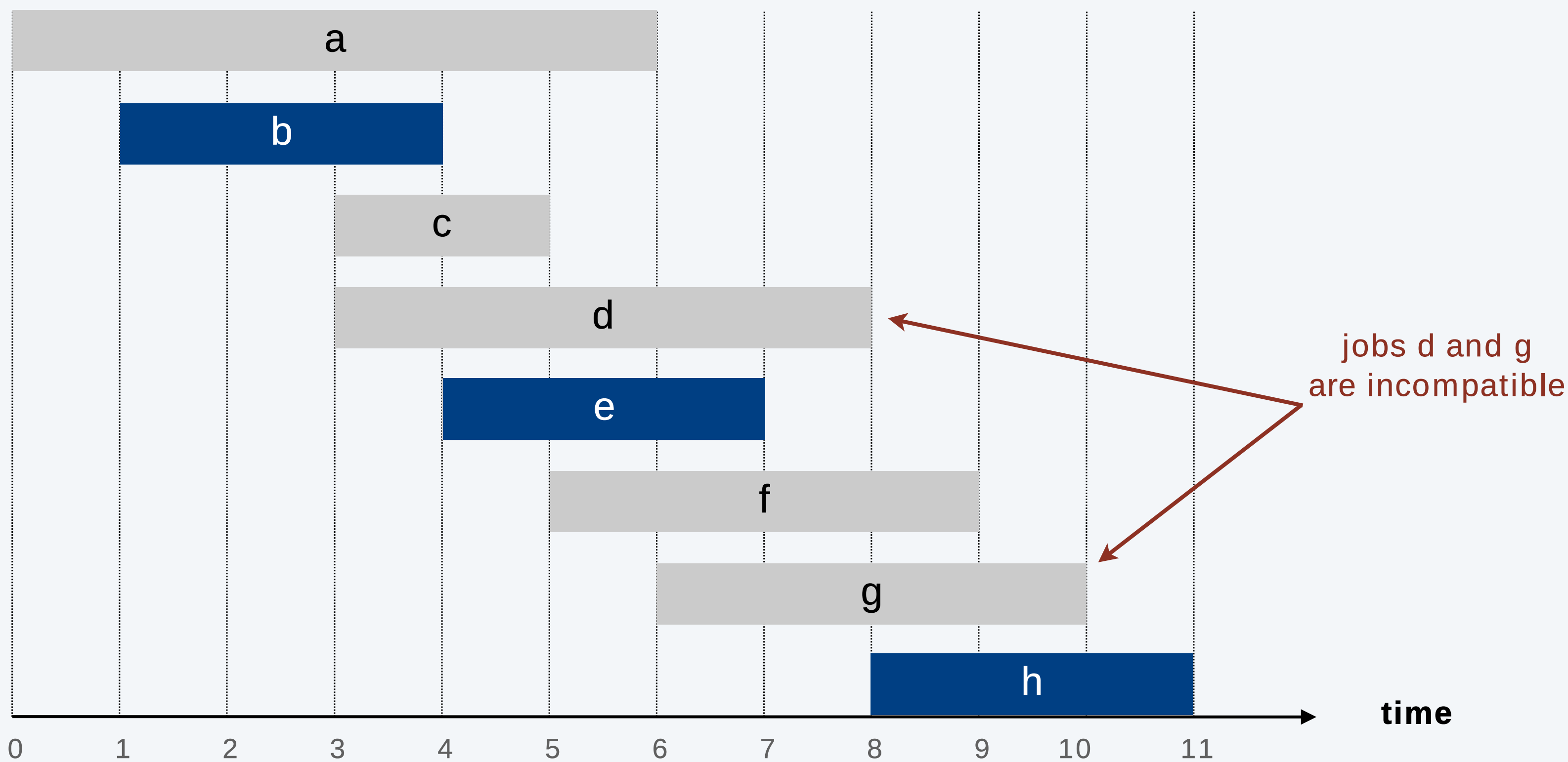
Interval scheduling

- Job j starts at s_j and finishes at f_j.
- Two jobs compatible if they don't overlap.
- Goal: find maximum subset of mutually compatible jobs.



