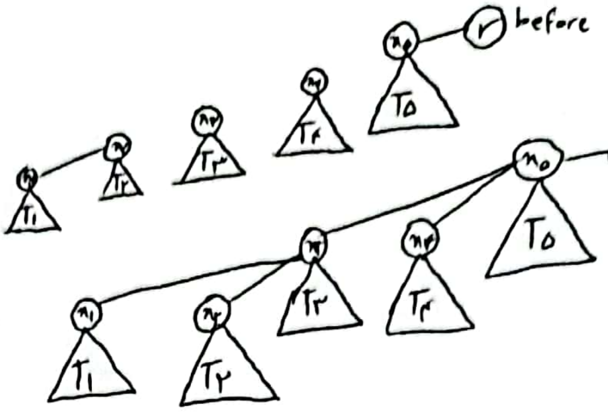
99102207 الهام محمدی

سؤال ۶ پایان ترم DS استاد صدیقین

پدر هر رأس در مسیر find را به پدربزرگش تغییر می‌دهیم و هزینه سرشکن را محاسبه می‌کنیم.
لم های مورد نیاز را هم در حل سؤال اثبات می‌کنیم. جواب سؤال بله است، هزینه سرشکن هر عملیات $O(\log^* n)$ است.
در ساختار داده DSU یا همون Disjoint-Set Union داخل تابع find، ۳ نوع بهبود داریم که همه را توضیح می‌دهم

۱) Path-halving

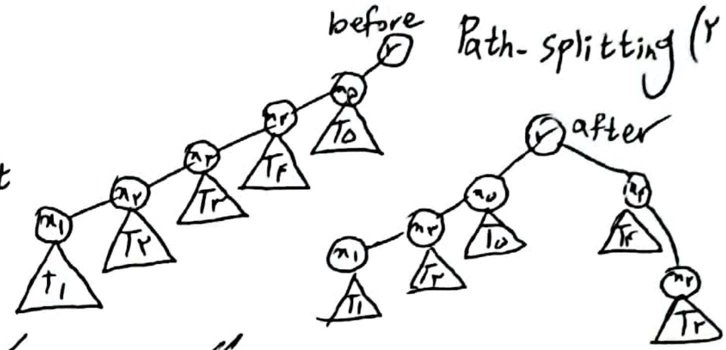
```
function Find
while x.parent ≠ x
    x.parent := x.parent.parent
    x := x.parent
    return x
```



در هنگام find برای هر رأس، پدرش را به پدربزرگش تغییر می‌دهیم

۲) Path-splitting

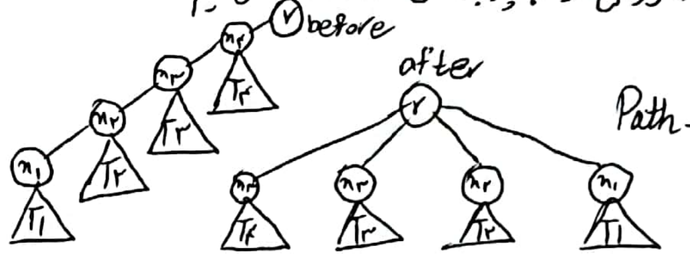
```
function Fid
while x.parent ≠ x
    x, x.parent := x.parent, x.parent.parent
    return x
```



در مسیر find، پدر هر رأس را به پدربزرگش switch می‌کنیم.

۳) Path-compression

```
function Find
if x.parent ≠ x
  x.parent := Find
  return x.parent
```



در هنگام find، تمام رأس های مسیر به ریشه وصل می‌شوند مستقیماً

روش مدنظر سؤال، روش دوم است و اُردر زمانی هر ۳ حالت بالا در worst case، $O(m \log^* n)$ می‌باشد.
با توجه به اینکه اگر رنگ یک رأس k باشد، زیر درخت آن شامل حداقل 2^k رأس است و نیز این موضوع که تعداد رئوس با رنگ k حداکثر $\frac{n}{2^k}$ است، برای محاسبه هزینه سرشکن، رئوس را تقسیم بندی به گروه‌هایی می‌کنیم.
گروه بندی اینگونه است که رئوس با رنگ صفر، در گروه صفرم، رئوس با رنگ یک در گروه یکم و به همین ترتیب ...
دسته ها به این شکل می‌شوند:
$\{0\}$ و $\{1\}$ و $\{2, 3\}$ و $\{4, 5, \dots, 15\}$ و ... و $\{r, \dots, 2^r - 1\}$

n: تعداد رئوس

پس ماکزیمم تعداد رئوس هر $\underbrace{\text{دسته بندی}}_{\text{هر گروه } \{r, \dots, 2^r-1\}}$، $\frac{2n}{2^r}$ است و در بدترین حالت به $\log^* n$ (برای n رأس) گروه نیاز می‌باشد.

اگر برای یک رأس رندوم مانند n هم، k تا اجداد هم گروه داشته باشیم، با یک بار اجرای تابع find()، حداکثر $\frac{k}{2}$ تا اجداد هم گروه خواهیم داشت چون تو هر عمل find، پدر هر رأس با رأس پدربزرگش عوض می‌شود.

