آنالیز الگوریتم (۲۲۸۹۱)

مدرس: حسین بومری

[پاییز ۹۹]

تمرین ۱: تقسیم و حل، حریصانه و مرور ساختمان داده

نگارنده: ۹۷۱۰۰۵۷۹ فرزین نصیری

مراسم بزرگ

هدف این است که بازه‌های داده شده را در کمینه دسته‌های ممکن دسته‌بندی بکنیم به طوری که در هر دسته هیچ دو بازه‌ای باهم تداخل نداشته باشند. این مسئله را با استفاده از روش حریصانه [1] حل خواهیم کرد.

تعریف ۱. *موارد زیر را تعریف میکنیم:*

s_i *: شروع بازه*

e_i *: پایان بازه*

$T[j]$ *: آرایه شامل تمام بازه‌های داده شده با اندازه* n

الگوریتم ۱. *الگوریتم به صورت زیر است:*

ابتدا تمام بازه‌های داده شده را بر اساس زمان شروع بازه s_i *به صورت صعودی (زودترین در مکان اول) مرتب میکنیم. اینکار را می‌توان در زمان* $O(n \log n)$ *انجام داد.*

همچنین زمان پایان آخرین بازه اختصاص داده شده به یک سالن را در یک $priority\ \ queue$ *نگه‌داری میکنیم. با توجه به اینکه این ساختمان داده از* $heap$ *استفاده میکند، دسترسی‌های ما با عمل* $heapify$ *به* $O(\log n)$ *زمان نیاز دارد.*

برای هر بازه از آرایه T *که بالاتر تعریف کرده و سپس مرتب کردیم، بررسی میکنیم که در کدام سالن باید قرار بگیرد. شرط قرار گرفتن در یک سالن این است که با هیچ بازه‌ی دیگری که قبلا به سالن اختصاص داده شده تداخل نداشته باشد که یعنی زمان شروع آخرین عضو اضافه شده* s_j *باید از پایان آخرین عضو قبلی* e_{j-1} *نا کمتر باشد.*

[1] Greedy

Scheduling($Interval[]\ T, PrioQueue\ halls$)

```
۱  sort(T)
۲  result = ۰ // number of halls is stored here
۳  for i = ۱ to T.length
۴
۵      if halls.check(T[i])
۶          // T[i] is added to one hall
۷      else
۸          halls.addNewHalls(T[i])
۹          result++
۱۰ return result
```

دقت کنید عملیات check روی صف اولیت halls برسی میکند که آیا بازه داده شده در میتواند در سالنی قرار بگیرد یا خیر (زمان شروع آن بعد از زمان پایان آخرین سخنرانی سالن است) . همچنین عملیات addNewHalls یک سالن جدید را به صف اضافه میکند.

اثبات. اثبات درستی زمان به صورت زیر است:

خط ۱ به حداکثر $O(n \log n)$ زمان برای مرتب کردن نیاز دارد. حلقه خط ۳ n بار اجرا می شود و شرط خط ۵ با توجه مطالب بالاتر به $O(\log n)$ زمان نیاز دارد. در نتیجه کل الگوریتم به $O(n \log n)$ زمان نیاز دارد. □

اثبات. اثبات درستی الگوریتم به صورت زیر است:

برای تضمین کردن کمینه تعداد سالن های باید به این دقت کنیم که برای ساعتی، حداثل به تعداد بازه‌هایی که با آن ساعت تداخل دارند ، نیاز به سالن است. از طرفی مرتب کردن اولیه آرایه باعث میشود الگوریتم ابتدا زمان‌های ابتدایی را برسی کند که در نتیجه حتما کمینه تعداد سالن‌ها را پیدا کردیم.

همچنین با توجه به اینکه الگوریتم ما تضمین میکند که اگر یک بازه زمانی با زمان‌های قبلی در یک سالن تداخل داشته باشد، آن را به سالن اضافه نمی کند. پس در هیچ سالنی دو بازه زمانی که باهم تداخل داشته باشند وجود ندارد. □