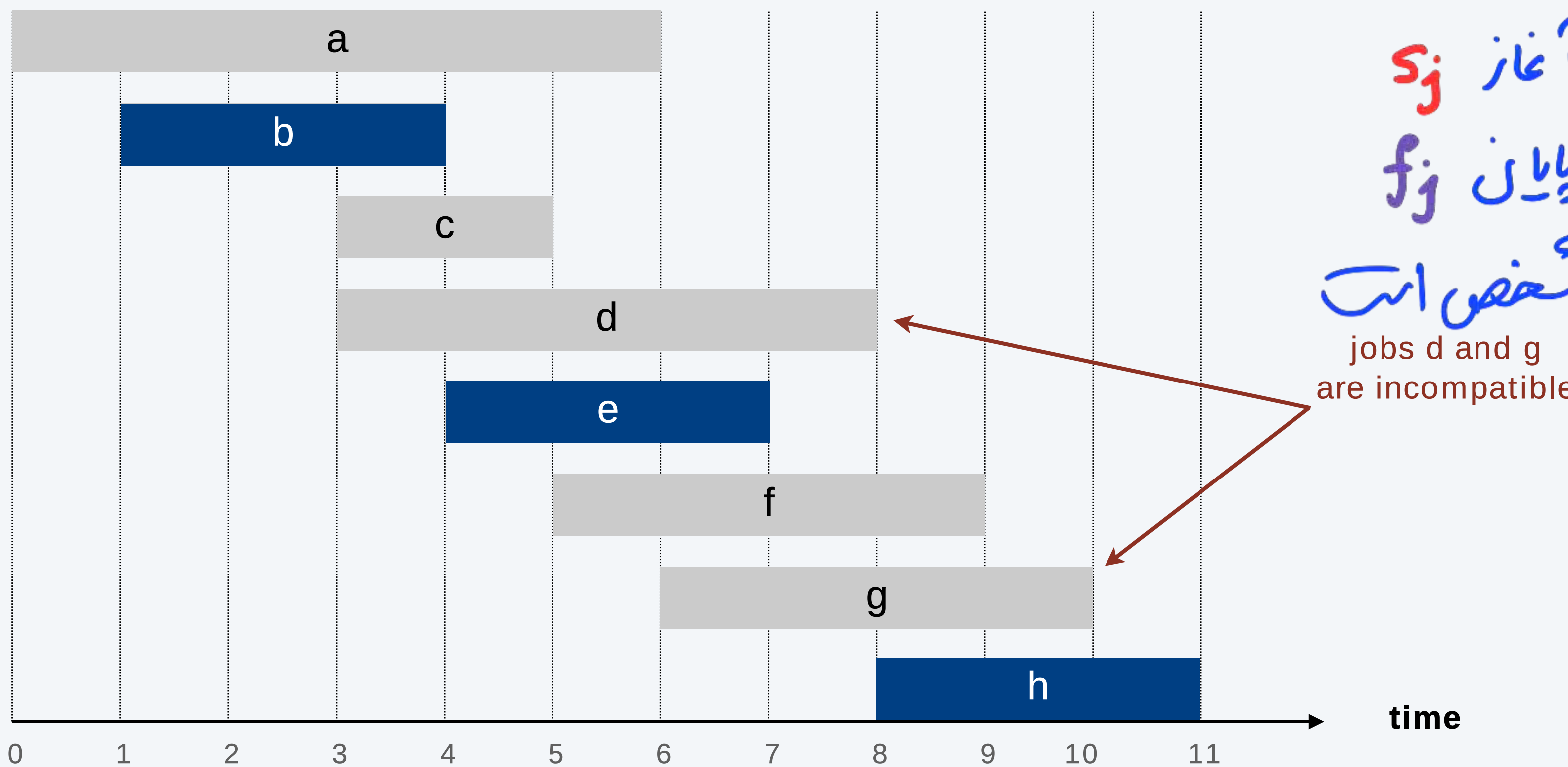
2 Interval scheduling

مسئلهٔ زمان‌بندی بازه‌ای یا
"Activity Selection"
مسئلهٔ انتخاب فعالیت
(CLRS 16.1)

- Job j starts at s_j and finishes at f_j.
- Two jobs compatible if they don't overlap.
- Goal: find maximum subset of mutually compatible jobs.

مسئول هماهنگی تالار سخنرانی: لیستی از تقاضاها برای برگزاری سخنرانی داده شده

در هر تقاضا
زمان آغاز s_j
و زمان پایان f_j
از قبل مشخص است



Interval scheduling: greedy algorithms

در این مسئله بررسی feasibility همانا بررسی ناهمپوشان بودن است.

