این نامساوی غلط است پس تساوی بالا نادرست می باشد.

سوال ۶) ابتدا S را مرتب می کنیم. سپس با حلقه‌ی for یک iteration ایجاد می کنیم و برای هر عنصر i یک binary-search انجام می دهیم ببینیم آیا عنصری برابر با x-i وجود دارد یا خیر. اگر پیدا شد الگوریتم true را return می کند در غیر اینصورت False.

الگوریتم n بار تکرار می شود که هر بار یک binary-search روی یک آرایه با سایز n انجام می شود که اوردر زمانی $\Theta(n \lg n)$ را به ما می دهد. اگر ابتدا با merge-sort مرتب سازی را انجام دهیم این کار نیز $\Theta(n \lg n)$ دیگری را حاصل می کند که یک مقدار ثابت در انتهای الگوریتم به اوردر زمانی اضافه می شود اما نه به صورت asymptotic time.

شبه کد:

```
pair-exists(S, X):
    A = Merge-Sort(S)
    for i = 1 to S.length
        if binary-search(A, X - S[i]) != NIL
            return true
    return false
```

سوال 7) اگر به حلقه‌ای که در insertion_sort است نگاه کنیم:

while i > 0 and A[i] > key
    A[i+1] = A[i]
    i = i-1

این حلقه دو هدف دارد: ① یک جستجوی خطی تا زیرآرایه‌ی سورت شده را برای پیدا کردن جای مناسب key اسکن کند.

② عناصری که از key بزرگتر هستند را به انتهای آرایه‌ی شیفت دهد تا key را در جایگاه مناسب بگذارد.

اگرچه می‌توانیم تعداد مقایسه‌ها را با استفاده از binary_search کاهش دهیم اما همچنان نیاز داریم همه‌ی عناصر بزرگتر از key را شیفت دهیم تا برای key جا باز کنیم که این شیفت کردن عناصر از ما زمان می‌برد حتی در average case که باید نصف عناصر را شیفت دهیم. $\Theta(n)$ پس در نهایت worst-case زمان اجرای insertion-sort همچنان $\Theta(n^2)$ خواهد ماند.

سوال 8)

$$T(n) = T(n-1)+3 + T(n-2) + 4 + T(n-1)$$
$$= 2T(n-1) + T(n-2) + 7$$

small condition: $T(1) = 0$, $T(0) = 0$

سوال 9) فرض کنیم تمام با تجربه ها مرده اند و فقط دو نفر باقی مانده اند. a_i سهم اختصاص یافته به نفر i ام در هر مرحله به طوری که a_1 تقسیم بندی کننده)

$(a_5, a_6) = (300, 0)$ → 1 رأی دارد ✓

$(a_4, a_5, a_6) = (299, 0, 1)$ → چون نفر آخر می داند در مرحله ی بعد همین 1 دلار را از دست می دهد پس رأی می دهد.

نفر پنجم می داند که در سری بعد نفر آخر 1 دلار می گیرد پس موافقت می کند. → $(a_3, a_4, a_5, a_6) = (299, 0, 1, 0)$

نفر آخر و چهارم نمی خواهند همان 1 دلار را از دست بدهند → $(a_2, a_3, a_4, a_5, a_6) = (298, 0, 1, 0, 1)$ قبول می کنند.

$(a_1, a_2, a_3, a_4, a_5, a_6) = (298, 0, 1, 0, 1, 0)$ → این بار نفرات پنجم و سوم نمی خواهند 1 دلار را از دست بدهند پس قبول می کنند.

در نتیجه دزد ارشد به صورت آخر تقسیم بندی می کند.

سوال 10)

$(a_5, a_6) = (1, 0)$, $(a_4, a_5, a_6) = (0, 0, 1)$

$(a_3, a_4, a_5, a_6) = (0, 0, 1, 0)$ → می تواند به نفر 4 یا 5 بدهد فرقی ندارد و خودش هم به خودش رأی می دهد.

