# آموزش:الگوریتم:مجموعه‌های مجزا [المپیا]

## مجموعه‌های مجزا

### مقدمه

در علم کامپیوتر یک داده ساختار از مجموعه های مجزا، اطلاعات تعدادی داده را که در تعدادی مجموعه مجزا تقسیم شده اند را نگه می دارد. سه عملیاتی که روی این داده ساختار تعریف می شود عبارتند از: پیدا کردن اینکه یک عنصر در کدام مجموعه قرار گرفته است (find) ، ترکیب کردن دو تا از مجموعه های مجزا (union) و افزودن یک مجموعه جدید شامل تنها یک عضو (make set).

## روش اول (با استفاده از لیست پیوندی)

در این روش هر مجموعه را با یک لیست پیوندی نمایش می دهیم که عناصر موجود در آن مجموعه در آن لیست پیوندی قرار دارند. همچنین برای عناصر موجود در لیست ها یک آرایه تعریف می کنیم که خانه i ام آرایه اشاره گر به لیستی است که عنصر i ام قرار دارد. به این ترتیب برای عملیات make set کافیست یک لیست جدید شامل یک عنصر بسازیم و همچنین یک خانه به آرایه اضافه کنیم که اشاره گر به لیست جدید باشد. برای عملیات find هم کافیست مقدار خانه i ام آرایه که اشاره گر به لیستی است که عنصر i ام در آن قرار دارد را بر گردانیم. برای عملیات union هم بدین ترتیب عمل می کنیم که آن مجموعه ای که اعضای کمتری دارد را درون مجموعه ای که اعضای بیشتری دارد قرار می دهیم. یعنی تک تک اعضای لیست پیوندی کوچک را به لیست پیوندی بزرگ اضافه کرده و اشاره گر آنها را نیز به روز رسانی می کنیم. حال نکته ای که این کار دارد این است که به ازای یک عنصر، زمانی که از لیست کوچک به لیست بزرگتری فرستاده می شود، سایز لیست جدیدش حداقل دو برابر لیست قدیمی اش است. بنابراین اگر تعداد کل عناصر n باشد، هر عنصر در تمام عملیات ها حداکثر logn بار به یک لیست جدید اضاقه می شود. ( چرا؟ ) بنابراین اگر m تعداد کل عملیات ها از هر سه نوع باشد، این روش از مرتبه زمانی O(m + nlogn) می باشد.

## روش دوم (جنگل)

در این روش هر مجموعه را بصورت یک درخت از اعضایش در نظر گرفته که یکی از اعضا ریشه می باشد. بنابراین همه ی مجموعه ها در کنار هم تشکیل یک جنگل را می دهند. برای هر عنصر par[x] را تعریف می کنیم یکی از پدران عنصر x در درخت. ( در واقع ما برای یک راس به دنبال ریشه درختی که آن راس در آن قرار گرفته هستیم و با استفاده از آرایه par این کار را انجام می دهیم.) همچنین برای هر مجموعه یک rank تعریف کرده که توضیح آن در ادامه آمده است. به این ترتیب برای عملیات make set یک خانه به آرایه par اضافه کرده و مقدار آن خانه را برابر شماره آن خانه قرار می دهیم زیرا یک درخت تک راسی به جنگل خود اضافه کرده ایم. همچنین یک خانه با مقدار ۰ به آرایه rank اضافه می کنیم.

```
(function MakeSet(x
  x.parent := x
  x.rank   := 0
```

برای ترکیب دو مجموعه که با دو عضوشان به ما داده شده اند، ابتدا ریشه درختی که آن دو عضو در آن هستند را find می کنیم. اگر دو مقدار برابر بدست آوردیم یعنی آن دو عضو از ابتدا در یک مجموعه بوده و نیاز نیست کاری بکنیم. اگر اینگونه نبود، آن مجموعه که rank کمتری دارد را به آن مجموعه که rank بیشتری دارد اضافه می کنیم.

```
(function Union(x, y
 (xRoot := Find(x
 (yRoot := Find(y
if xRoot == yRoot
      return
```

```
                                    .x and y are not already in same set. Merge them //
                                                        if xRoot.rank < yRoot.rank
                                                         xRoot.parent := yRoot
                                                    else if xRoot.rank > yRoot.rank
                                                         yRoot.parent := xRoot
                                                                              else
                                                         yRoot.parent := xRoot
                                                    xRoot.rank := xRoot.rank + 1
```

برای عملیات find هم که قرار است ریشه درختی که عنصر مورد نظر در آن قرار دارد را پیدا کند، مطابق این تابع عمل می کنیم و همانطور که مشاهده می شود به ازای عناصری که در تابع بازگشتی زیر به آنها برخورد می کنیم، مقدار par همگی آنها را به روز رسانی میکنیم.

```
                                                          (function find(x
                                                    if x.parent != x
                                              (x.parent := Find(x.parent
                                                     return x.parent
```

اثبات می شود در این الگوریتم، زمان انجام هر عملیات به طور میانگین از O(alpha(n)) می باشد که در اینجا alpha تابع آکرمن می باشد که می توان گفت برای تمام n هایی که در مسایل با آنها سر و کار داریم کمتر از ۵ می باشد.

## کاربردها

از این مساله در مساله همبندی گراف و همچنین الگوریتم کروسکال برای محاسبه MST در یک گراف استفاده می شود.