Greedy template. Consider jobs in some "natural order."
Take each job provided it's compatible with the ones already taken.

1 • [Earliest start time] Consider jobs in ascending order of s_j.

قاعده زودآمدترین

2 • [Earliest finish time] Consider jobs in ascending order of f_j.

* قاعده زودپایان‌ترین

3 • [Shortest interval] Consider jobs in ascending order of $f_j - s_j$.

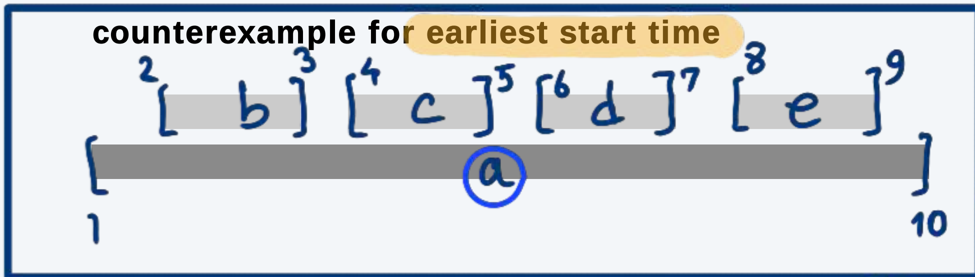
قاعده کوتاه‌ترین زمان

4 • [Fewest conflicts] For each job j, count the number of conflicting jobs c_j. Schedule in ascending order of c_j.

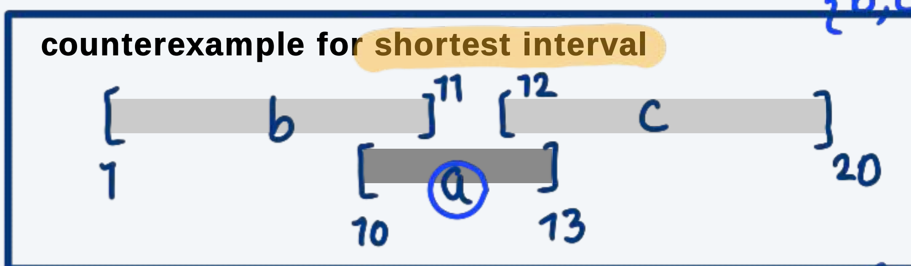
قاعده کم تداخل‌ترین

Interval scheduling: greedy algorithms

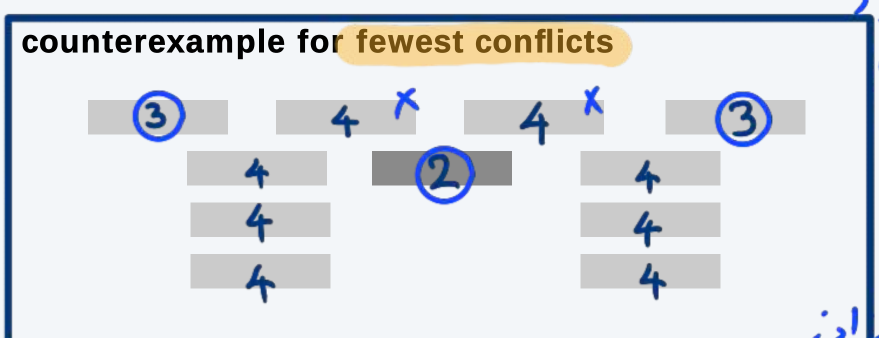
Greedy template. Consider jobs in some natural order.
Take each job provided it's compatible with the ones already taken.



مثال نقض برای قاعده زودآمدترین
⇐ فقط a انتخاب می‌شود
در حالیکه جواب بهینه = {b,c,d,e}



مثال نقض برای قاعده کوتاهترین زمان
⇐ فقط a انتخاب می‌شود
در حالیکه جواب بهینه = {b,c}



