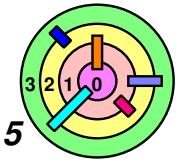
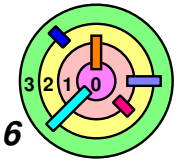
Scheduling in Hard Real Time Systems

- Known chores and durations
  - find schedule satisfying constraints
    - each job has a *completion deadline*
  - uniprocessor
    - rate-monotonic scheduling of *cyclic chores*, i.e., *periodic tasks*
      - $t_i \rightarrow T_i, P_i$ where T is *execution time* and P is *period*
      - e.g., a job is required to execute 2 seconds every 6 seconds
      - deadline is implied
    - Earliest Deadline First (EDF)
      - $t_i \rightarrow d_i$ where d is a *deadline*
      - optimality proof exists under certain conditions
  - multiprocessor
    - often NP-complete ...



Assumptions

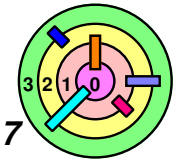
- Interrupts don't interfere (too much) with schedule
  - bounded interrupt delays
- Execution time really is predictable
  - what about effects of caching and paging?



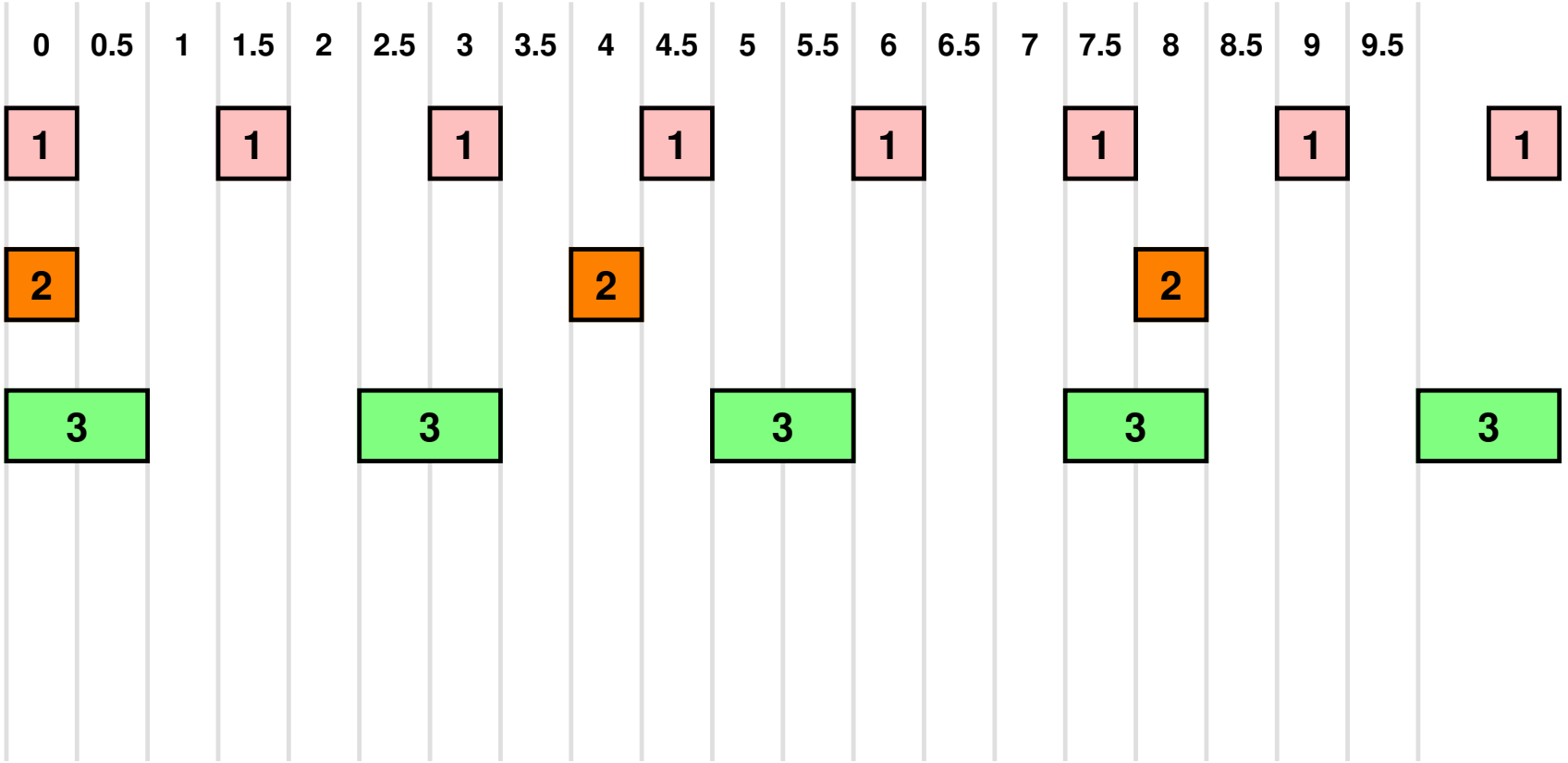
Rate-Monotonic Scheduling

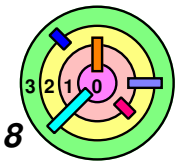
- **Periodic chores**
  - period P_i
  - *per-cycle* processing time $T_i \ (\leq P_i)$
    - T_i / P_i is the *utilization (or "duty cycle")* of chore *i*
  - necessary condition for feasibility: $\sum_{i=0}^{n-1} (T_i / P_i) \leq 1$
- **Rate-monotonic scheduling**
  - each chore *i* is handled by a thread with *priority* $1/P_i$
  - *preemptive*, *priority scheduling*
  - works when $\sum_{i=0}^{n-1} (T_i / P_i) \leq n(2^{1/n}-1)$
  - $= \ln 2 \approx 0.6931$ in the limit

n	$n(2^{1/n}-1)$
1	1
2	0.8284
3	0.7798
4	0.7568
5	0.7435
...	...