اثبات این موضوع را داریم:

اگر تعداد رأس های با رنگ k حداکثر $\Longleftarrow \{r,\dots,2^{r}-1\}$ اندازه $= \frac{n}{2^r}+\frac{n}{2^{r+1}}+\dots\frac{n}{2^{2^r-1}} \leq \frac{2n}{2^r}$

حالا هزینه m تا عمل find را حساب می‌کنیم و پیچیدگی زمانی کل را بر m تقسیم می‌کنیم تا هزینه سرشکن find بدست آید.

حالا برای هزینه سرشکن پس هزینه حرکات مختلف را بررسی می‌کنیم:

1) هزینه حرکاتی که به ریشه ختم می‌شوند با توجه به اینکه فقط حرکت آخر به ریشه ختم می‌شود: $O(m)$

2) هزینه حرکات بیرون گروهی که به ریشه ختم نمی‌شوند با توجه به اینکه تعداد گروه‌ها حداکثر $\log^* n$ است: $O(m\log^* n)$
و به $\log^* n$ وقت ماکسیمم نیازه

3) هزینه حرکات درون گروهی که به ریشه ختم نمی‌شوند را بررسی می‌کنیم:

هزینه را زمانی که از رأس u به v_1 در گروهی مثل گروه $\{r,\dots,2^{r}-1\}$ قرار دارد، حساب کنیم.

فرض می‌کنیم در یک گروه، n تا رأس v_n ، $v_1,\dots,v_n$ در کنار رأس u قرار دارند، چون در هر find، پدر رأس‌ها با پدربزرگش عوض می‌شود و همچنین تعداد رئوس با رنگ r نیز حداکثر $\frac{n}{2^r}$ است، طول گروه حداکثر می‌تواند طول بازه‌ی $\{r,\dots,2^{r}-1\}$ باشد

و نیز داریم که حداکثر $\log^* n$ گروه حداکثر داریم پس:

$$\sum_{k=0}^{\log^* n}\left(\frac{n}{2}+\frac{n}{4}+\dots\right) = \sum_{k=0}^{\log^* n}\sum_{r=(2\uparrow k)+1}^{2\uparrow(k+1)}\frac{n}{2^r}\left(2\uparrow(k+1)\right) \geq \text{هزینه حرکات درون گروهی که به ریشه ختم نمی‌شوند}$$

$$n\log^* n = \sum_{m=0}^{\log^* n} n \geq \text{هزینه حرکات درون گروهی} \dots \Longleftarrow$$

$\Longrightarrow$ هزینه حرکات درون گروهی که به ریشه ختم نمی‌شوند $= O(n\log^* n)$

حالا پس برای هزینه سرشکن بدست می‌آوریم:

$$\text{هزینه سرشکن} = \frac{(1)+(2)+(3)}{m} = \frac{O(m)+O(n\log^* n)+O(n\log^* n)}{m}$$

$\Longrightarrow$ هزینه سرشکن $= O(\log^* n)$

حالا لم های مورد نیاز در سوال را اثبات می‌کنیم:

(1) شماره گروه ها حداکثر $\log^* n$ است.
فرض کنیم بیشترین شماره m باشد، یعنی رأسی وجود دارد که رنگ آن حداقل $1+2\uparrow m$
است، طبق (3) بدست می آید:

$$\frac{n}{2^{(1+2\uparrow m)}} \geq 1 \Rightarrow n \geq (2\uparrow m+1) \geq (2\uparrow m) \Rightarrow \log^* n > m$$

(2) اثبات اینکه در مسیر find، دنبالهٔ رنگ های رئوس طی شده صعودی اکید است.
داریم $rank(parent(a)) > rank(a)$ چون وقتی union یک رأس، رأس دیگر می شود،
رنگ پدر از رنگ فرزند بیشتر می شود؛ پس بعد از find هم رنگ پدربزرگ از رنگ نوه
بیشتر است.

(3) اثبات اینکه تعداد رئوس با رنگ r، حداکثر $\frac{n}{2^r}$ است. (در ابتدا رنگ همه رئوس 0 بوده)
با استقرا روی r ثابت می کنیم اگر $r = rank(n)$ باشد، زیردرخت فرزندان خود n،
حداقل 2^r رأس دارد.
حکم ما برای $r=0$ درست می باشد.
اگر رنگ رأس قبل n برای اولین بار r بشود، یعنی قبل از آن $r-1$ بوده
که طبق فرض استقرا، حداقل تعداد رئوس زیر درختش 2^{r-1} هستند.
حالا یک رأس z با رنگ $r-1$، فرزند n شده است که $rank(n)$ را یک واحد
زیاد کرده ایم. پس تعداد رئوس زیردرخت فرزندان n حداقل $2^r = 2^{r-1} + 2^{r-1}$ است.
حالا اگر در کل m رأس با رنگ r داشته باشیم، داریم:

$$n \geq 2^r \times m \Rightarrow \frac{n}{2^r} \geq m$$