مثال نقض برای قاعده کم تداخل‌ترین ⇐ ۳ سخنرانی
در حالیکه جواب بهینه = ۴ سخنرانی

Interval scheduling: earliest-finish-time-first algorithm

EARLIEST-FINISH-TIME-FIRST $(n, s_1, s_2, \ldots, s_n, f_1, f_2, \ldots, f_n)$

SORT jobs by finish time so that $f_1 \le f_2 \le \ldots \le f_n$

$A \leftarrow \phi$ ← set of jobs selected

FOR $j = 1$ TO n

    IF job j is compatible with A

        $A \leftarrow A \cup \{ j \}$

RETURN A

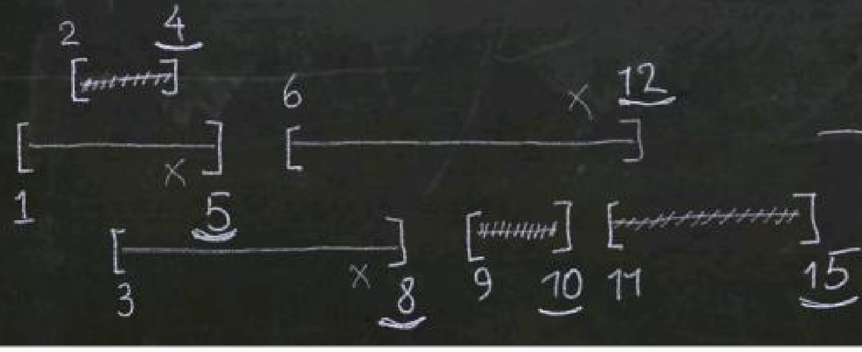
Proposition. Can implement earliest-finish-time first in $O(n \log n)$ time.

- Keep track of job j^* that was added last to A.
- Job j is compatible with A iff $s_j \ge f_{j^*}$.
- Sorting by finish time takes $O(n \log n)$ time.

فرض می‌کنیم پیش از فراخوانی تابع زیر، بازه‌ها به ترتیب صعودی برحسب f مرتب شده

```
GREEDY-ACTIVITY-SELECTOR(s, f)
n = s.length
A = {a_1}            // selected jobs; A
k = 1                // index of last selected job
for m = 2 to n
    if s[m] ≥ f[k]   // feasibility test
        A = A ∪ {a_m}
        k = m
return A
```

می‌توان نشان داد جواب بهینه (حداکثر تعداد بازه‌های دو به دو ناهمپوشان) برای مسئله انتخاب فعالیت توسط معیار انتخاب حریصانه Earliest-finish-time قاعده زود پایان‌ترین بدست می‌آید.



۳ فعالیت انتخاب می‌شود

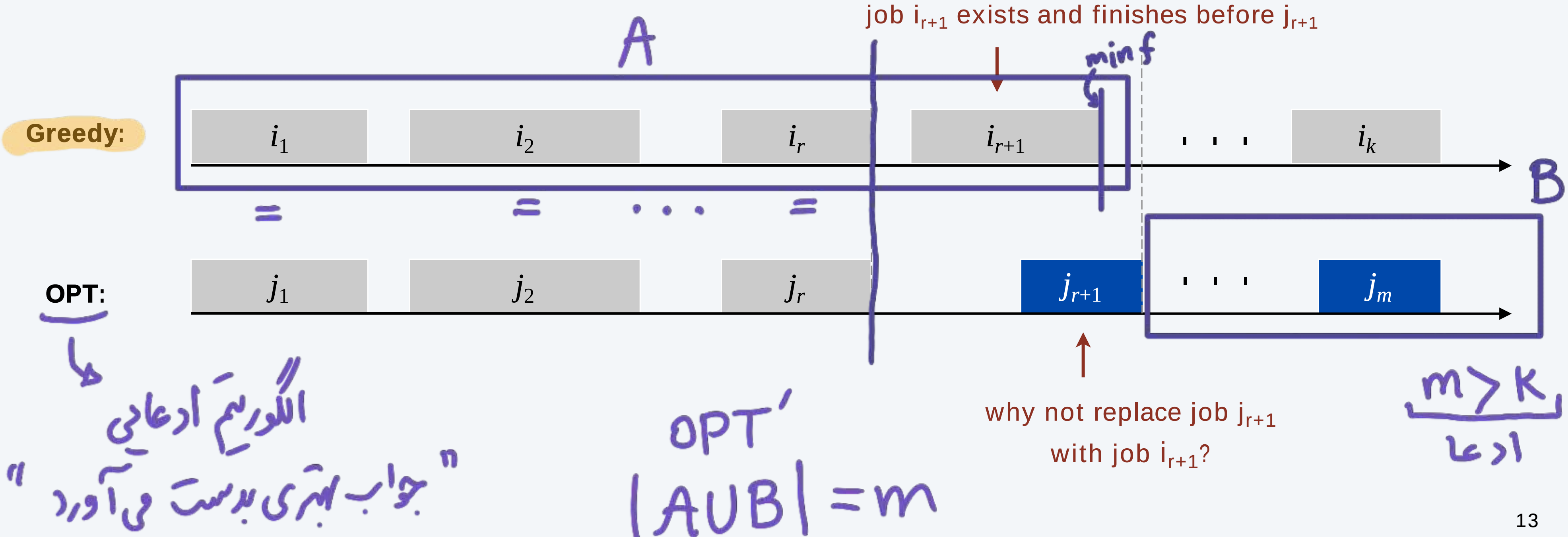
$$T(n) = O(n \lg n) + O(n) = O(n \lg n)$$