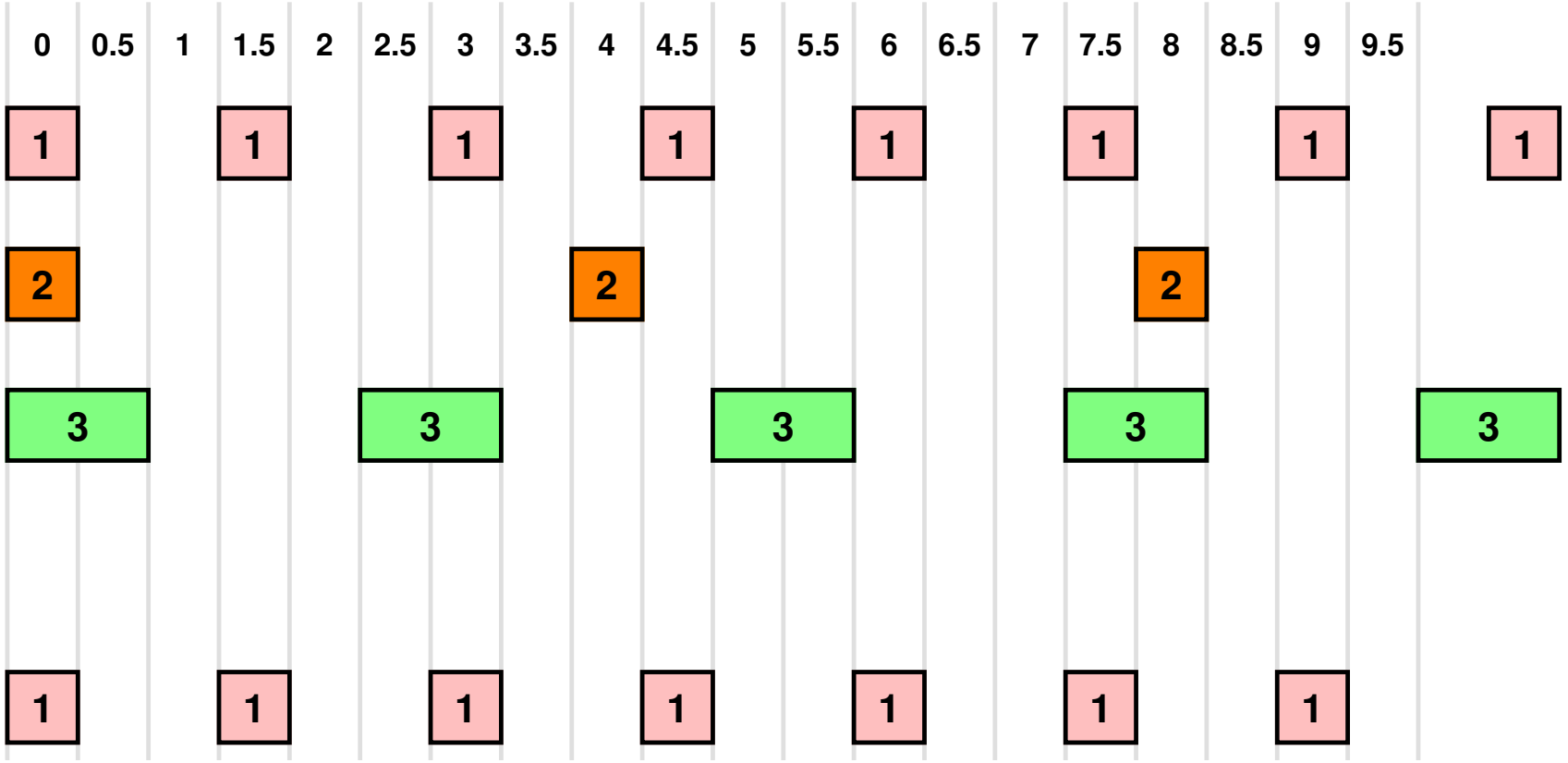


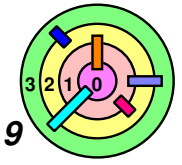
Scenario 1



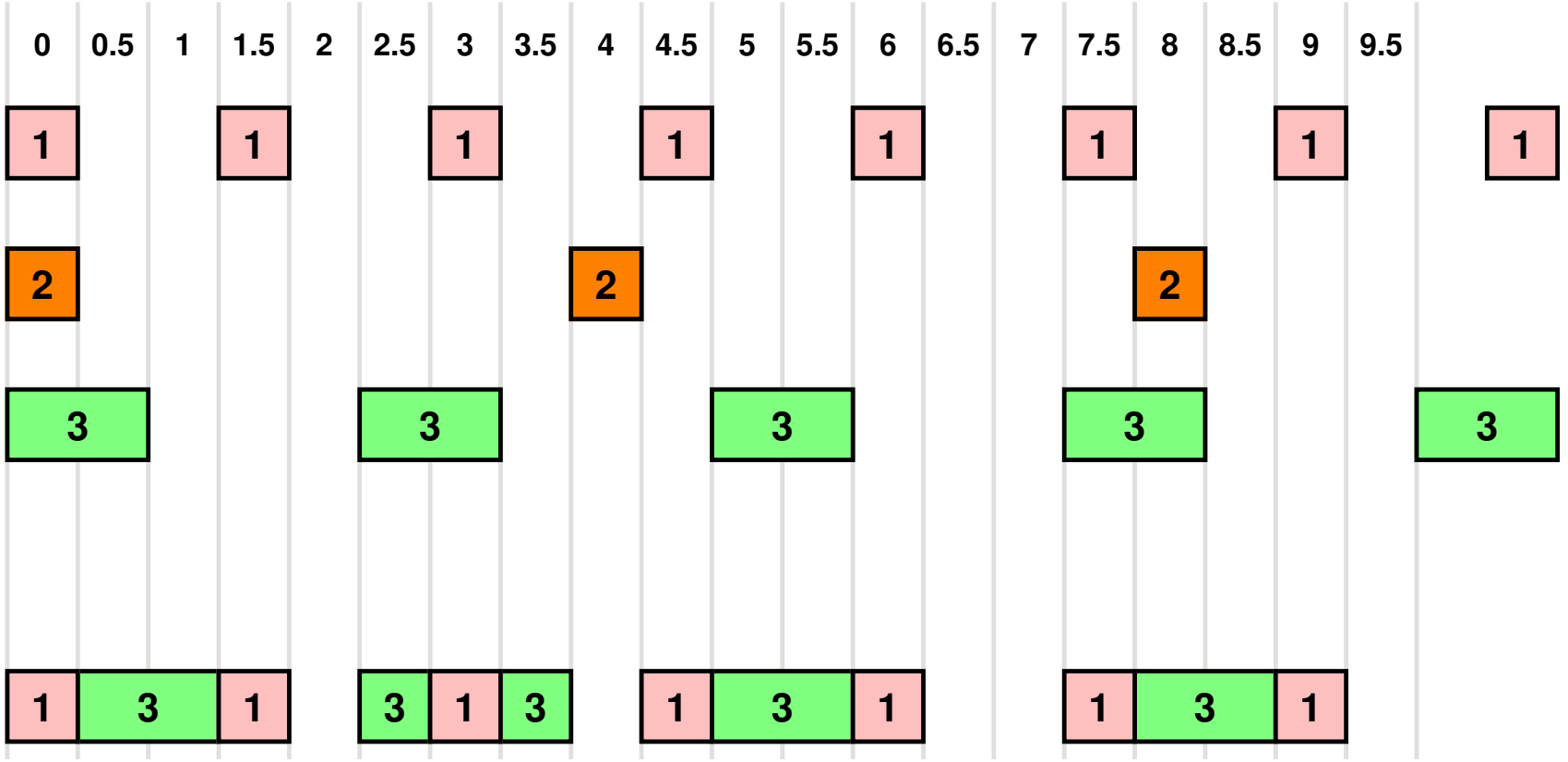


Scenario 1

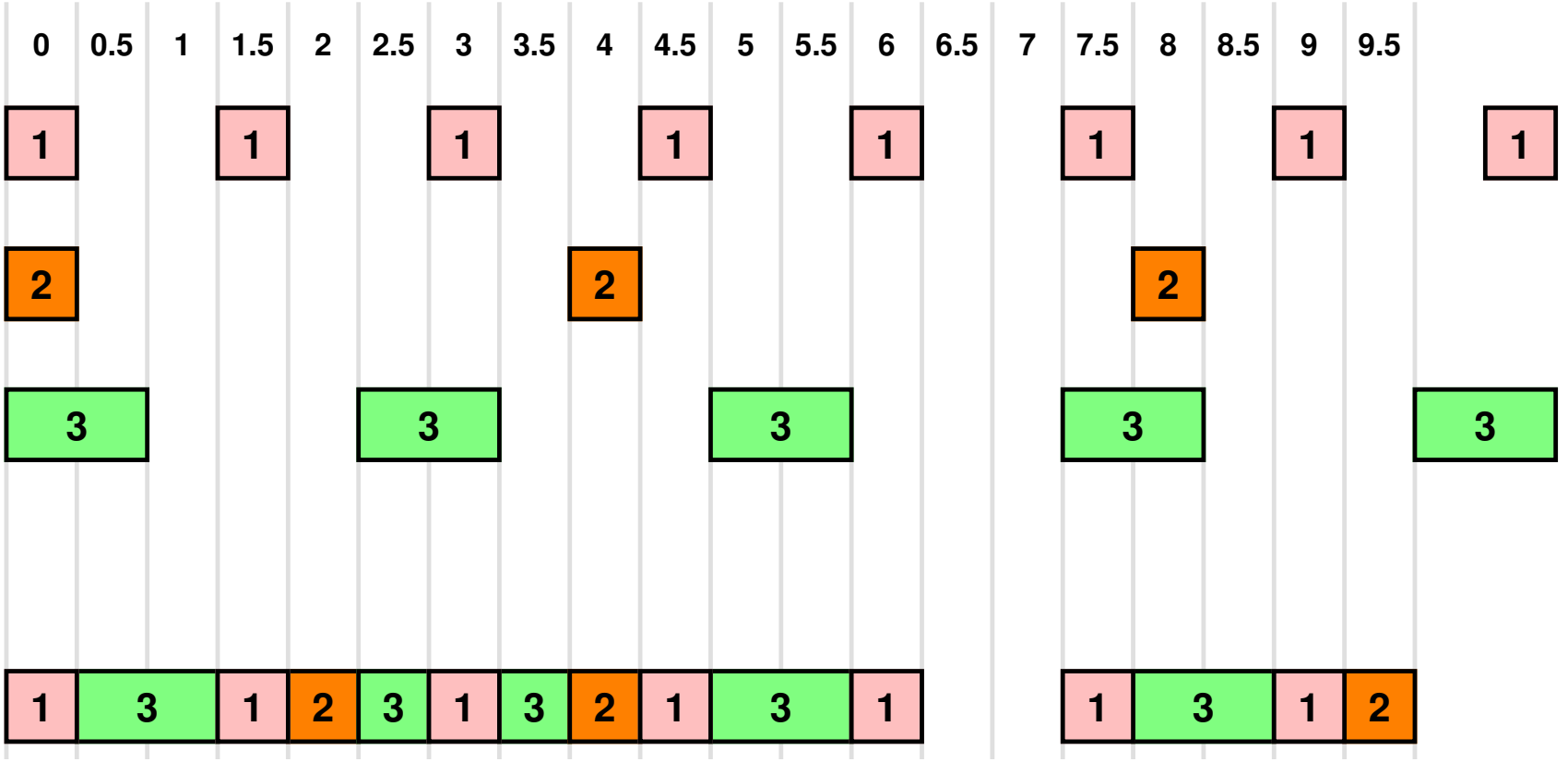




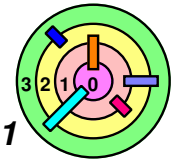
Scenario 1



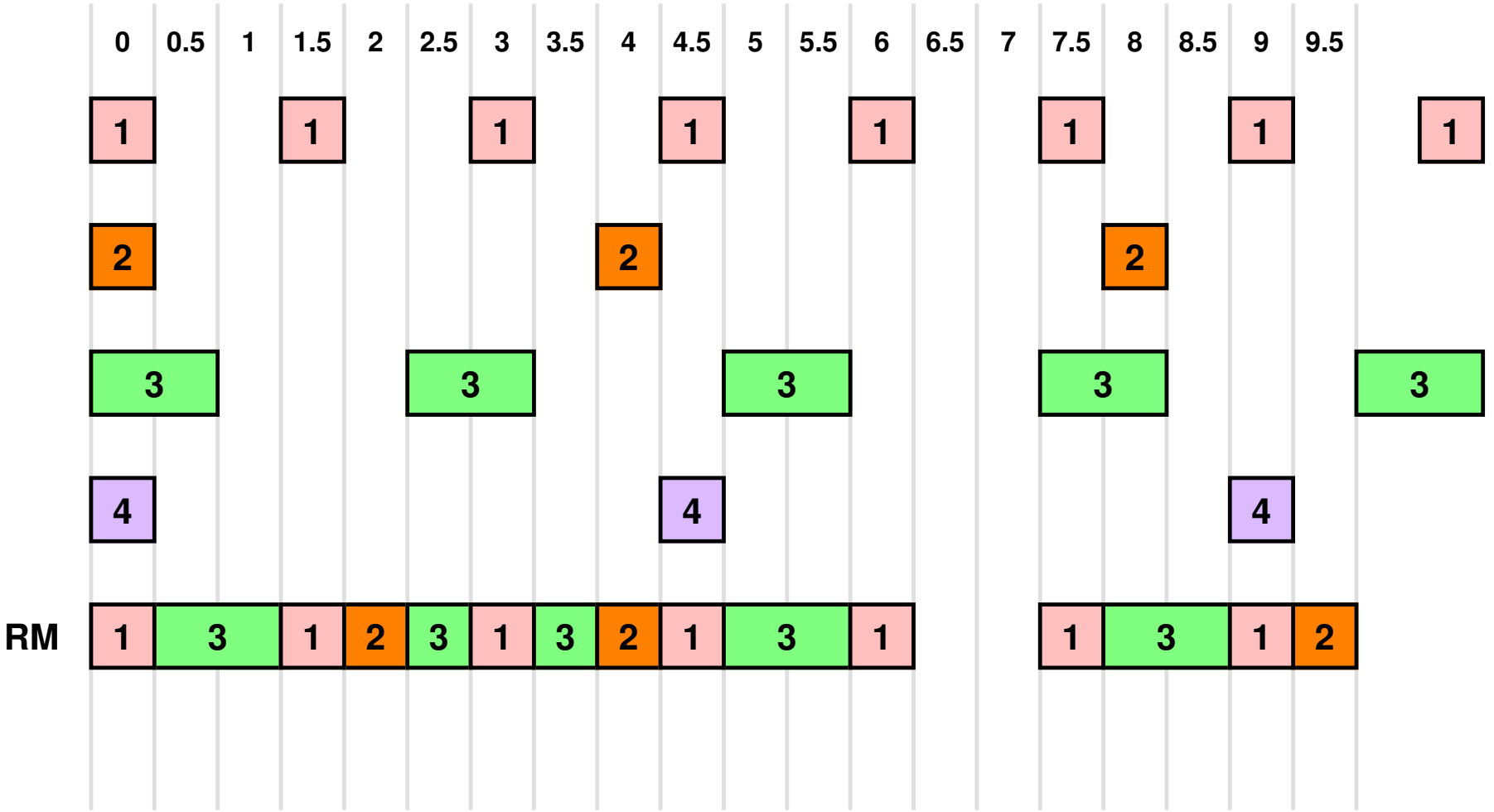
Scenario 1



- rate-monotonic scheduler succeeded
  - need to simulate till time = 60 to be certain