$(a_2, a_3, a_4, a_5, a_6) = (-, -, -, -, -)$ → به هر ترتیبی تقسیم کند رأی نمی آورد چون به 2 رأی نیاز دارد اما 1 دلار دارد.

$(a_1, a_2, a_3, a_4, a_5, a_6) = (-, —)$ ، //

سوال ۱۱) یک روش برای Merge کردن n عبارتی یک دنباله ایجاد می کند:

$\langle P_k : k=0,\dots,m\rangle$ به طوری که: (1) $P_0=\{1,\dots,m\}$ (2) P_{k+1} یک Partition از P_k است، $k=0,\dots,m-1$

(3) $|P_{k+1}|<|P_k|$ برای $k=0,\dots,m-1$ (4) $|P_m|=1$

این دنباله به وضوح یک درخت ریشه دار با n برگ برچسب دار تولید می کند.

مطابقت خوبی بین دنباله های تقسیم و درختان به یکی اجازه می دهد تا فقط یک فرزند راس داخلی داشته باشد به طوری که هر P_k مربوط به یک سطح از درخت باشد. و نیاز دارد که برگ ها هم در یک سطح باشند. اما ما به وضوح می توانیم شاخه های غیر شاخه ای را فرو بشکنیم و در عوض درختان ریشه دار با برگ های دارای برچسب را بشماریم که در آن ها هر راس داخلی حداقل دو فرزند دارد.

$M(n)$ را تعداد چنین دنباله های پارتیشن یا درختان در نظر بگیرید. واضح است $M(1)=M(2)=1$ و

$M(3)=\binom{3}{2}+\binom{3}{3}=4$.

در واقع اعداد درست ۱، ۱، 4، 26، 236، 2752 هستند. اعداد درخت Polygenetic با n راس. $M(n)$

جمع تمام پارتیشن ها است که $\sum_{i=1}^{m} n_i p_i = n$ که p_i ها اعداد صحیح متمایزی هستند.

$$M(n+1)=(n+2)M(n)+2\sum_{k=2}^{n-1}\binom{n}{k}M(k)M(n-k+1)$$

سوال 12)

$$2101 * 1130 = (21 \times 10^2 + 1) \times (11 \times 10^2 + 30)$$

$$= (21 \times 11) \times 10^4 + C_1 + 10^2 + 1 \times 30$$

$$\Rightarrow C_1 = (21+1) \times (11+30) - (21 \times 11) - (1 \times 30)$$

$$21 \times 11 = (2 \times 10 + 1) \times (1 \times 10 + 1) = (2 \times 1) \times 10^2 + C_2 \times 10 + 1 \times 1$$

$$\Rightarrow C_2 = (2+1) \times (1+1) - 2 \times 1 - 1 \times 1$$

$$22 \times 31 = (2 \times 10 + 2) \times (3 \times 10 + 1) = (2 \times 3) \times 10^2 + C_3 + 10 + 2 \times 1$$

$$\Rightarrow C_3 = (2+2) \times (3+1) - 2 \times 3 - 2 \times 1$$

سوال 13) در این روش هر بار مسئله را به 7 زیر مسئله به اندازه $\frac{1}{2}$ تقسیم می کنیم.

برای تشکیل این ماتریس 10 بار عمل جمع ماتریس ها رخ می دهد، و هر بار بازگشت شامل 8 بار عمل جمع ماتریس ها است:

$$T(n) = 7T(n/2) + 18(n/2)^2 \quad , \quad T(1) = 0$$

$$f(n) = 18\frac{n^2}{4} \qquad n^{\log_b^a} = n^{\log_2^7}$$

$$f(n) = \frac{9}{2}n^2 = \frac{9}{2}n^{\log_2^4} = O(n^{\log_2^{7-\varepsilon}}) \rightarrow \varepsilon = 3 > 0 \checkmark$$

$$\rightarrow T(n) = (n^{\log_2^7})$$

RAZZAGHI - رزاقی