Interval scheduling: analysis of earliest-finish-time-first algorithm

Theorem. The earliest-finish-time-first algorithm is optimal.

Pf. [by contradiction]

- Assume greedy is not optimal, and let's see what happens.
- Let $i_1, i_2, \ldots i_k$ denote set of jobs selected by greedy.
- Let $j_1, j_2, \ldots j_m$ denote set of jobs in an optimal solution with $i_1 = j_1, i_2 = j_2, \ldots, i_r = j_r$ for the largest possible value of r. $(r < m)$



OPT'

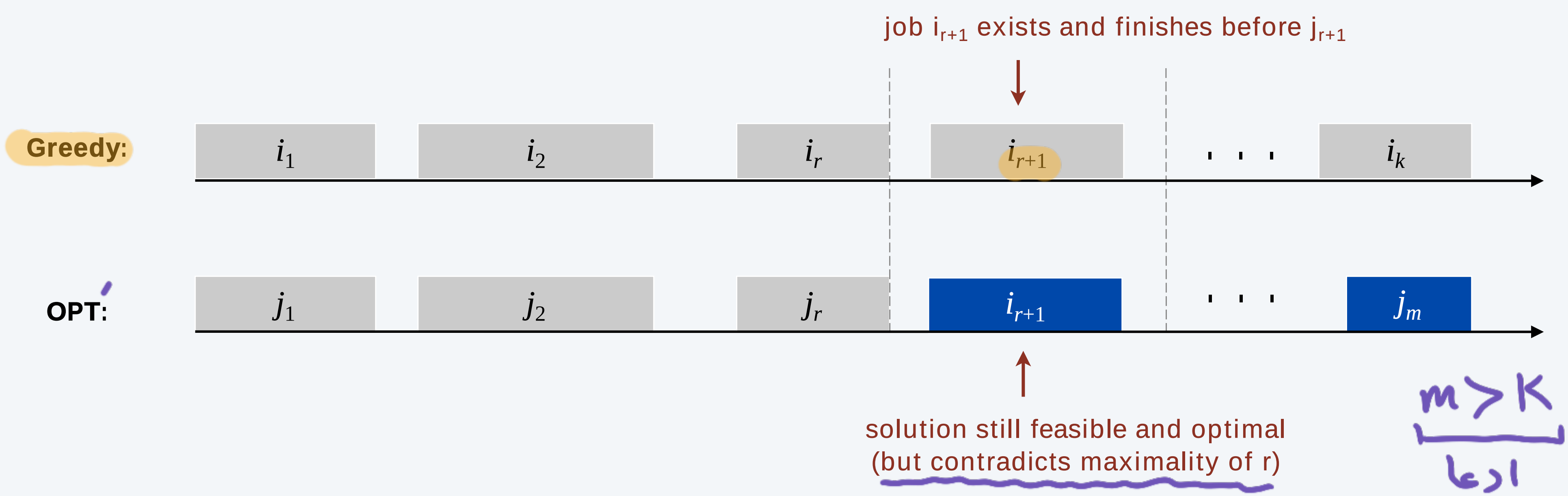
$|A \cup B| = m$

Interval scheduling: analysis of earliest-finish-time-first algorithm

Theorem. The earliest-finish-time-first algorithm is optimal.

Pf. [by contradiction]

- Assume greedy is not optimal, and let's see what happens.
- Let $i_1, i_2, \ldots i_k$ denote set of jobs selected by greedy.
- Let $j_1, j_2, \ldots j_m$ denote set of jobs in an optimal solution with $i_1 = j_1, i_2 = j_2, \ldots, i_r = j_r$ for the largest possible value of r.