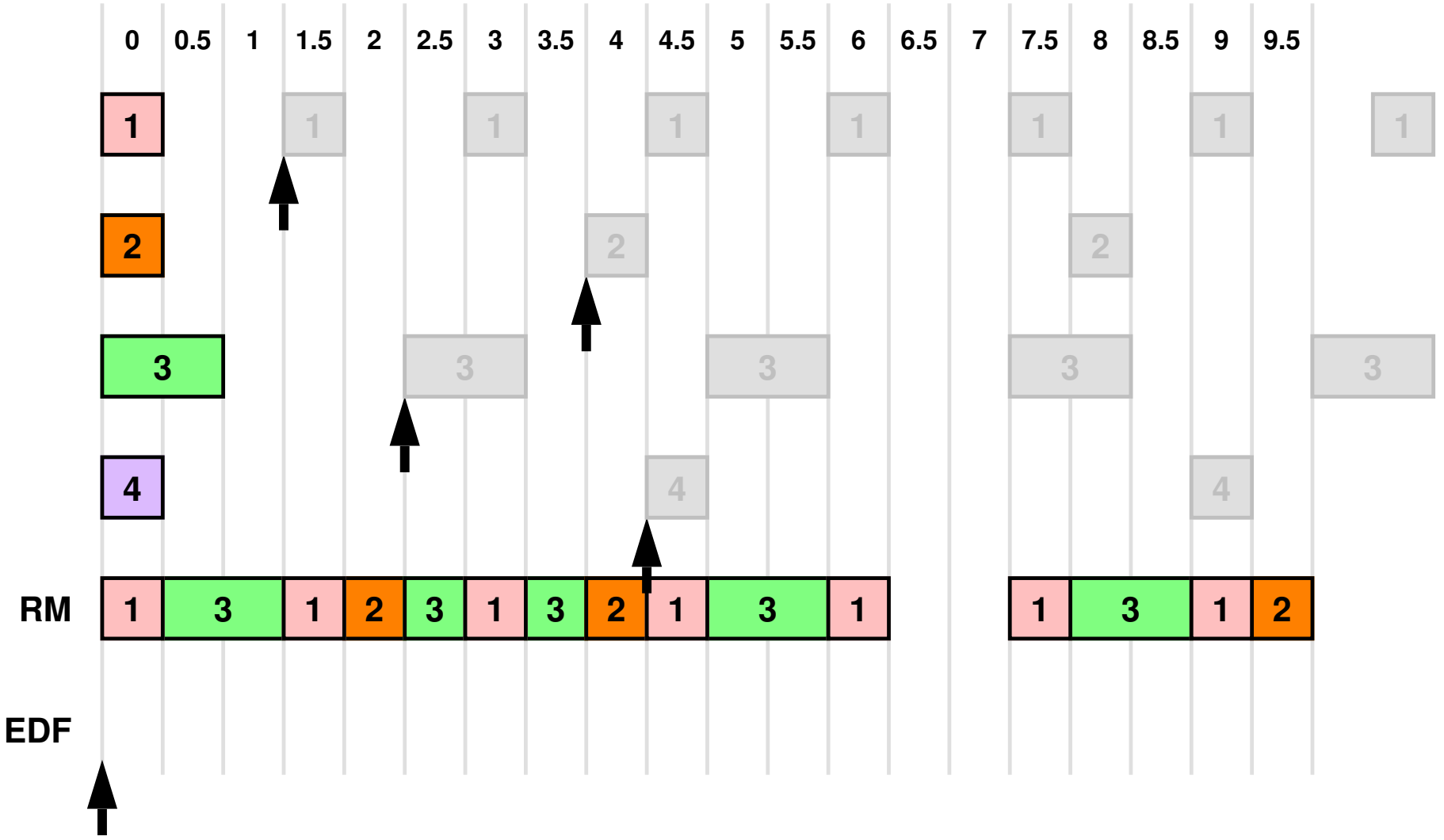


Scenario 2

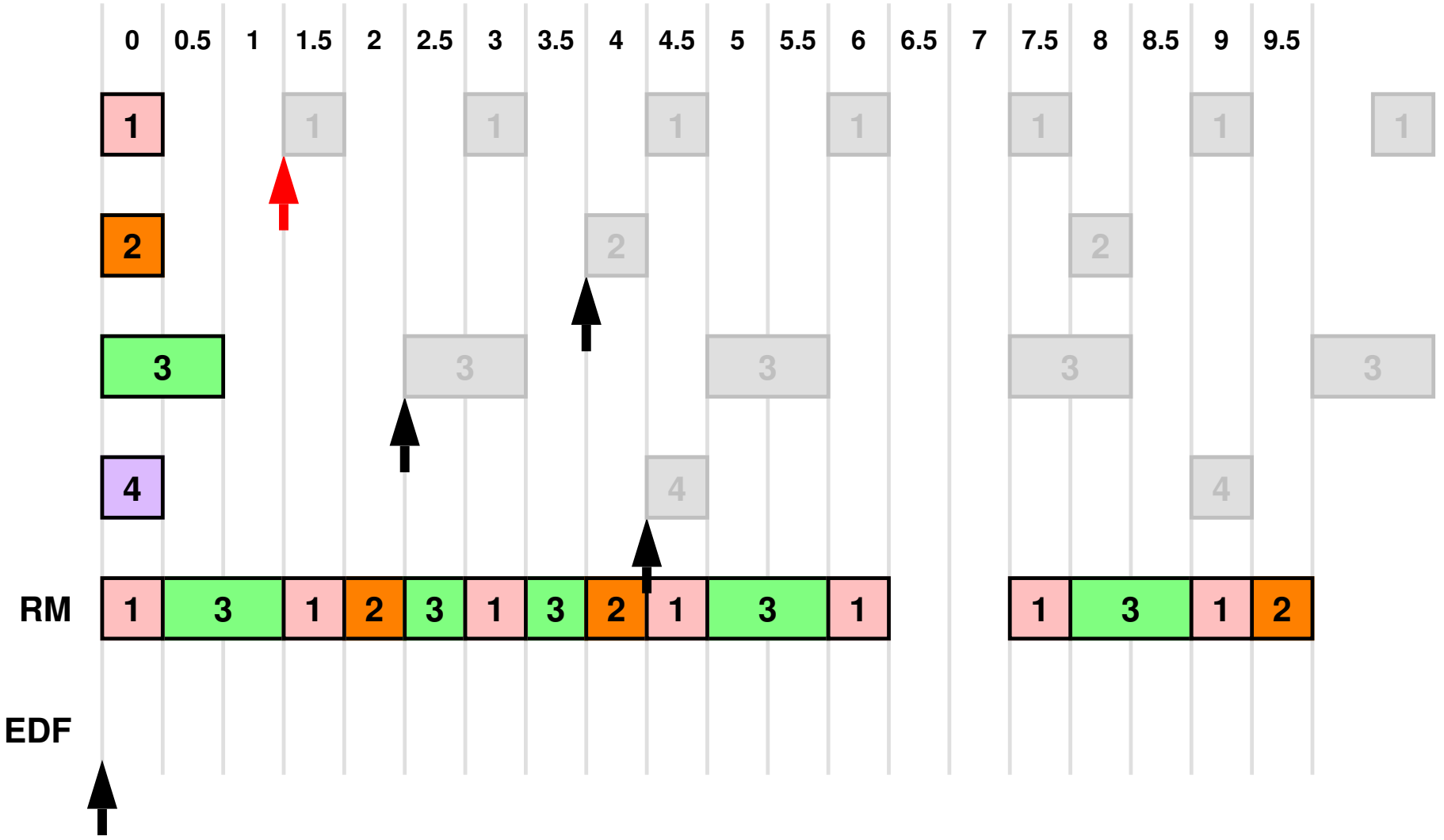


- rate-monotonic scheduler would fail in this case
- try EDF

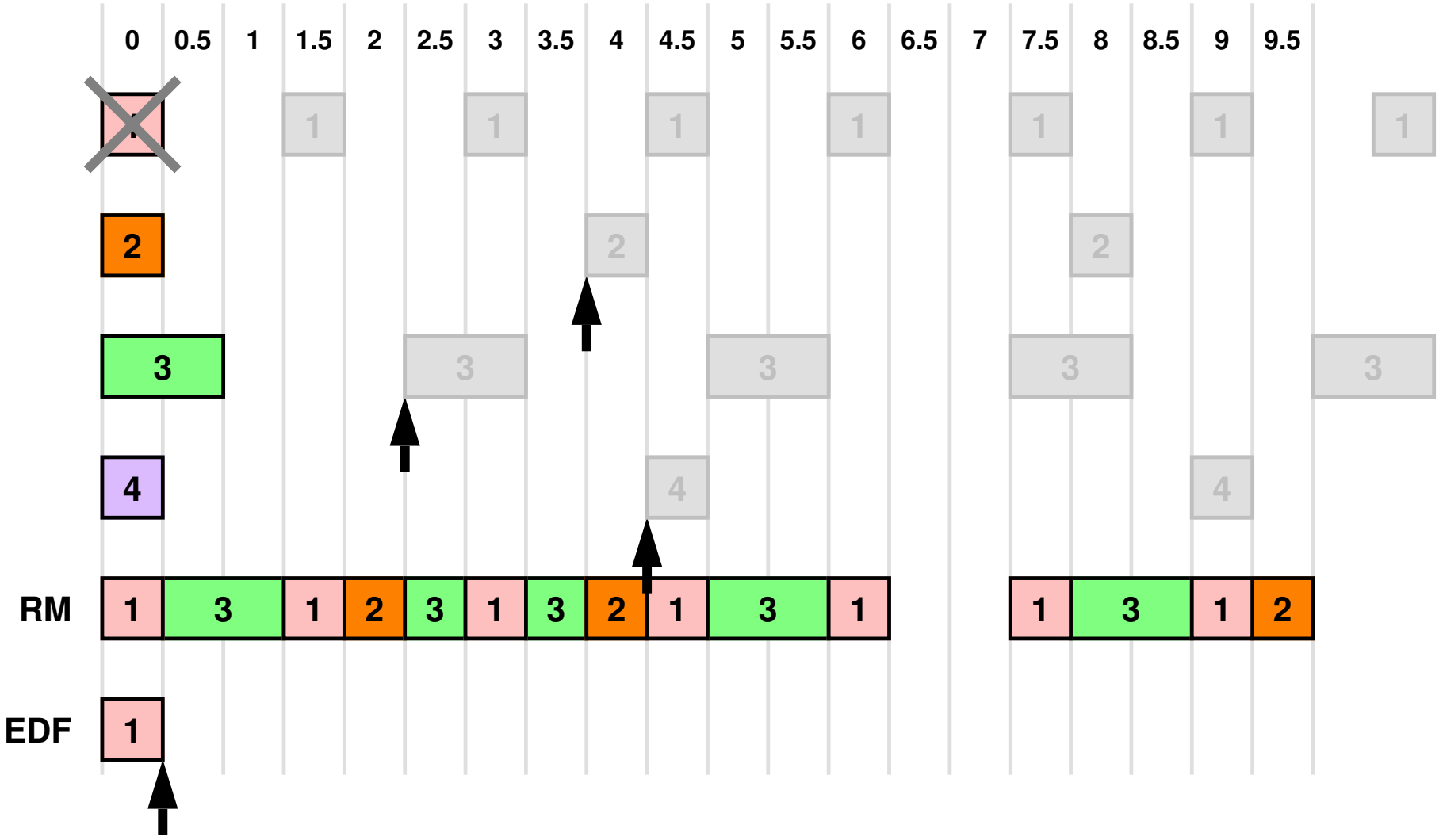
Earliest Deadline First



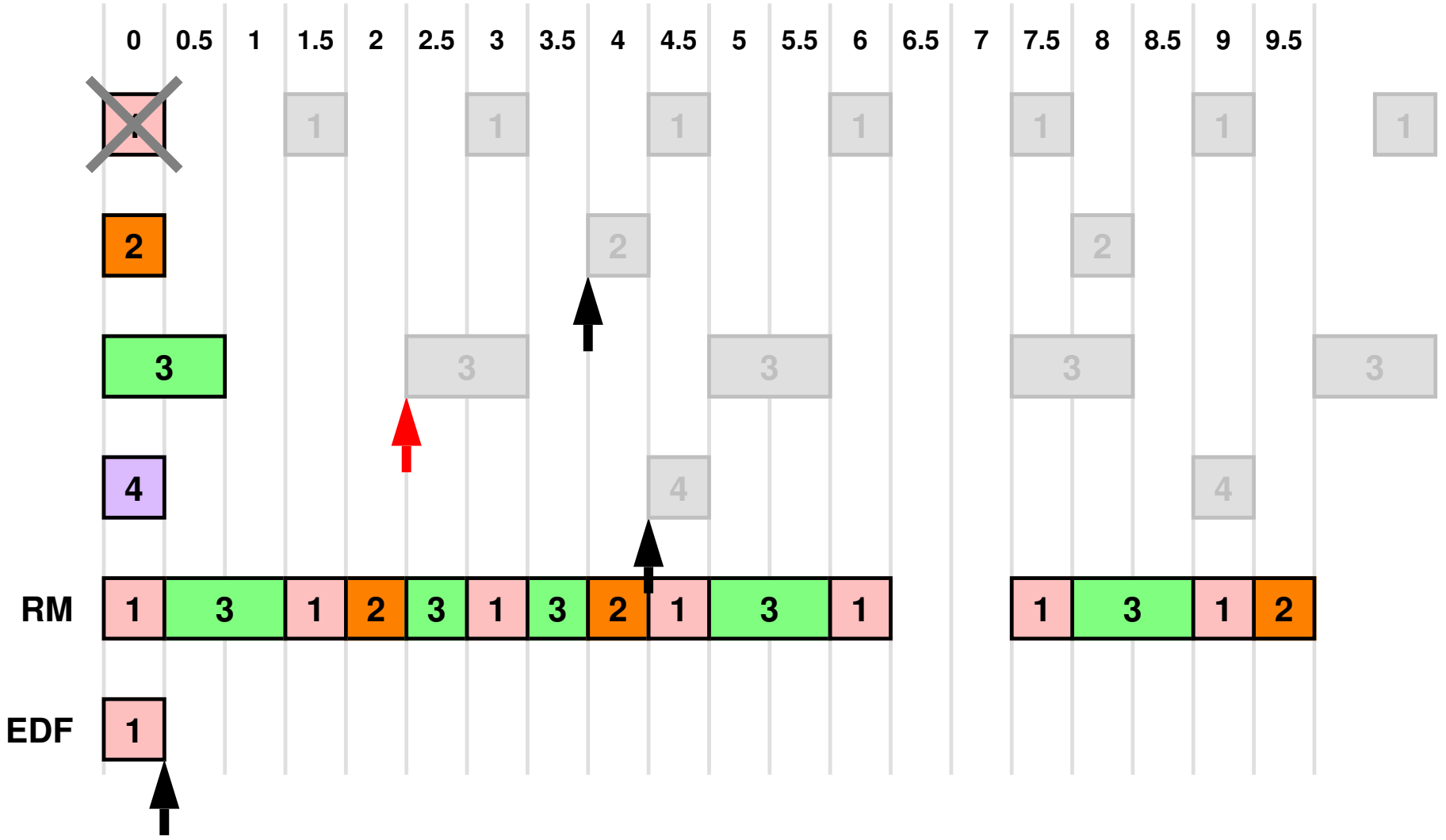
Earliest Deadline First



