Algorithms

DATE: 15/03/2022

GREEDY ALGORITHMS

$\text{SELECT}_M(I)$: ← INPUT, INSTANCES

$S = [\ ]$

WHILE $|I| \geq 1$:

$O(|I|)$ PICK $(s_j, f_j) \in I$ according to rule M

$N = \{(s_i, f_i) \mid (s_i, f_i) \in I$ AND S.T. (s_i, f_i) and (s_j, f_j) ARE INCOMPATIBLE $\}$ ← IT CONTAINS ALL THE REMAINING INTERVALS INCOMPATIBLE with (s_j, f_j)

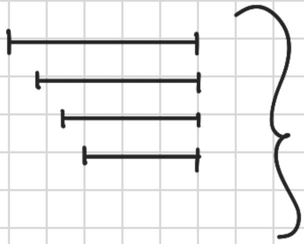
$I -= N$ // I and N are sets $O(|I|)$

$O(1)$ S. APPEND$((s_j, f_j))$ // appending compatible intervals

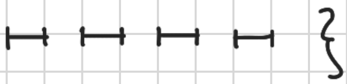
RETURN S

M = "PICK THE INTERVAL THAT ENDS SOONEST"

IF n is the number of input intervals



with this input one iteration is sufficient.



with this input n. the iterations are necessary

OBS : The number of iterations of the while loop is $\leq n$.

P : At least one interval is removed from I in the generic iteration

With simple data structure ("I" is just a linked list), the generic iteration takes:

$$O(|I|) + O(|I|) + O(1) = O(\max(|I|, |I|, 1)) = O(|I|)$$

Let $I_0 = I$ be the input set of intervals ($|I| = |I_0| = n$ intervals)

Let I_i be the value of I after the ith iteration of the while loop.

$|I_0| = n$

we always cut AT LEAST one interval at every iter.

$|I_0| > |I_1| > |I_2| > \dots > |I_t| = 0$ (IF THE LOOP ITERATES T TIMES)

$|I_0| \geq |I_1| + 1$

$|I_1| \geq |I_2| + 1$

$\vdots$

$|I_{t-2}| \geq |I_{t-1}| + 1$

$|I_{t-1}| \geq |I_t| + 1 = 1$

$|I_t| = 0$

$1 + 2 + 3 + 4 + 5 + 6$

$\frac{n}{2}(n+1)$

$$\sum_{i=1}^{n} i = \frac{(n+1)\cdot n}{2} = \frac{n^2+n}{2} = O(n^2)$$

$|I_{t-3}| \geq |I_{t-2}| + 1 \geq 2+1=3$

$|I_{t-j}| \geq j$

$n^2 \geq \frac{n^2+m}{2}$

This implies that t cannot be larger than n.

Total runtime $= \sum_{i=0}^{t-1} O(|I_i|) \leq O(|I_0|) + O(|I_1|) + \dots + O(|I_{t-1}|) \leq O(n + (n-1) + (n-2) + \dots + 2 + 1) =$

$= O\left(\sum_{i=1}^{n} i\right) = O(n^2)$

FASTALG (I):

$O(n \lg n)$ – Sort the intervals in I increasingly by finishing time.

RUNTIME OF MERGE-SORT

– Let $I = \{I_1, \dots I_n\}$ with $f(I_1) \leq f(I_2) \leq f(I_n)$

we set this so the first test is always true / it works also with empty interval

$O(1)$ - Set $T \leftarrow -\infty$, $S = \emptyset$

- For $i = 1, \dots, m$
  - IF $s(I_i) > T$ — STARTING TIME OF I_i

    $S = S \cup \{I_i\}$
    $T = f(I_i)$

    $O(1)$
- RETURN S

Let $T \leftarrow f(I_1)$, $S \leftarrow \{I_1\}$

For $i = 2, 3, \dots, n$

EX: Prove that fastalg returns an optimal solution to interval scheduling.
(HINT: show that it behaves like SELECT_H with $H =$ "EARLIEST TO FINISH")

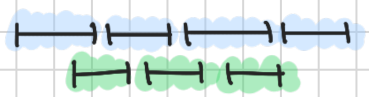
L: The runtime of fast alg. is $O(n \log n)$

P: $O(n \log n) + n \cdot O(1) \leq O(n \log n)$

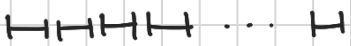
INTERVAL PARTITIONING

We are given a set of intervals I.

We aim to schedule each interval on the minimum possible number of resources

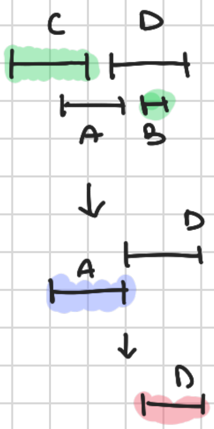


TO SCHEDULE EACH INTERVAL, I NEED 2 RESOURCES



HERE, ONE RESOURCE IS SUFFICIENT

$H^* =$ "EARLIEST TO FINISH"

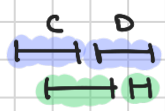


SELECT_{H^*}

$\{C, B\}$

$\{A\}$ → SELECT_{H^*} APPLIED GREEDILY USES 3 RESOURCES

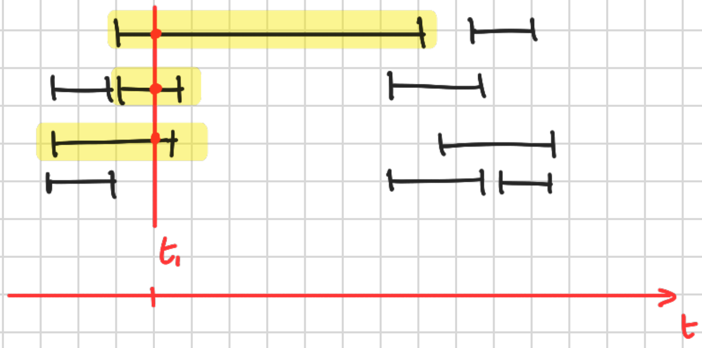
$\{D\}$



BUT 2 RESOURCES ARE OPTIMAL!

1. Find the min. number of resources
2. Find a schedule for them

EXAMPLE:



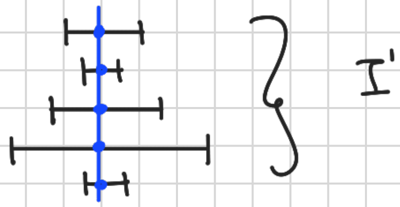
How can we run those 3 intervals at the same time if we don't have 3 resources?

DEPTH(I') = 3

max number of intervals that can run at the same time

DEF: DEPTH (I) is the minimum integer d S.T. $\forall t \in \mathbb{R}$

$$|\{ I_i \mid I_i \in I \wedge t \in I_i \}| \leq d$$



DEPTH (I') = 5

NOTE: the depth will represent the minimum resources we need but we will have to prove it first.

DEF: Let OPT(I) be the minimum number of resources to schedule each interval in I.

L: OPT(I) ≥ DEPTH (I)

P: There must exist a time t when exactly DEPTH (I) intervals are running at the same time.
At time t, we then need DEPTH (I) resources to schedule all the intervals: OPT (I) ≥ DEPTH (I). ■