3 Interval partitioning

مسئله افراز بازه‌ای

«مسئله کمینه سازی تالارهای سخنرانی»

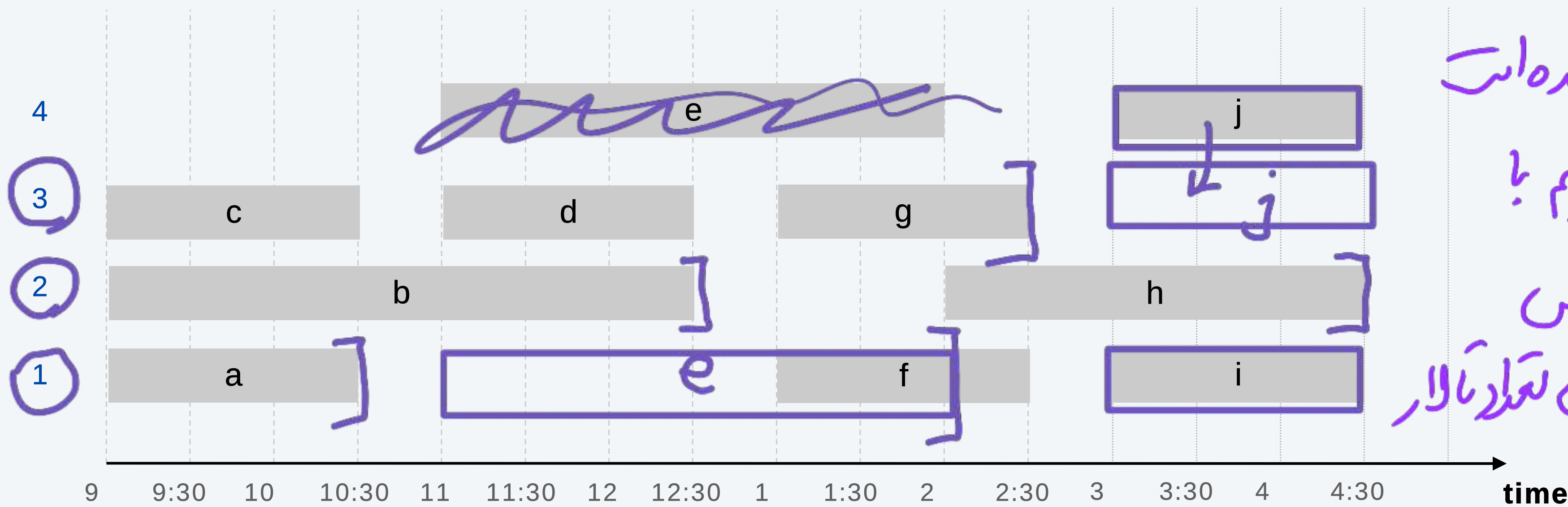
Interval partitioning.

- Lecture j starts at s_j and finishes at f_j.
- Goal: find minimum number of classrooms to schedule all lectures so that no two lectures occur at the same time in the same room.

مشابه مسئله قبل n سخنرانی، با زمان آغاز s_j و زمان پایان f_j داده شده است می‌خواهیم با تخصیص کمترین تعداد تالار

Ex. This schedule uses 4 classrooms to schedule 10 lectures.