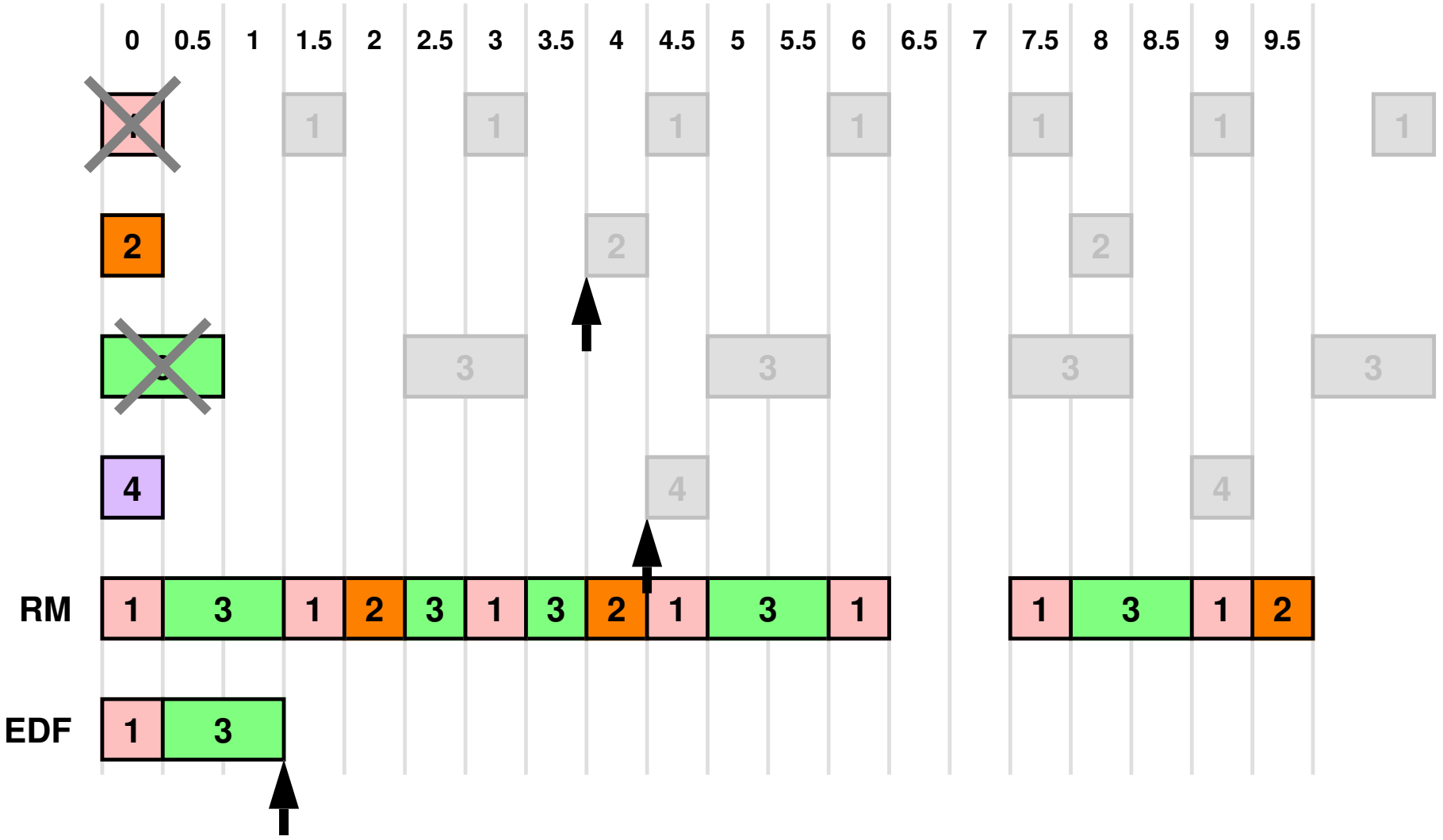
Earliest Deadline First



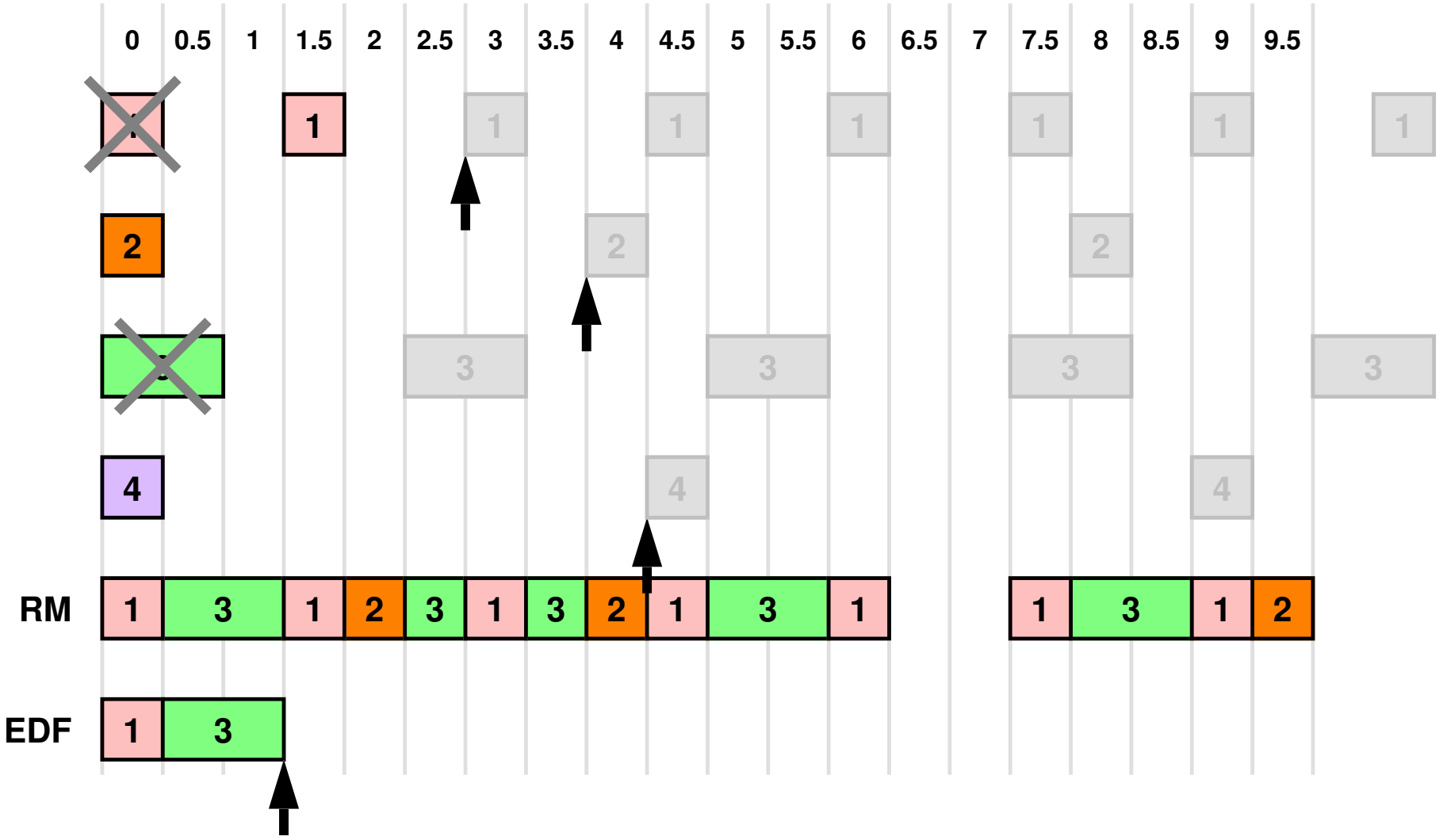
Earliest Deadline First



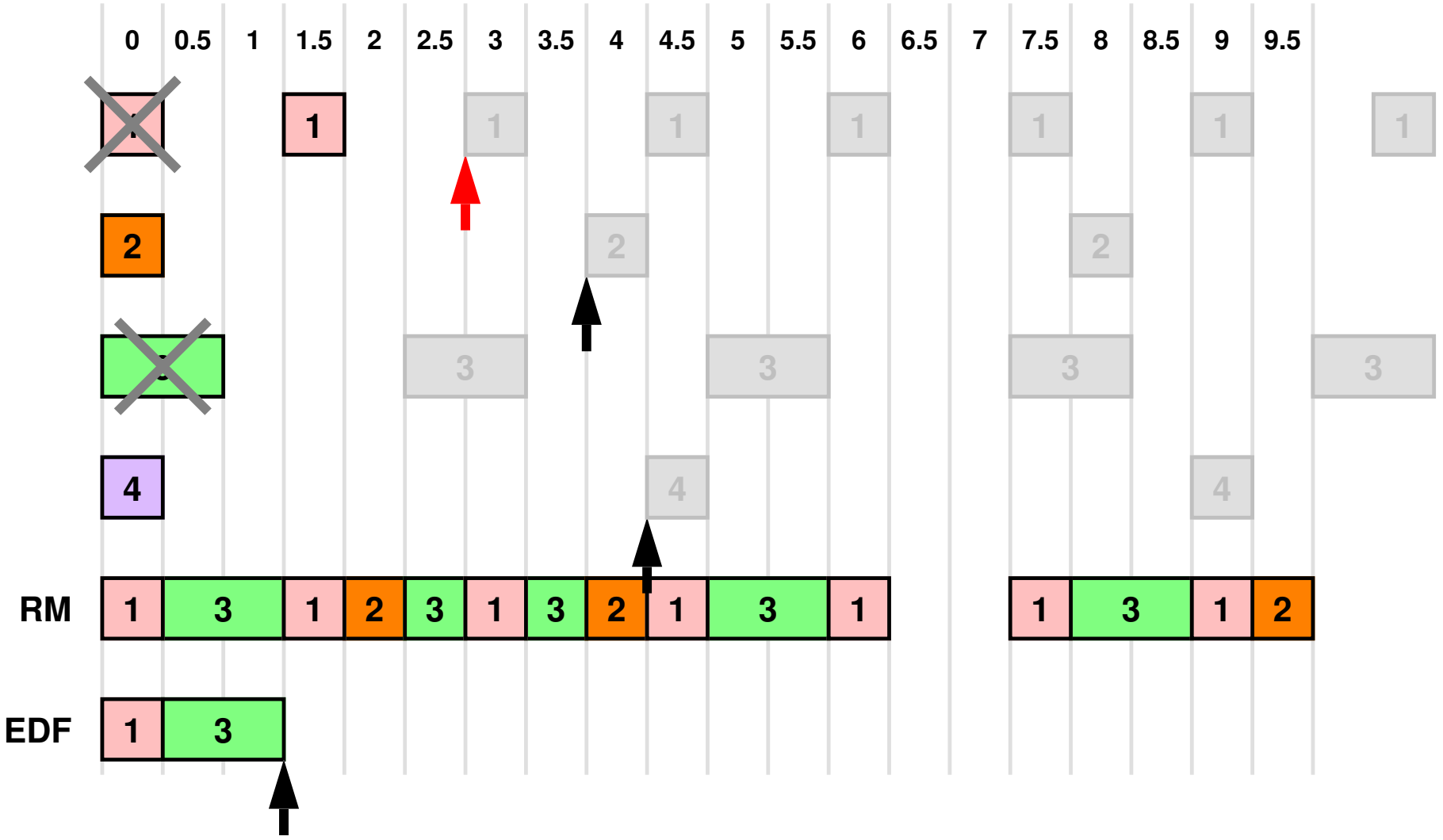
Earliest Deadline First



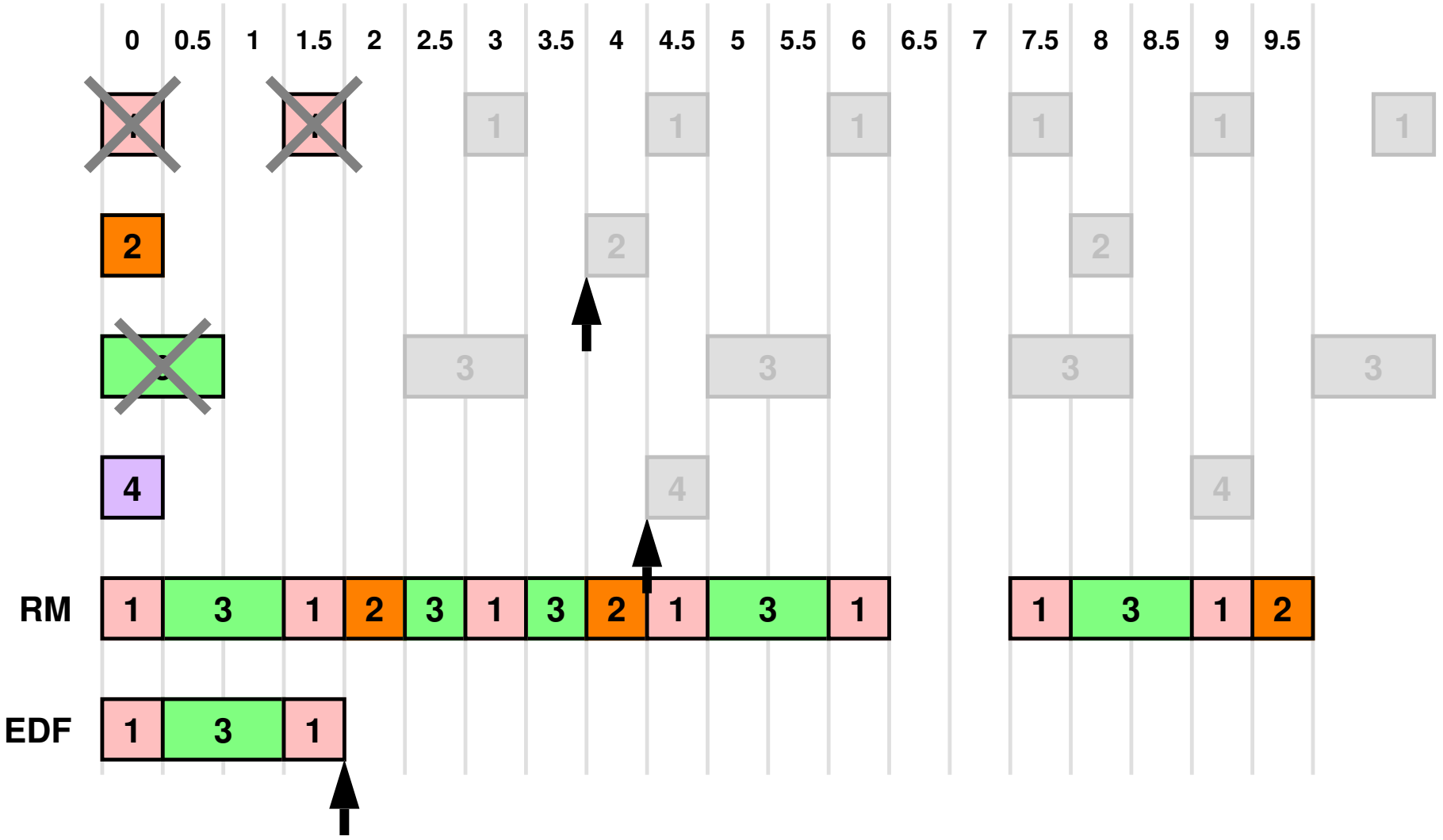
Earliest Deadline First



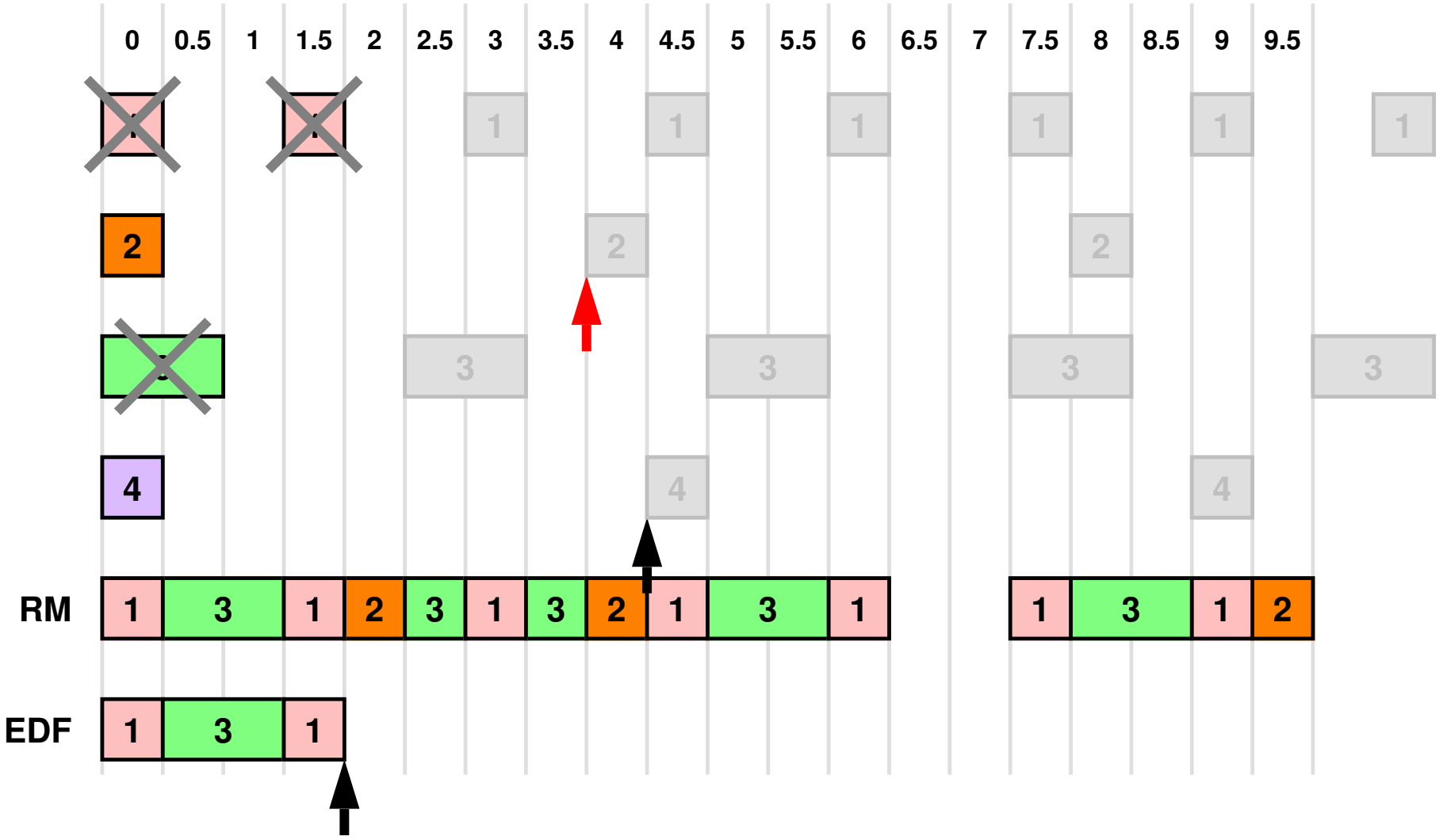
