جلسه پنجم

13:16 Tuesday, March 2, 2021

موضوع: الگوریتم‌های حریصانه - برنامه ریزی کارها.

یک الگوریتم حریصانه از تعدادی مرحله تشکیل شده و در هر مرحله بهترین انتخاب بر اساس شرایط محلی آن مرحله صورت می‌گیرد.

۱- دایسترا ۲- MST

مثال ۱) خرد کردن پول:

اسکناس ۵ تومانی داده شده است. آن را با سکه‌های ۱ و ۲ و ۵ تومانی خرد کنید. هدف انجام این کار با کمترین تعداد سکه است.

الگوریتم حریصانه: هر مرحله، بزرگترین سکه‌ای که می‌توانیم را انتخاب کنیم.

۲۲ ۱۷ ۱۲ ۷ ۲ ۵

۵ ۵ ۵ ۵ ۲ ⟵ ۱×۲ + ۴×۵

چرا الگوریتم درست است؟

درجواب نهایی

* حداکثر ... سکه ۱ تومانی داریم؟
* حداکثر ... سکه ۲ تومانی داریم؟ → مجموع ۲ تومانی‌ها < ۵
* اگر ۲ تا دو تومانی داشته باشیم ۱ تومانی نداریم؟ → ۲ تومانی‌ها

حداقل ⌊۵/۵⌋ سکه ۵ تومانی داریم.

۱۵ ۱۱ ۵ ۱

۳×۵ ۱×۱۱ + ۴×۱

* آیا همیشه درست است؟

۵۰۰ ۱۰۰ ۵۰ ۲۵ ۱۰ ۵

۴۰ ۲×۲۰

۲۵×۱ + ۱۰×۱ + ۵×۱

مثال ۲) برنامه ریزی کارها

پردازنده‌ای قرار است n کار را انجام دهد. زمان لازم برای انجام کار iام $L[i]$ است. ترتیبی از انجام کارها مشخص کنید که مجموع زمان پایان کارها را کمینه کند.

مثال: $n=3$

$L[1]=5$
$L[2]=2$
$L[3]=7$

۳ ۲ ۱ → ۷ ۹ ۱۴ = ۳۰

۱ ۲ ۳ → ۵ ۷ ۱۴ = ۲۶

۲ ۱ ۳ → ۲ ۷ ۱۴ = ۲۳

$\frac{5}{1}$ $\frac{51}{51}$ $\frac{50}{1}$ →

راه‌حل) کارها را به ترتیب میزان $L[i]$ انجام بده [کمتر به بیشتر]

← در هر مرحله کار با کمترین میزان $L[i]$ را انتخاب کنید.

چرا این راه‌حل درست است؟ برهان خلف

* فرض کنید Π جایگشت بهینه $\Pi = \pi_1\ \pi_2\ \pi_3 \dots \pi_n$

بهینه: π_1 π_2 π_3 ... π_n

زمان اتمام کار π_i ؟ $\sum_{j=1}^{i} L[\pi_j]$

* اگر به ازای هر i داشته باشیم $L[\pi_i] \le L[\pi_{i+1}]$ ← دنباله براساس $L[\pi_i]$ مرتب شده است.

پس به ازای حداقل یک i، $L[\pi_i] > L[\pi_{i+1}]$

* π_i و π_{i+1} را جابه جا کنید!

OPT → π_1 π_2 ... π_i π_{i+1} ... π_n

با این جابه جایی

* زمان اتمام کارهای قبل از π_i: تغییر نمی کند

با این جابه جایی

* زمان اتمام کارهای قبل از Π_i : تغییر نمی کند
* زمان اتمام کارهای بعد از Π_{i+1} : تغییر نمی کند.

زمان اتمام کار Π_{i+1} ؟

← به اندازه $L[\Pi_i]$ کم می شود.

زمان اتمام کار Π_i

← به اندازه $L[\Pi_{i+1}]$ زیاد می شود

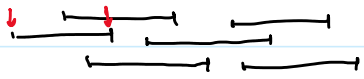
⇓

کل زمان اتمام کارها با این جابه جایی کم می شود.

⇐ این با بهینه بودن OPT در تناقض است.

مثال ۳) n تا بازه داریم. شروع بازه ام : s_i

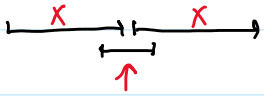
پایان بازه ام : t_i



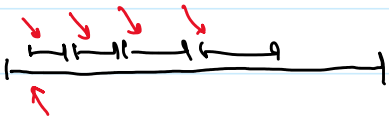
هدف انتخاب یک زیرمجموعه S از بازه ها است که هیچ دو بازه ای از S تلاقی نداشته باشند و اندازه S بیشینه شود.

راه حل های پیشنهادی :

Alg1 : در هر مرحله بازه ای را انتخاب کنیم که طول آن کمینه است و با بازه های انتخاب شده قبلی تلاقی ندارد.

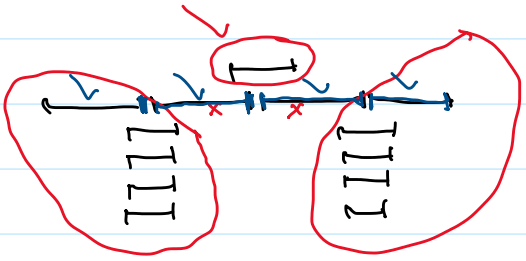


Alg2 : در هر مرحله بازه ای را انتخاب کنیم که شروع آن کمینه است و با بازه های انتخاب شده قبلی تلاقی ندارد.



Alg3 : در هر مرحله بازه ای را انتخاب می کنیم که کمترین تلاقی با بازه های باقی مانده

Alg3 : در مرحله بازه ای را انتخاب می‌کنیم که کمترین تلاقی با بازه های باقی مانده را داشته باشد و با بازه های انتخاب شده قبلی تلاقی نداشته باشد.



بهینه : ۴

→ ۳

Alg۴ : در هر مرحله کار با کمترین مقدار t_i را انتخاب کنیم " " " "

اثبات درستی : فرض کنید OPT مجموعه بازه‌های جواب بهینه باشد.

i_1 i_2 با ۳ ... i_m (مرتب شده برحسب انتها)

$i_{m'}$ $i_{m'+1}$

$m > m'$

فرض کنید جواب الگوریتم ۴ : j_1 j_2 j_3 ... $j_{m'}$

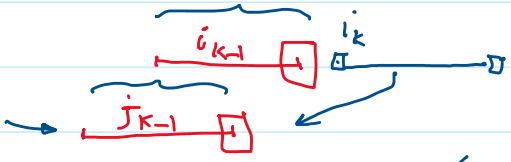
فرض کنید $m' < m$.

* به ازای هر k ، انتهای بازه j_k کمتر یا مساوی انتهای بازه i_k است.

استقرا : به ازای $k=1$

فرض کنید به ازای همه مقادیر کمتر از k برقرار است.

← به ازای k ؟



در زمان اجرای مرحله k الگوریتم امکان انتخاب i_k را داشتیم

و چون کمینه پایان را انتخاب کردیم : انتهای j_k از انتهای i_k

← $i_{m'+1}$ را می‌توانستیم در Alg۴ انتخاب کنیم.

← این با تمام شدن الگوریتم ۴ در تناقض x

و $m = m'$