CIS 575. Introduction to Algorithm Analysis
Material for April 8, 2024

Greedy Event Scheduling

©2020 Torben Amtoft

The topic of this note is presented in *Cormen*'s Section 15.1.

1 Scheduling Fixed Events

We shall now describe, and subsequently solve, another scheduling problem.

Setting We have **one server** which at each time is capable of hosting **one event**.
We also have a number of **events**, each with a **fixed start time** and a **fixed finish time** (no flexibility!)
Our **Goal** is to schedule **as many events as possible** (no event is more important than any other event).

Greedy Strategy, 1st Attempt Based on the previous scheduling problem we considered, a greedy solution may appear obvious:

> Give priority to the **shortest** events

But what is obvious is not always true (that's why we bother so much about writing proofs); consider the situation

```
event 1      XXXXXXXXXXXXXXX
event 2                   XX
event 3                    XXXXXXXXXXXXXX
```

where event 2 is very short, yet prevents any other event from being scheduled, whereas the two long events 1 and 3 do not preclude each other. We conclude that our **1st attempt failed**.

Greedy Strategy, 2nd Attempt Inspired by the above counterexample, we may believe that the problem is that event 2 overlaps with *two* jobs whereas events 1 and 3 each overlaps with only *one* job. Therefore, we may propose

> Give priority to the events with **fewest overlapping** events.

But even this strategy doesn't always work out, as a counterexample exists (it is rather long; you may see if you can find a shorter!)

```
event A    XXXX
event B     XXX
event C      XXXX
event D       XXX
event E        XXXX
event F          XXXX
event G            XXXX
event H              XXXX
event I              XXXX
event J               XXXX
event K                XXXX
```

Event F overlaps with only 2 other events, whereas all other events overlap with at least 3 other events. Hence the proposed strategy will first select F. But then we cannot select event E or event G, and can select at most one from ABCD and one from HIJK, for a total of 3 events. But it is possible to schedule 4 events: AEGK. We conclude that our **2nd attempt failed**.

Greedy Strategy, 3rd Attempt We don't give up! Let us consider the strategy

> Give priority to events that **finish early**.

For our first example, this strategy would first accommodate event 1, then not have room for event 2, but would then accommodate event 3.

For our second example, this strategy would first accomodate event A (or B), then have to skip events C and D but would next accommodate event E, then have to skip event F but would next accommodate event G, then have to skip events H,I,J but would finally accommodate event K.

For both our examples, this 3rd attempt works. But of course, this doesn't provide any guarantee that it *always* works. We need a general proof which we shall now provide.

Theorem: Given a set of events, with start & finish times. Let $E_1 \dots E_n$ be a valid schedule (in increasing order of accommodation time). Then any schedule produced by our strategy (which keeps choosing an event with earliest finish time) will be of the form $E'_1 \dots E'_m$ with $m \geq n$, and E'_n will not finish later than E_n.

Proof: We do induction in n. The case $n = 1$ follows directly from the definition of our strategy. Now assume that $E_1 \dots E_{n+1}$ is a valid schedule. Let S' be a schedule produced by our strategy. Inductively, it will be of the form $E'_1 \dots E'_m$ with $m \geq n$, and E'_n will not finish later than E_n. But that means that E'_n will not finish later than E_{n+1} starts, and thus E_{n+1} is among the events that our strategy may choose from when finding the $n+1$'st event. Hence our strategy can schedule at least $n+1$ events: $m \geq n+1$, and by the way E'_{n+1} is chosen, it does not finish later than E_{n+1}. This completes the proof. $\square$

We conclude that our **3rd attempt succeeded**.

Greedy Strategy, 4th Attempt We may also consider a strategy symmetric to the previous:

Give priority to events that **start late**.

For our first example, this strategy would first accommodate event 3, then not have room for event 2, but would then accommodate event 1.

For our second example, this strategy would first accomodate event K, then have to skip events H,I,J but would next accommodate event G, then have to skip event F but would next accommodate event E, then have to skip events D and C but would finally accommodate event B.

Thus the proposed strategy works for our two examples, and is indeed always optimal, as can be verified by a proof symmetric to what was given for the previous strategy. We conclude that our **4th attempt succeeded**.

Concluding Remarks We proposed four greedy strategies: the first two do not always work, but the last two do always produce an optimal schedule. The 4th strategy illustrates that (as in the real world) the **order of scheduling** may **not equal** the **order of execution**; an event that occurs later may be scheduled *before* an event that occurs earlier.