Earliest Deadline First



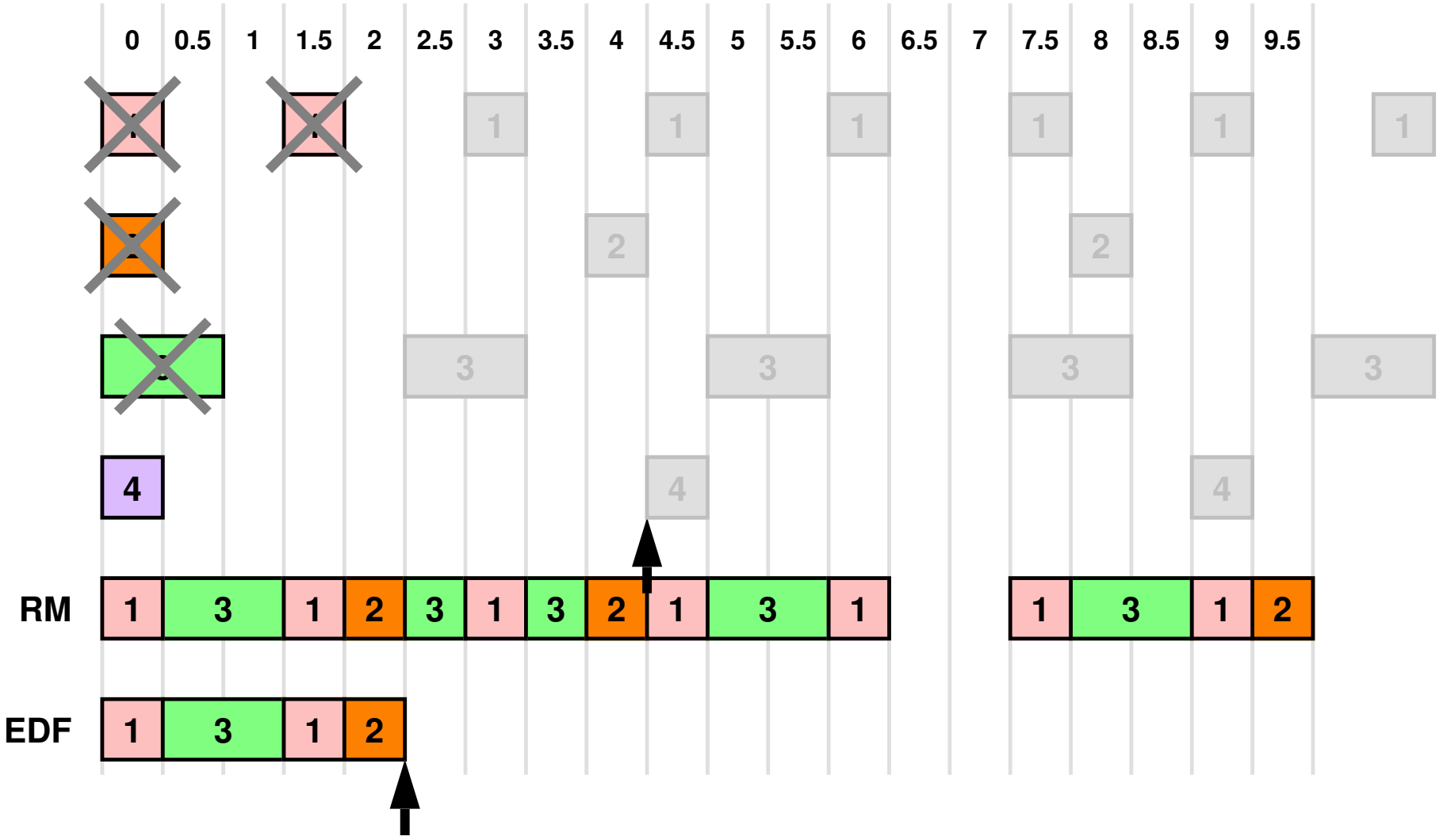
Earliest Deadline First



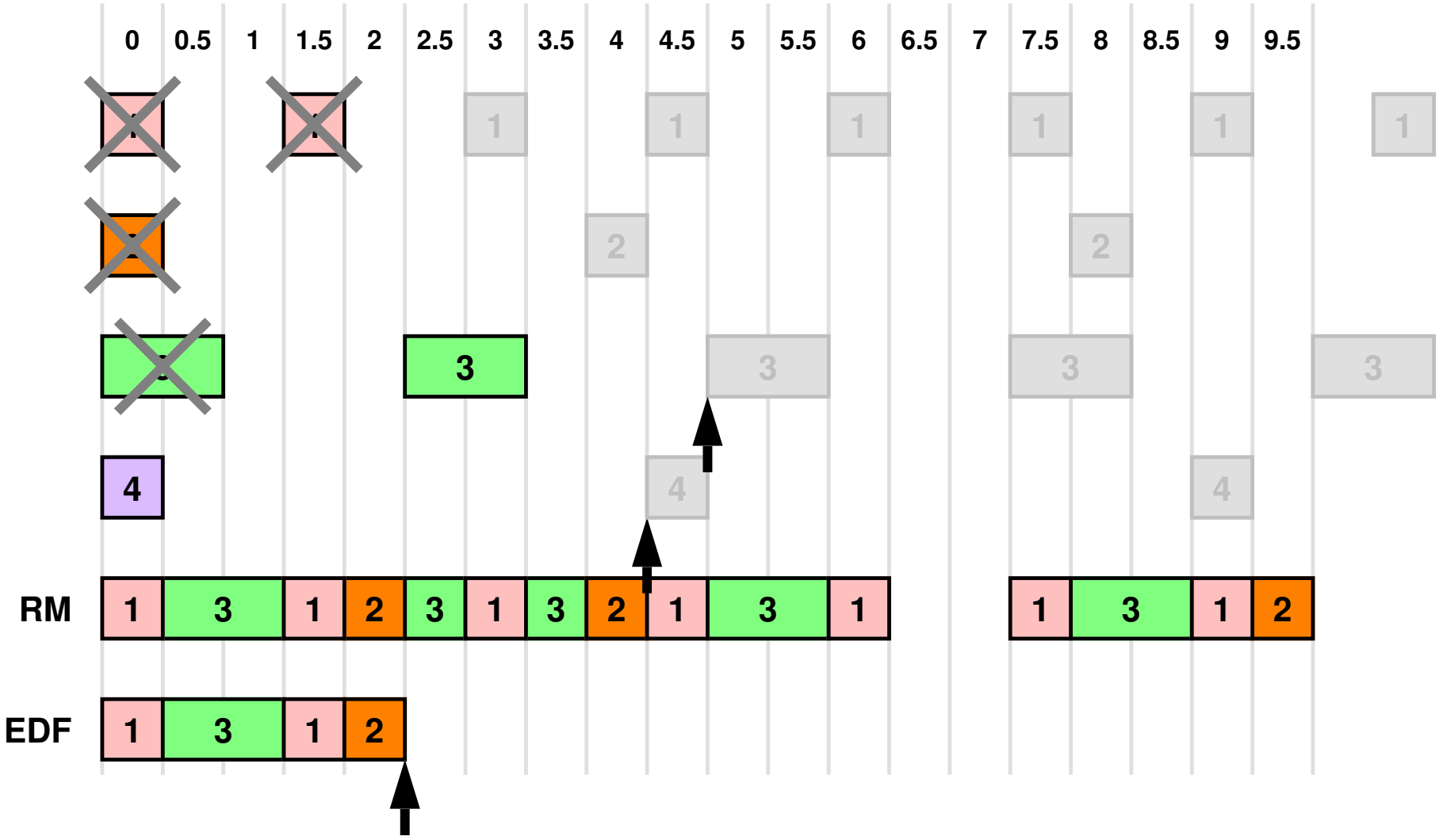
Earliest Deadline First



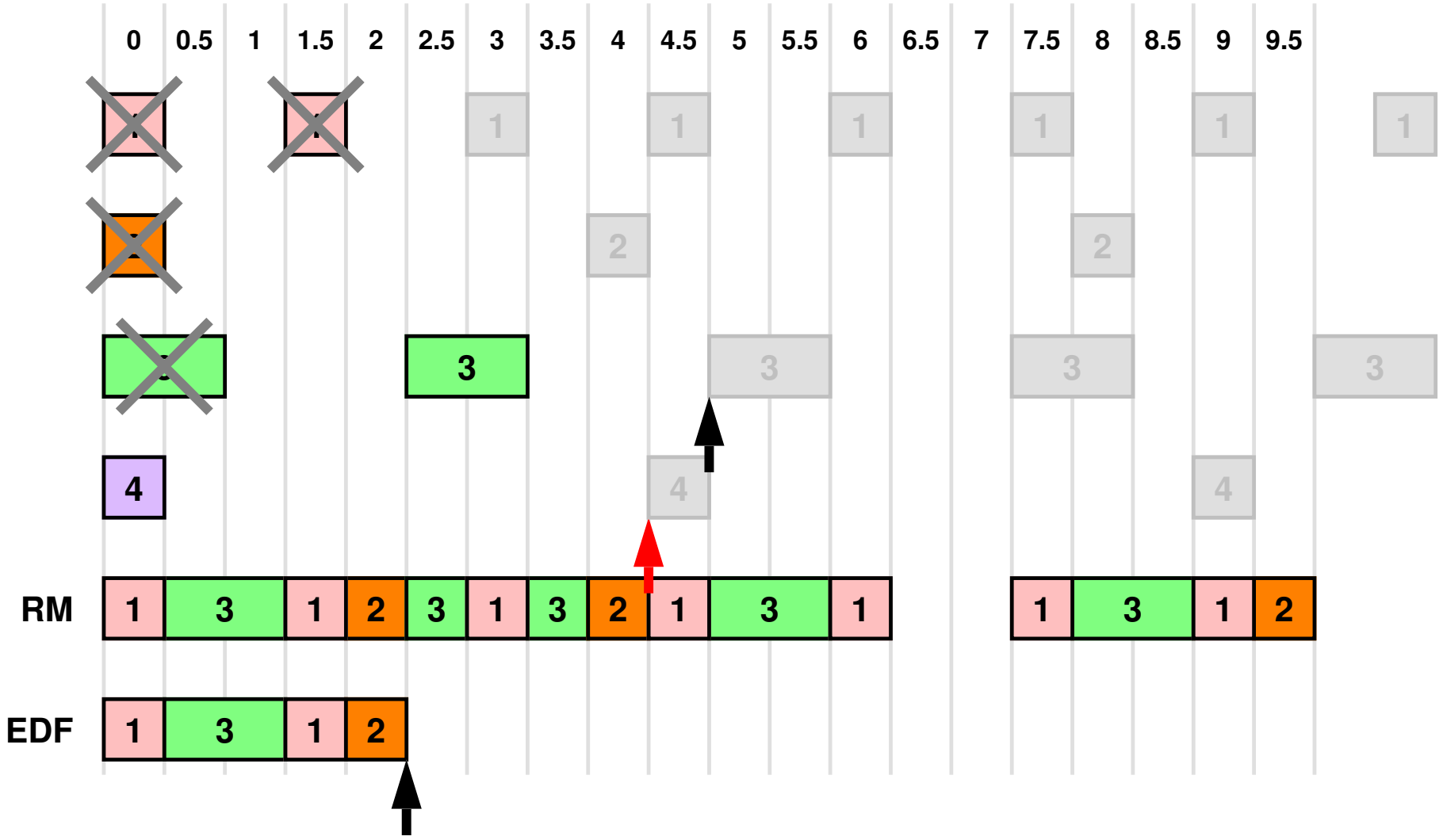
Earliest Deadline First



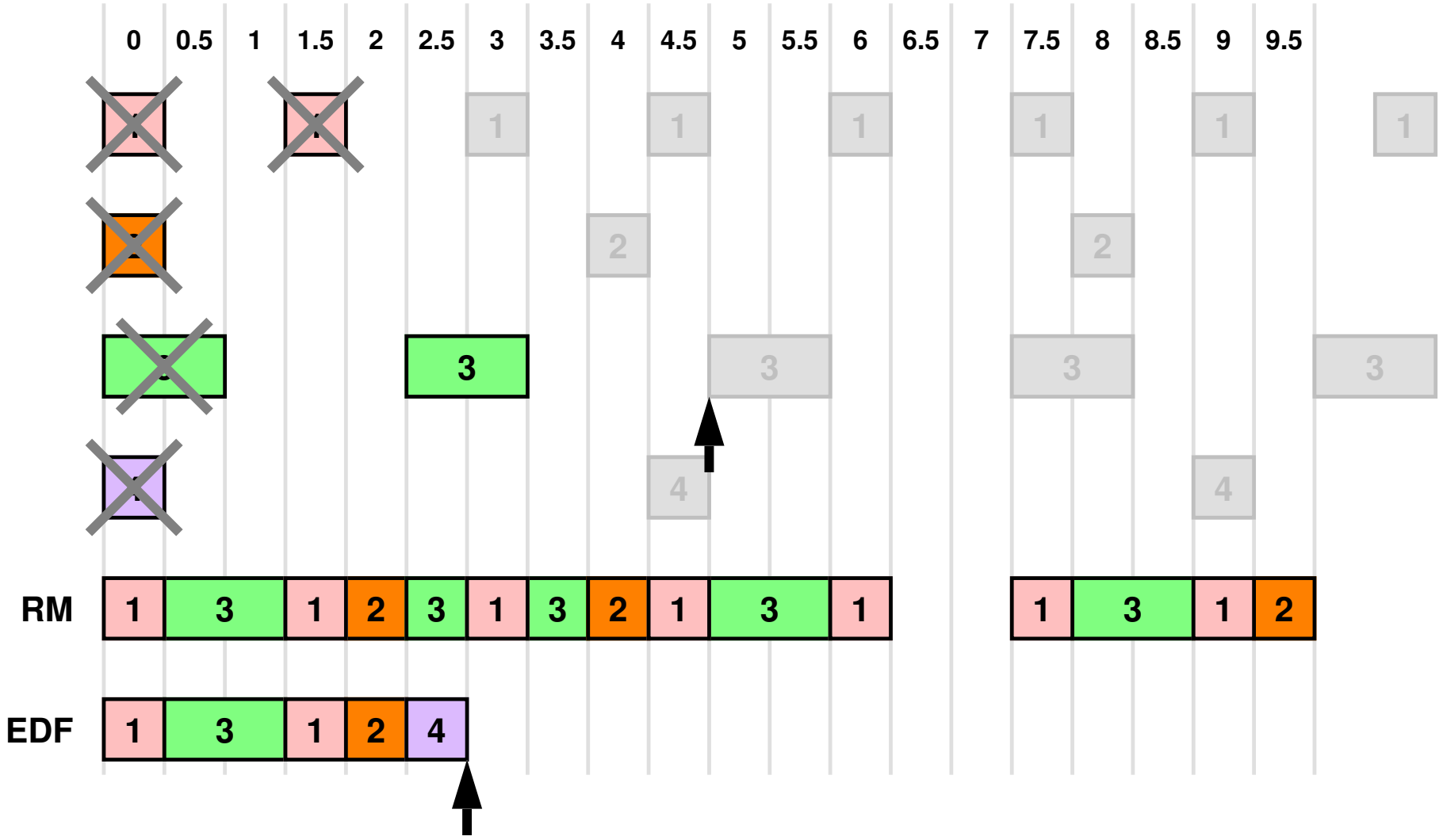
Earliest Deadline First



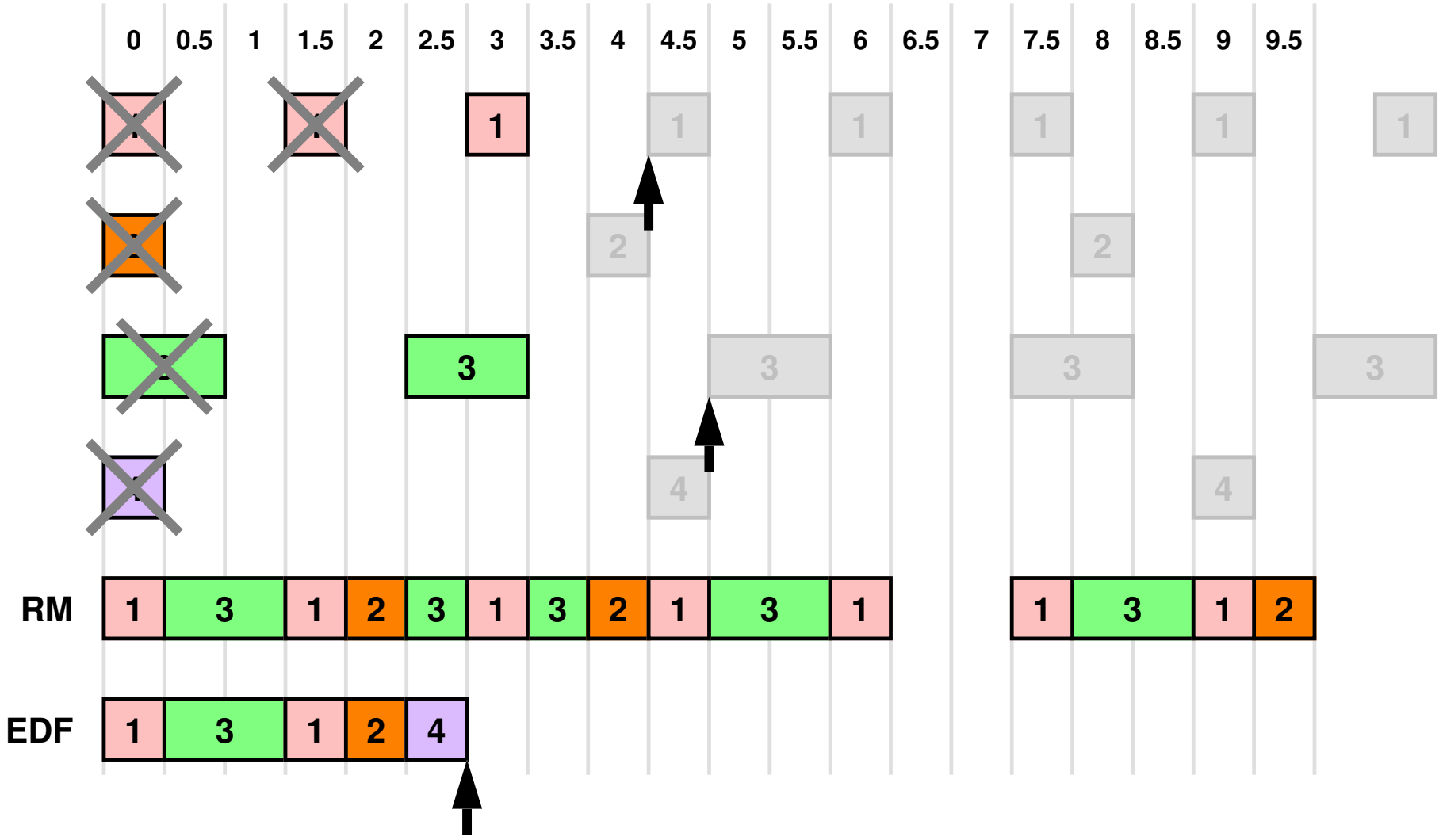