$n=10$



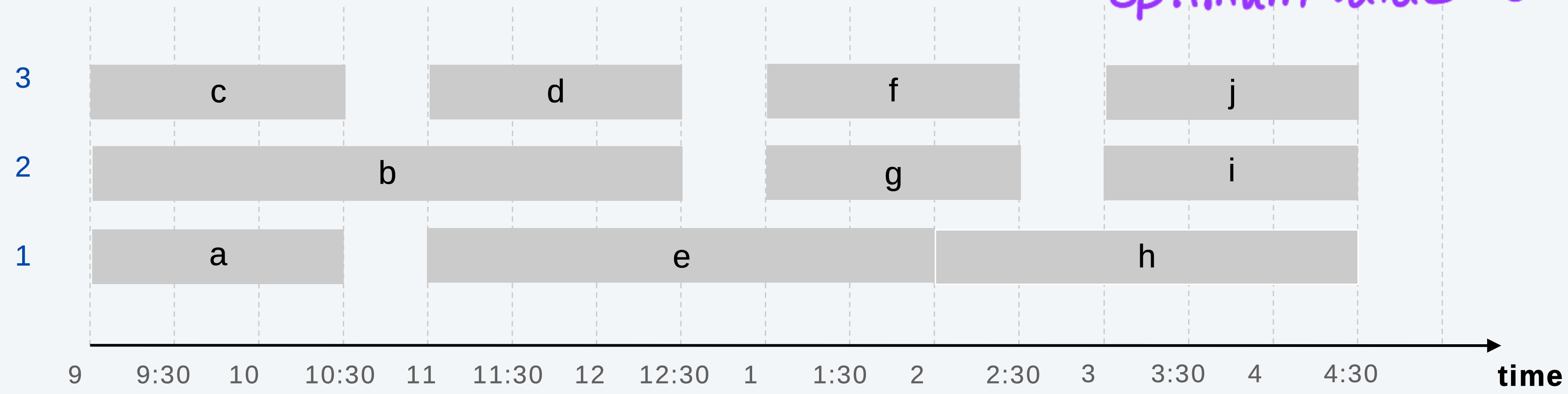
Interval partitioning

Interval partitioning.

- Lecture j starts at s_j and finishes at f_j.
- Goal: find minimum number of classrooms to schedule all lectures so that no two lectures occur at the same time in the same room.

Ex. This schedule uses 3 classrooms to schedule 10 lectures.

An Optimal Solution
optimum value = 3



Interval partitioning: greedy algorithms

Greedy template. Consider lectures in some natural order.
Assign each lecture to an available classroom (which one?);
allocate a new classroom if none are available.

1 • [Earliest start time] Consider lectures in ascending order of s_j. زود آمدترین

2 • [Earliest finish time] Consider lectures in ascending order of f_j. زود پایان ترین

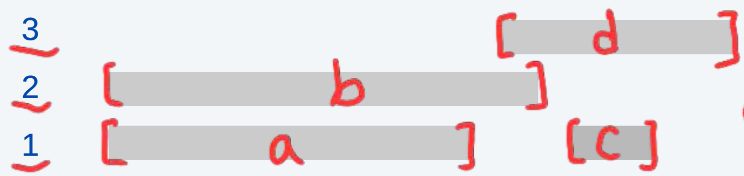
3 • [Shortest interval] Consider lectures in ascending order of $f_j - s_j$. کوتاه‌ترین بازه

4 • [Fewest conflicts] For each lecture j, count the number of conflicting lectures c_j. Schedule in ascending order of c_j. کم تداخل ترین

Interval partitioning: greedy algorithms

Greedy template. Consider lectures in some natural order. Assign each lecture to an available classroom (which one?); allocate a new classroom if none are available.

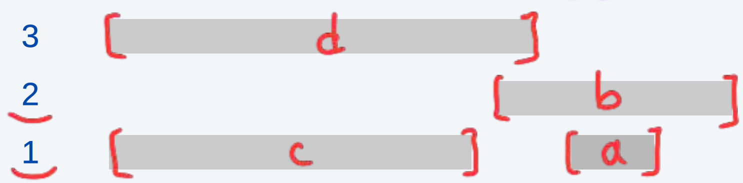
counterexample for earliest finish time



مثال نقض برای انتخاب زودپایان ترین ⇐ 3 تعداد تالارهای مورد نیاز

$\{a, d\}, \{b, c\}$ درحالیکه جواب بهینه ⇐ 2 تالار

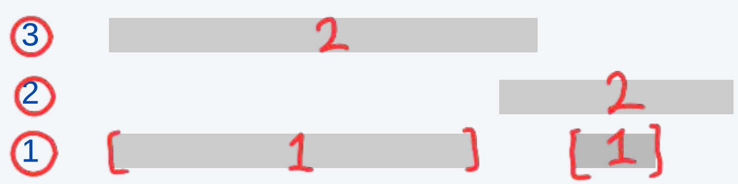
counterexample for shortest interval



مثال نقض برای انتخاب کوتاهترین بازه ⇐ 3 تالار

درحالیکه جواب بهینه ۲ تالار

counterexample for fewest conflicts



مثال نقض برای انتخاب کم تداخل ترین ⇐ 3 تالار

درحالیکه جواب بهینه = ۲ تالار

Interval partitioning: earliest-start-time-first algorithm

EARLIEST-START-TIME-FIRST $(n, s_1, s_2, \ldots, s_n, f_1, f_2, \ldots, f_n)$

SORT lectures by start time so that $s_1 \leq s_2 \leq \ldots \leq s_n$.