Earliest Deadline First



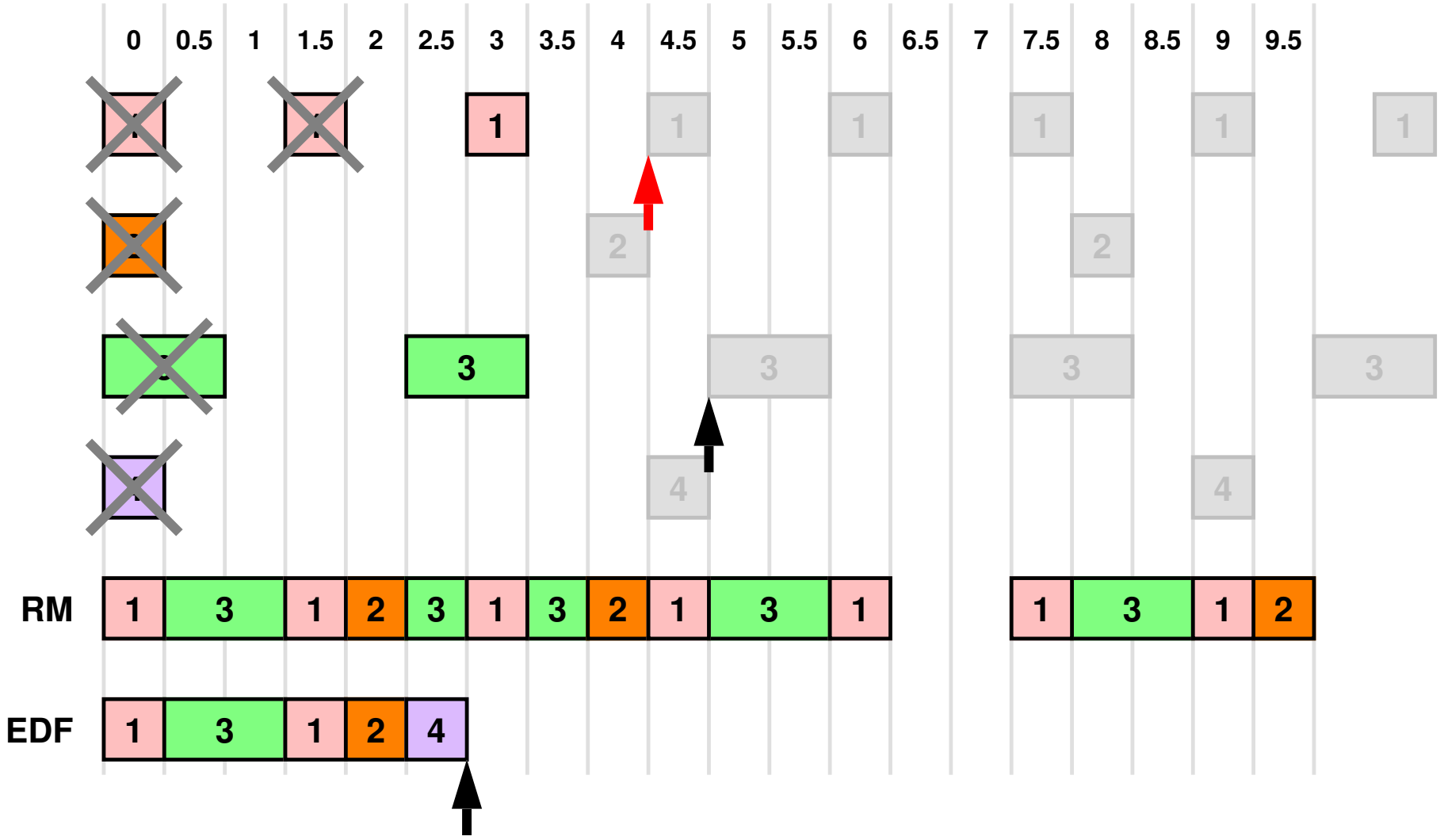
Earliest Deadline First



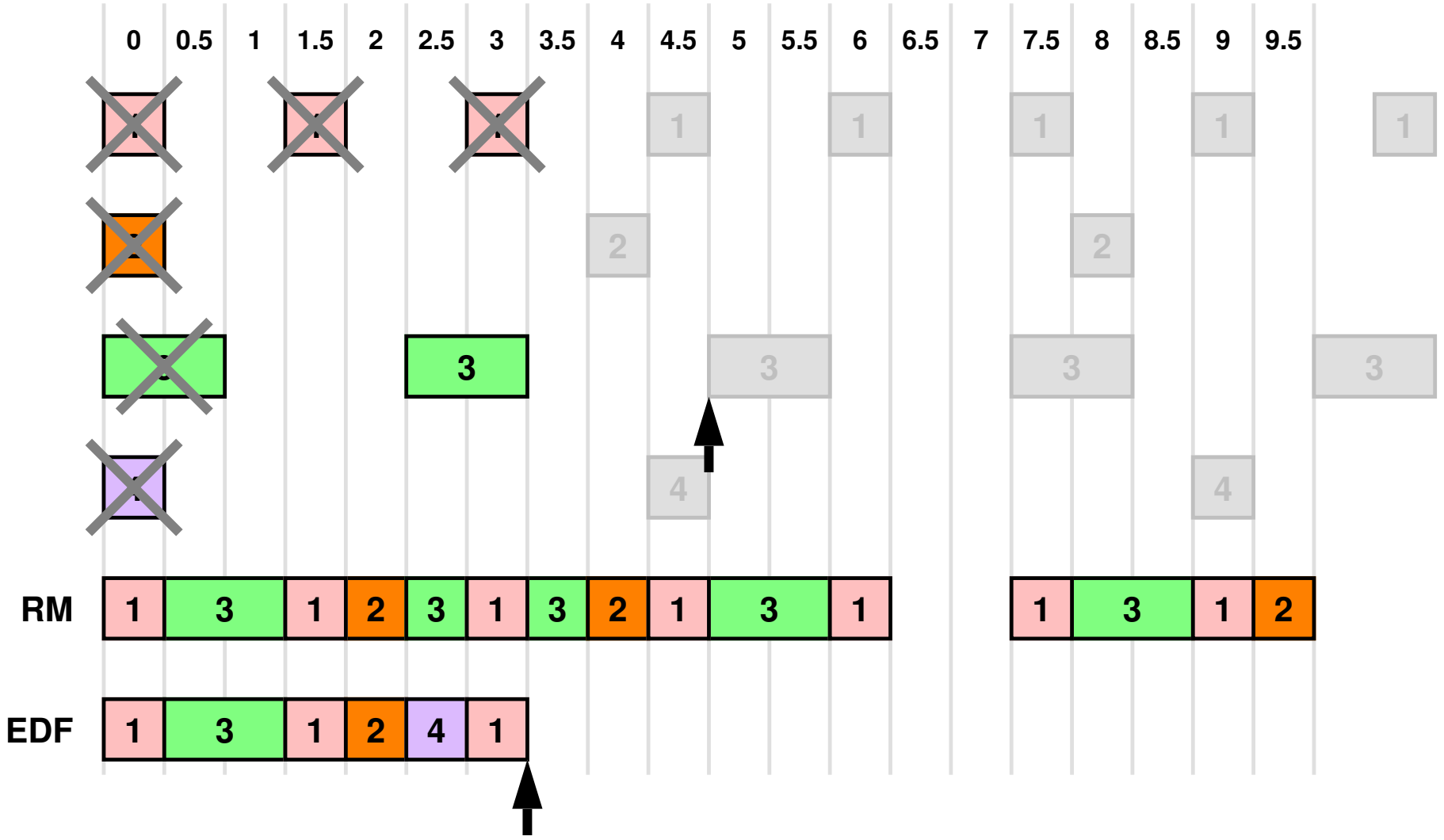
Earliest Deadline First



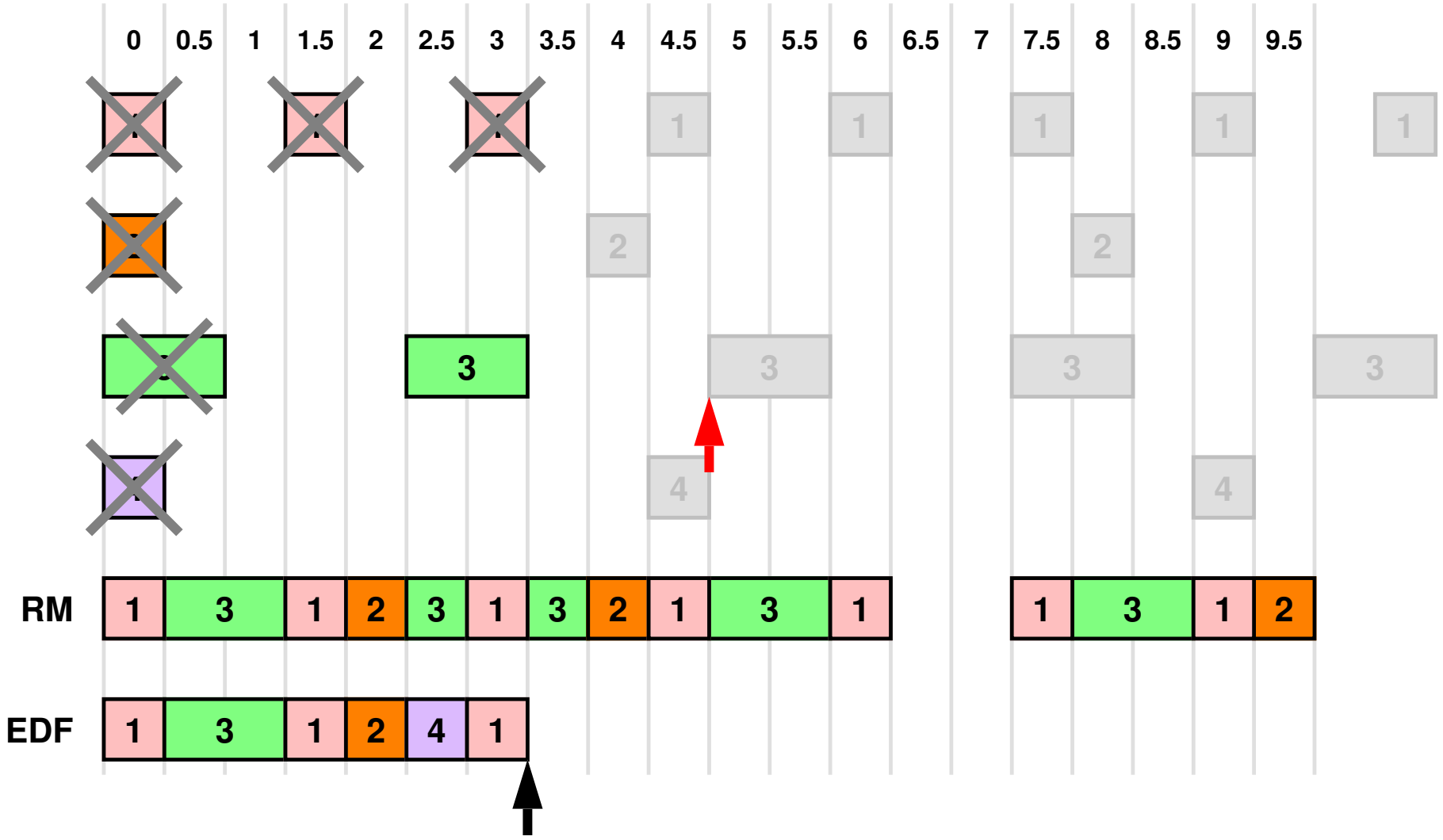
Earliest Deadline First



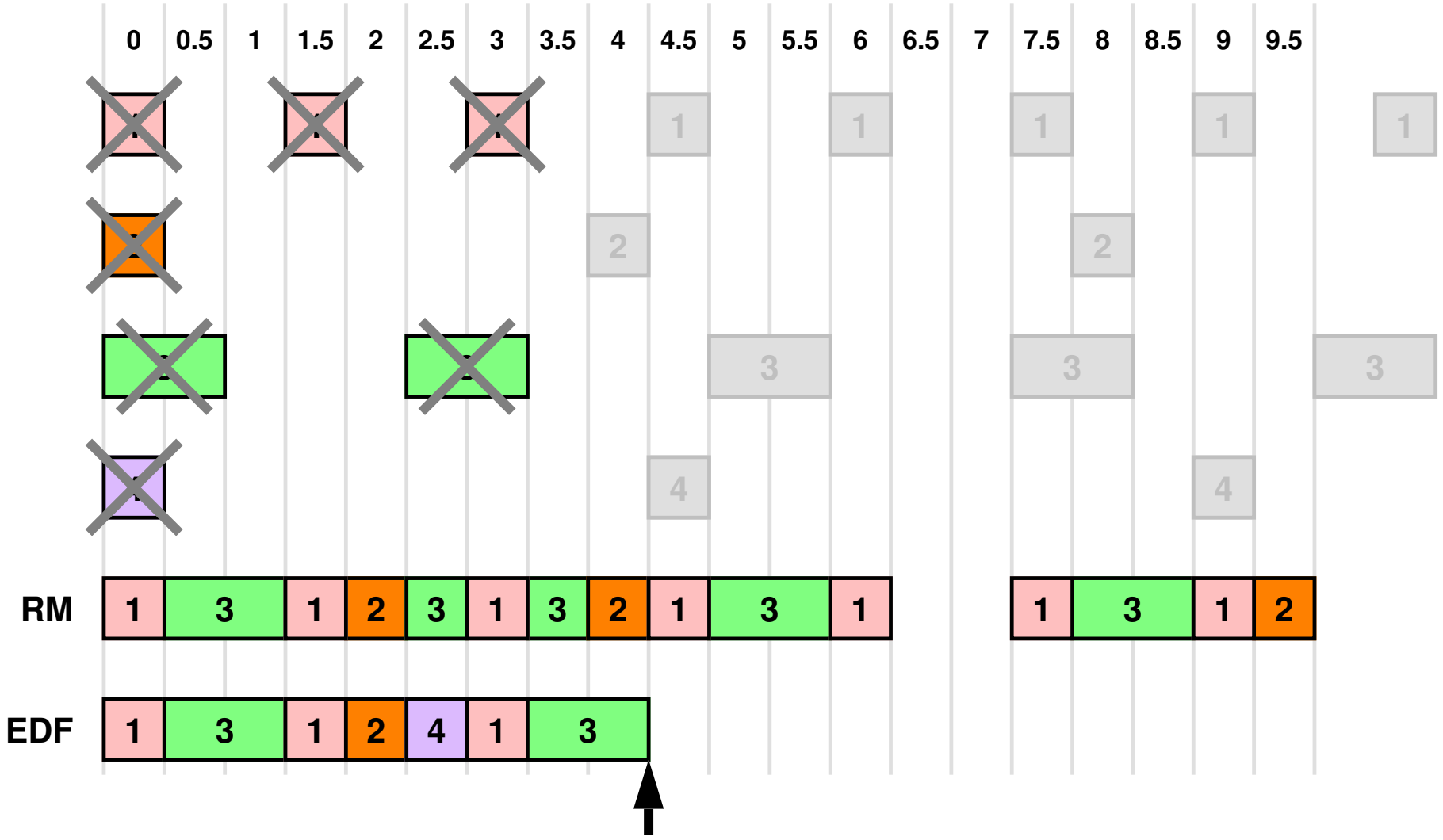
Earliest Deadline First



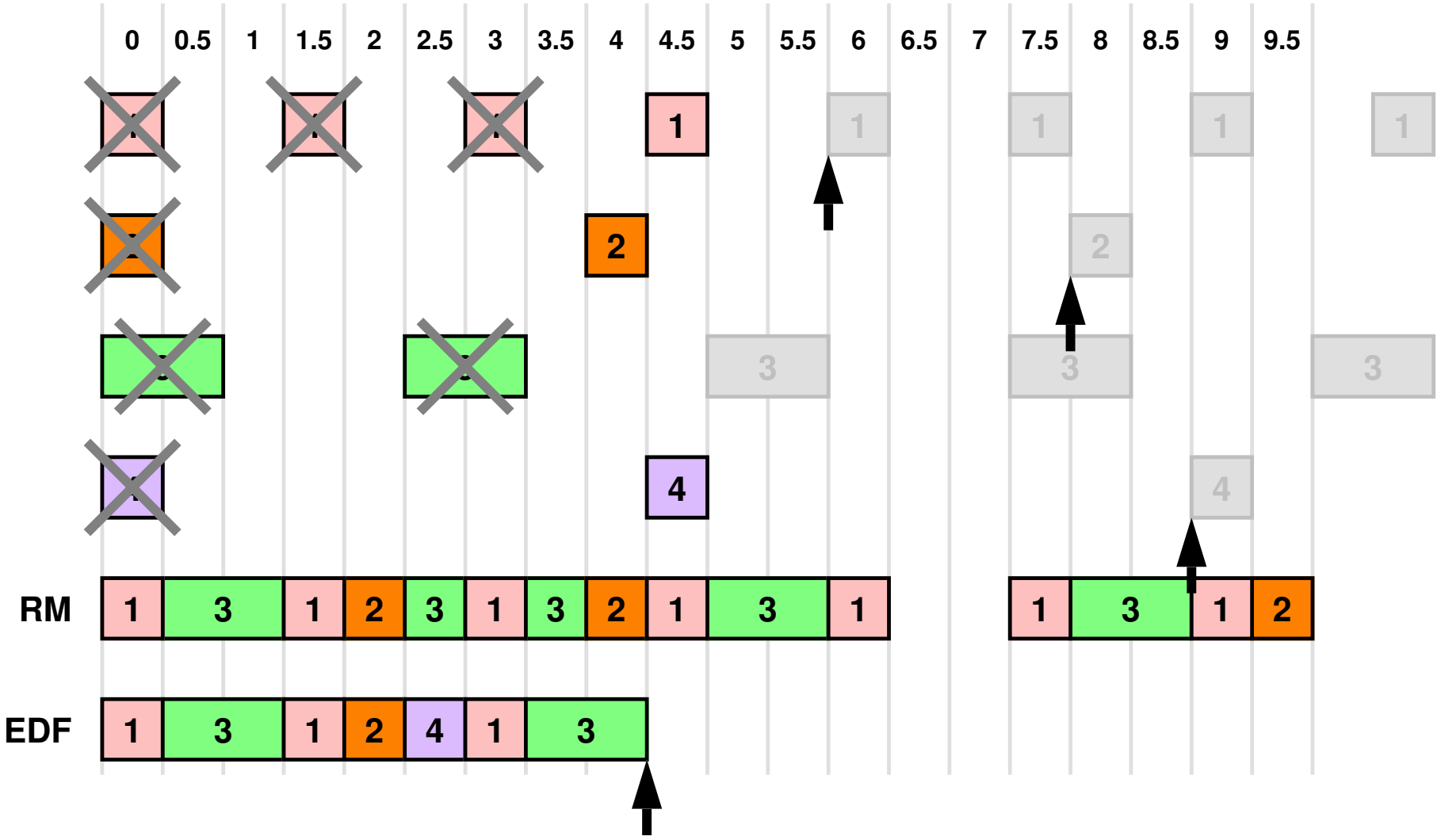
Earliest Deadline First



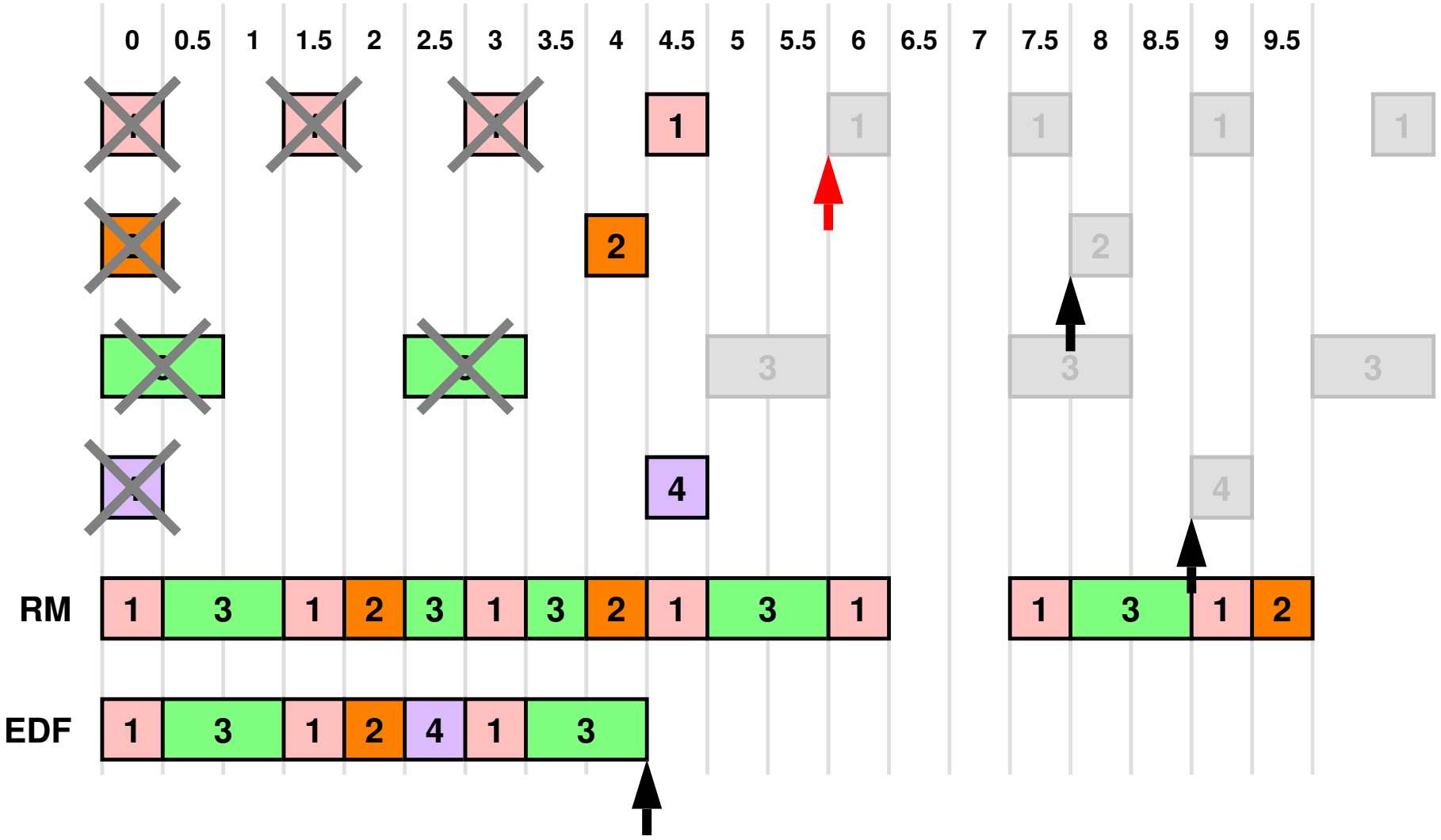
Earliest Deadline First



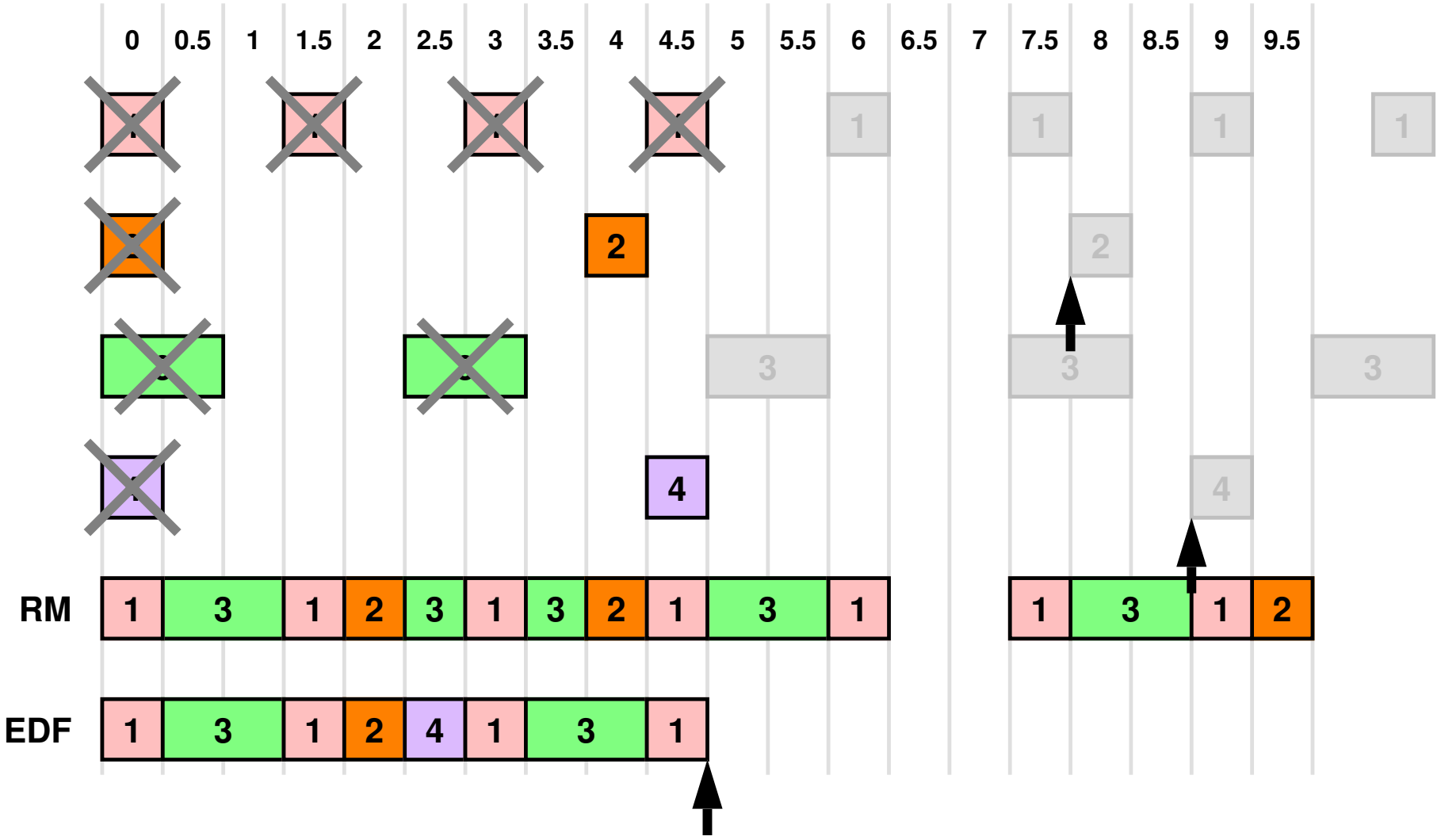
Earliest Deadline First



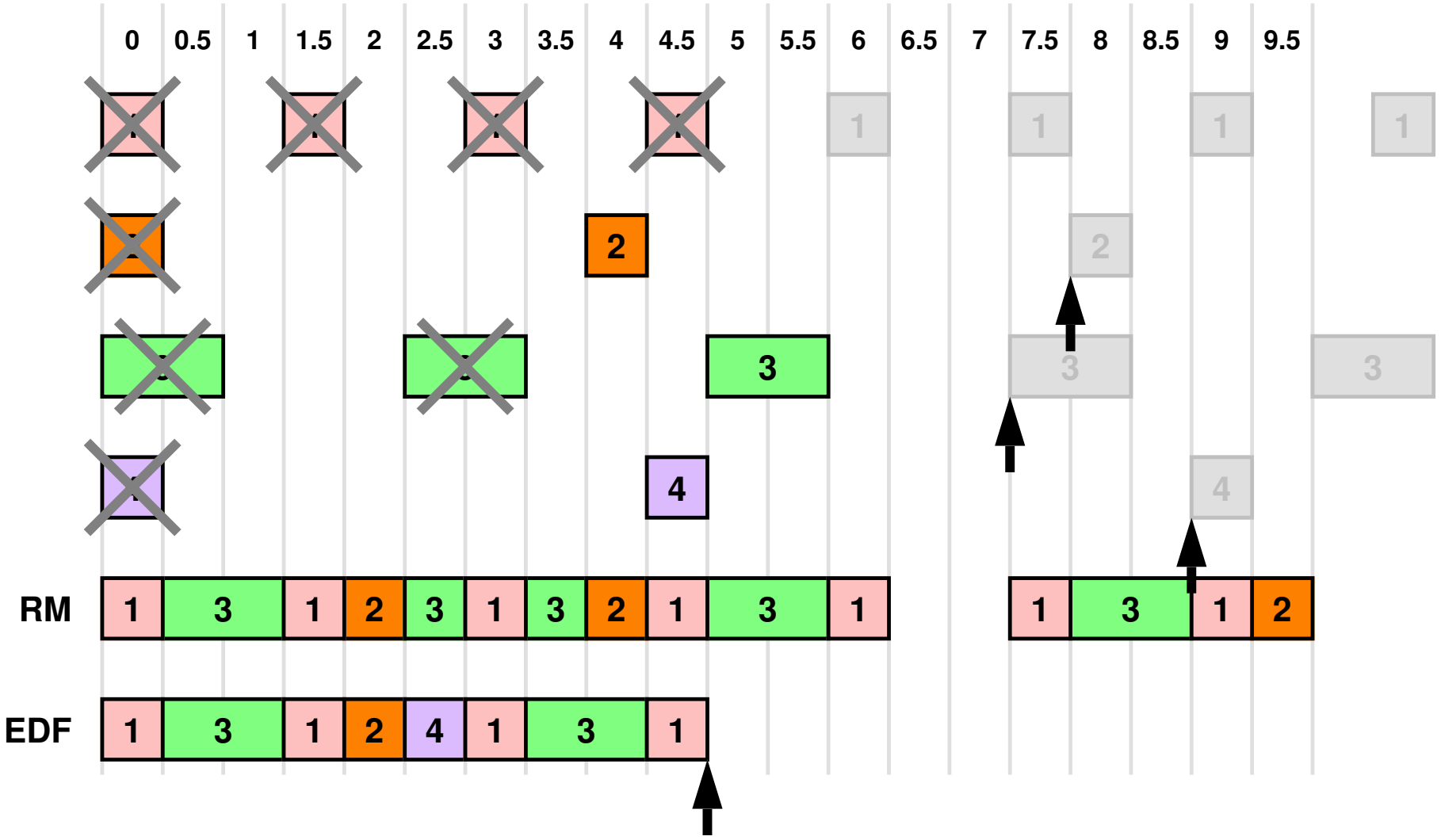
Earliest Deadline First



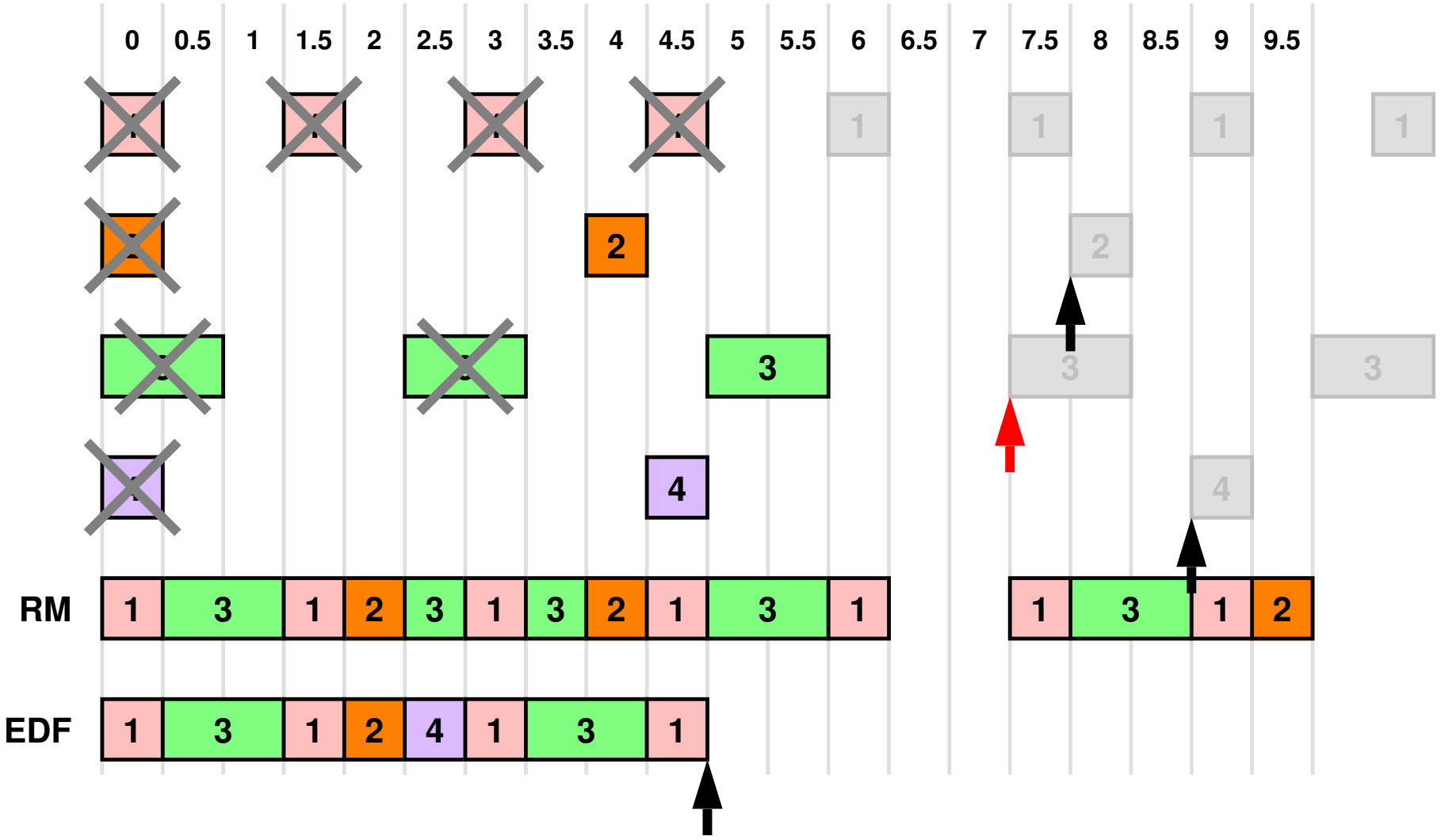