$d \leftarrow 0$ ← number of allocated classrooms تعداد تالارهای مورد نیاز

FOR $j = 1$ TO n

  IF lecture j is compatible with some classroom

    Schedule lecture j in any such classroom k.

  ELSE

    Allocate a new classroom $d + 1$.

    Schedule lecture j in classroom $d + 1$.

    $d \leftarrow d + 1$

RETURN schedule.

Interval partitioning: earliest-start-time-first algorithm

Proposition. The earliest-start-time-first algorithm can be implemented in $O(n \log n)$ time.

Pf. Store classrooms in a priority queue (key = finish time of its last lecture).

- To determine whether lecture j is compatible with some classroom, compare s_j to key of min classroom k in priority queue.
- To add lecture j to classroom k, increase key of classroom k to f_j.
- Total number of priority queue operations is $O(n)$.
- Sorting by start time takes $O(n \log n)$ time. ▪

Remark. This implementation chooses the classroom k whose finish time of its last lecture is the earliest.

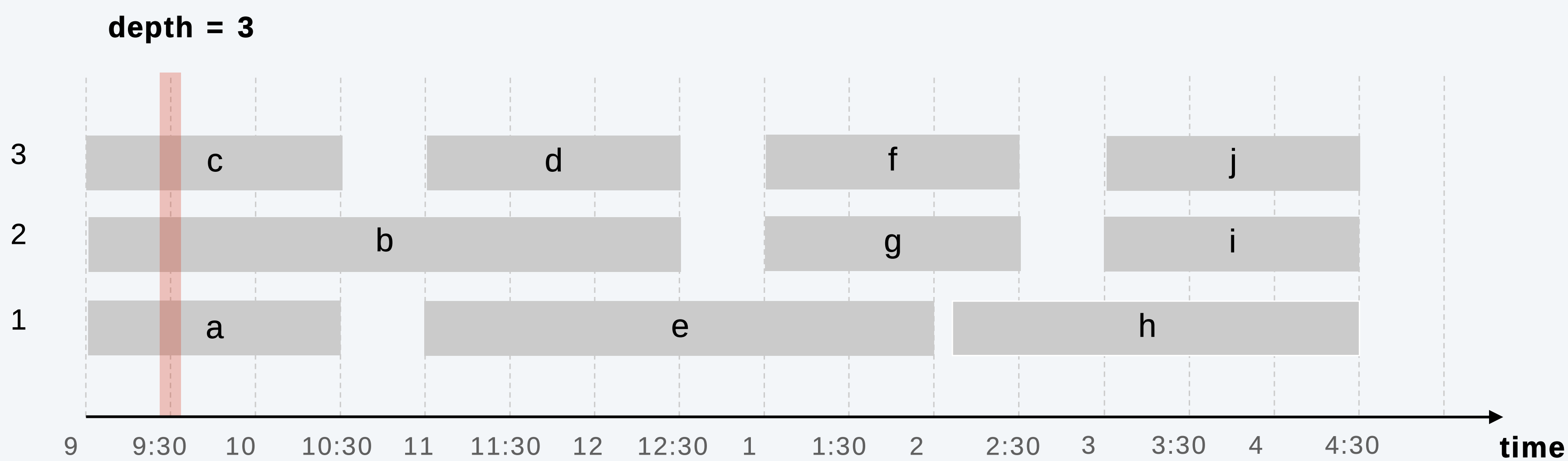
Interval partitioning: lower bound on optimal solution

Def. The depth of a set of open intervals is the maximum number that contain any given time.

Key observation. Number of classrooms needed $\geq$ depth.

Q. Does number of classrooms needed always equal depth?

A. Yes! Moreover, earliest-start-time-first algorithm finds one.



Interval partitioning: analysis of earliest-start-time-first algorithm

Observation. The earliest-start-time first algorithm never schedules two incompatible lectures in the same classroom.

Theorem. Earliest-start-time-first algorithm is optimal.

Pf.

- Let d = number of classrooms that the algorithm allocates.
- Classroom d is opened because we needed to schedule a lecture, say j, that is incompatible with all $d-1$ other classrooms.
- These d lectures each end after s_j.
- Since we sorted by start time, all these incompatibilities are caused by lectures that start no later than s_j.
- Thus, we have d lectures overlapping at time $s_j + \varepsilon$.
- Key observation $\Rightarrow$ all schedules use $\geq d$ classrooms. ▪