Earliest Deadline First



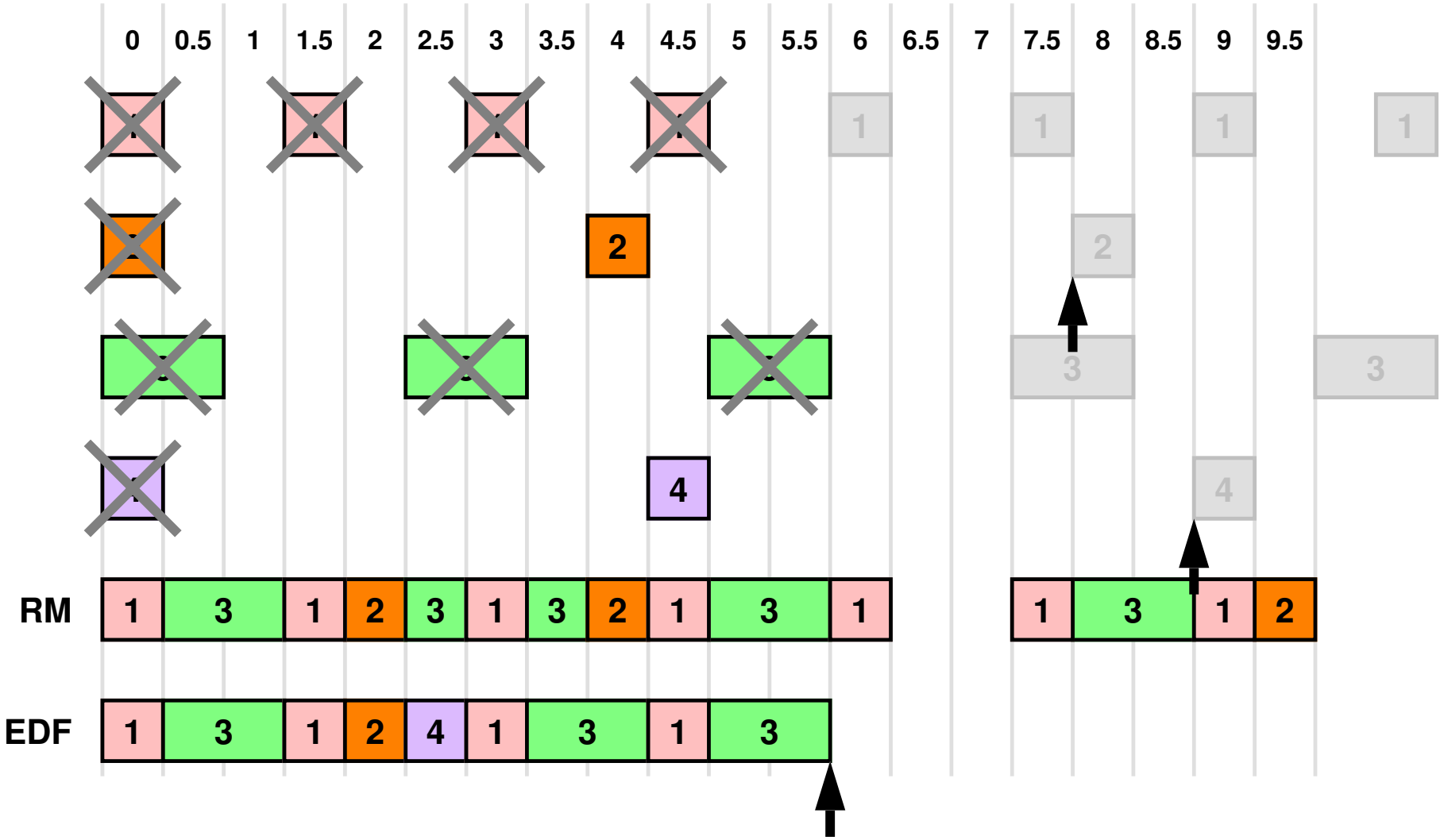
Earliest Deadline First



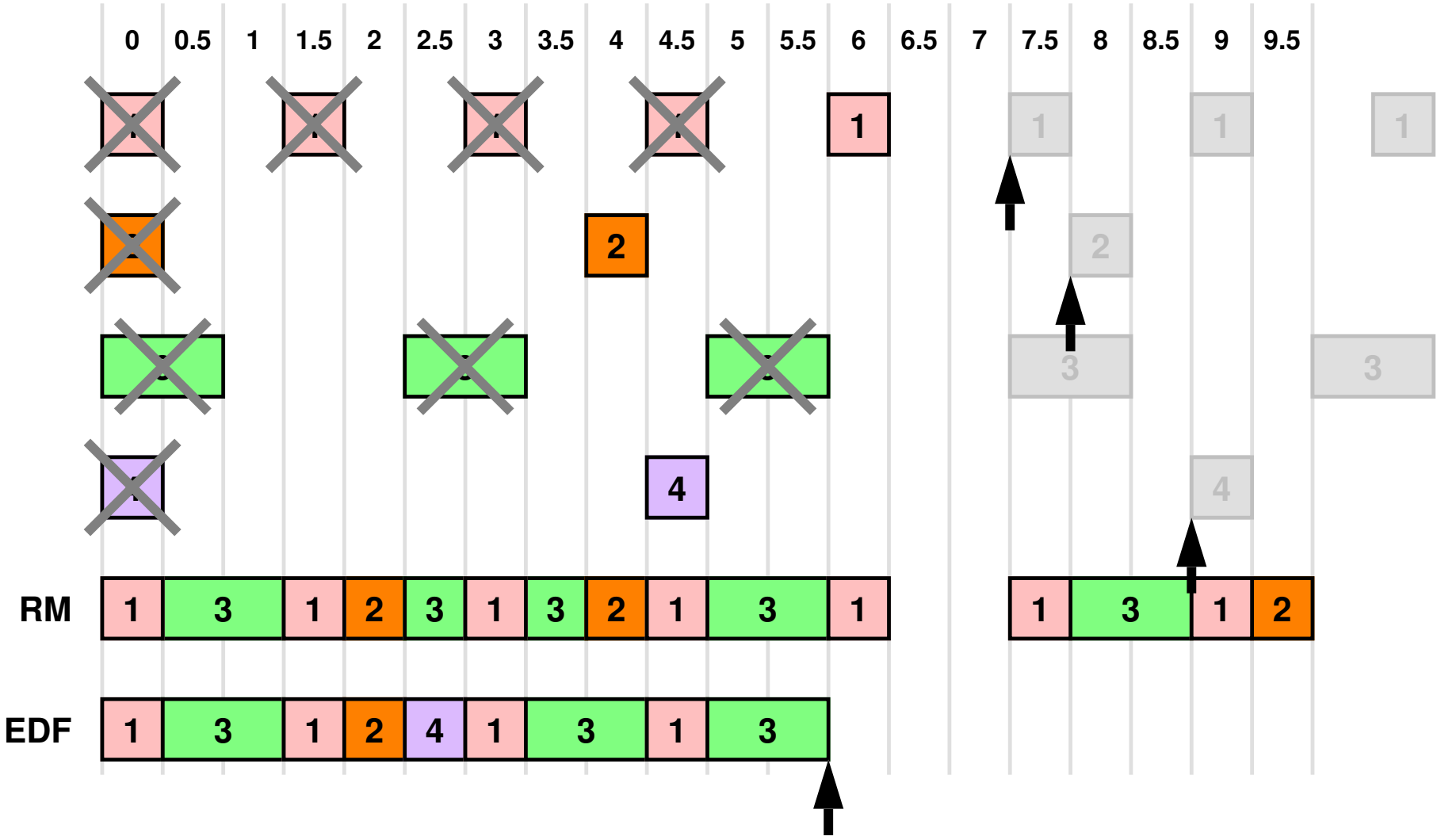
Earliest Deadline First



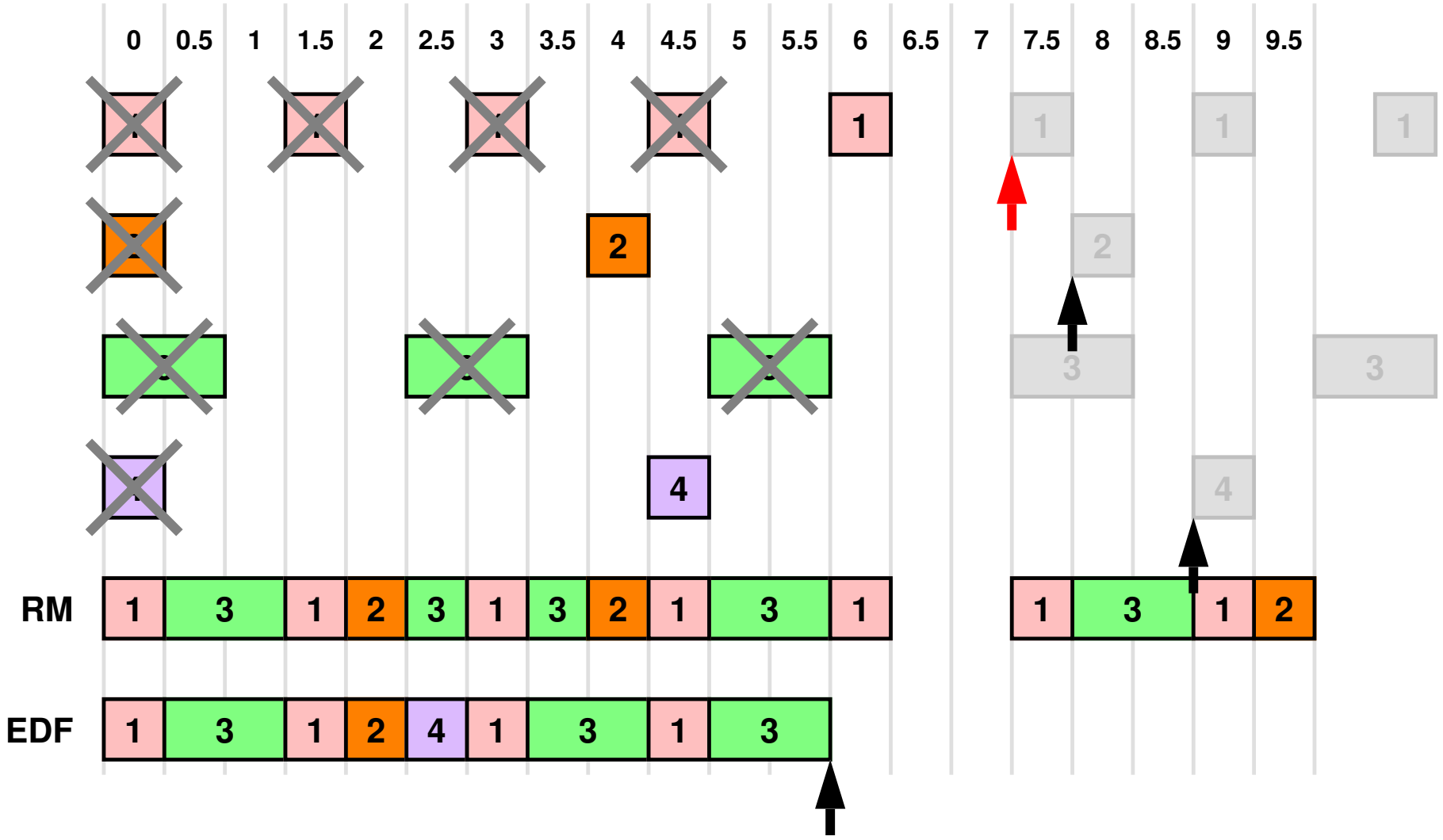
Earliest Deadline First



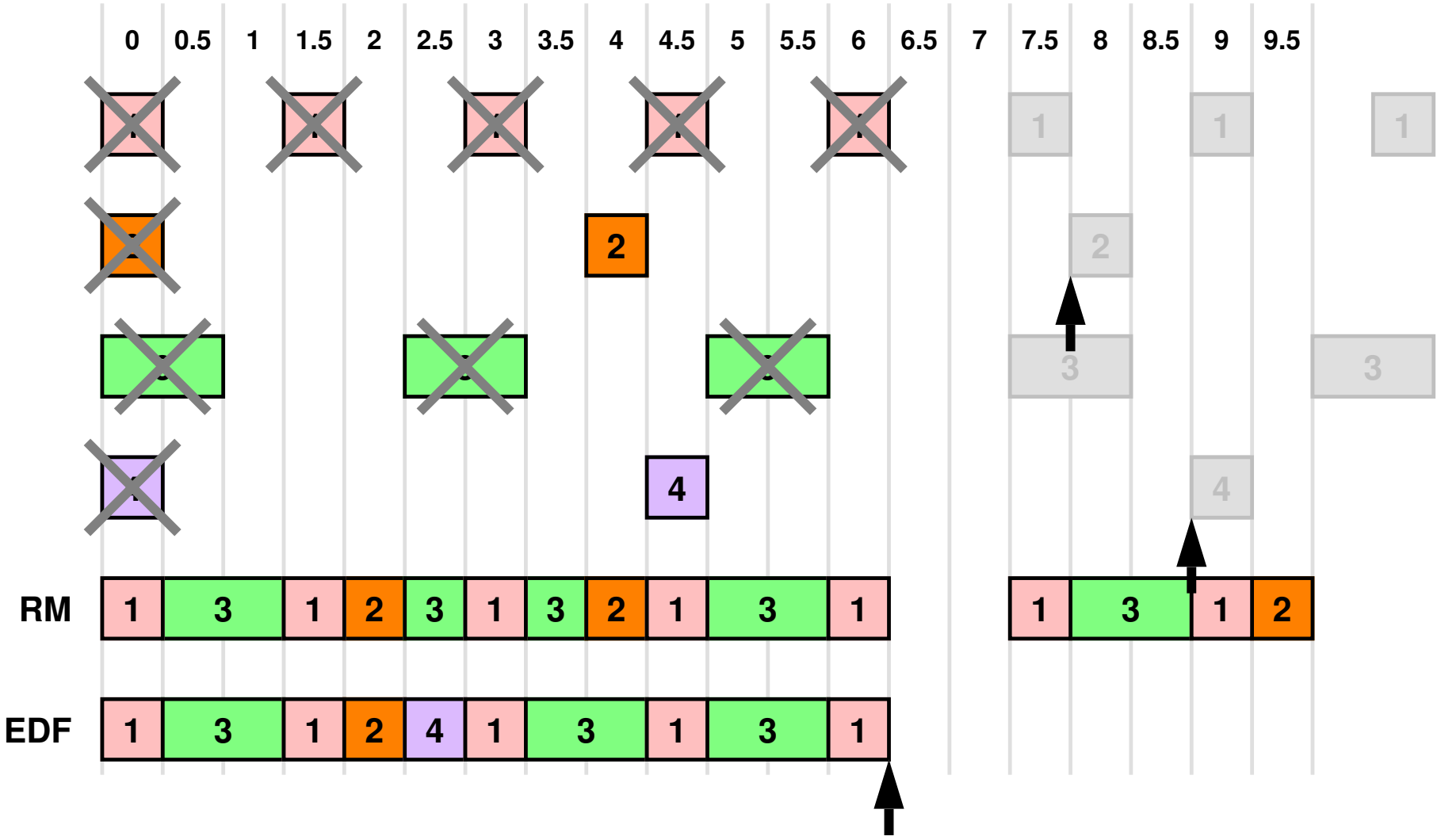
Earliest Deadline First



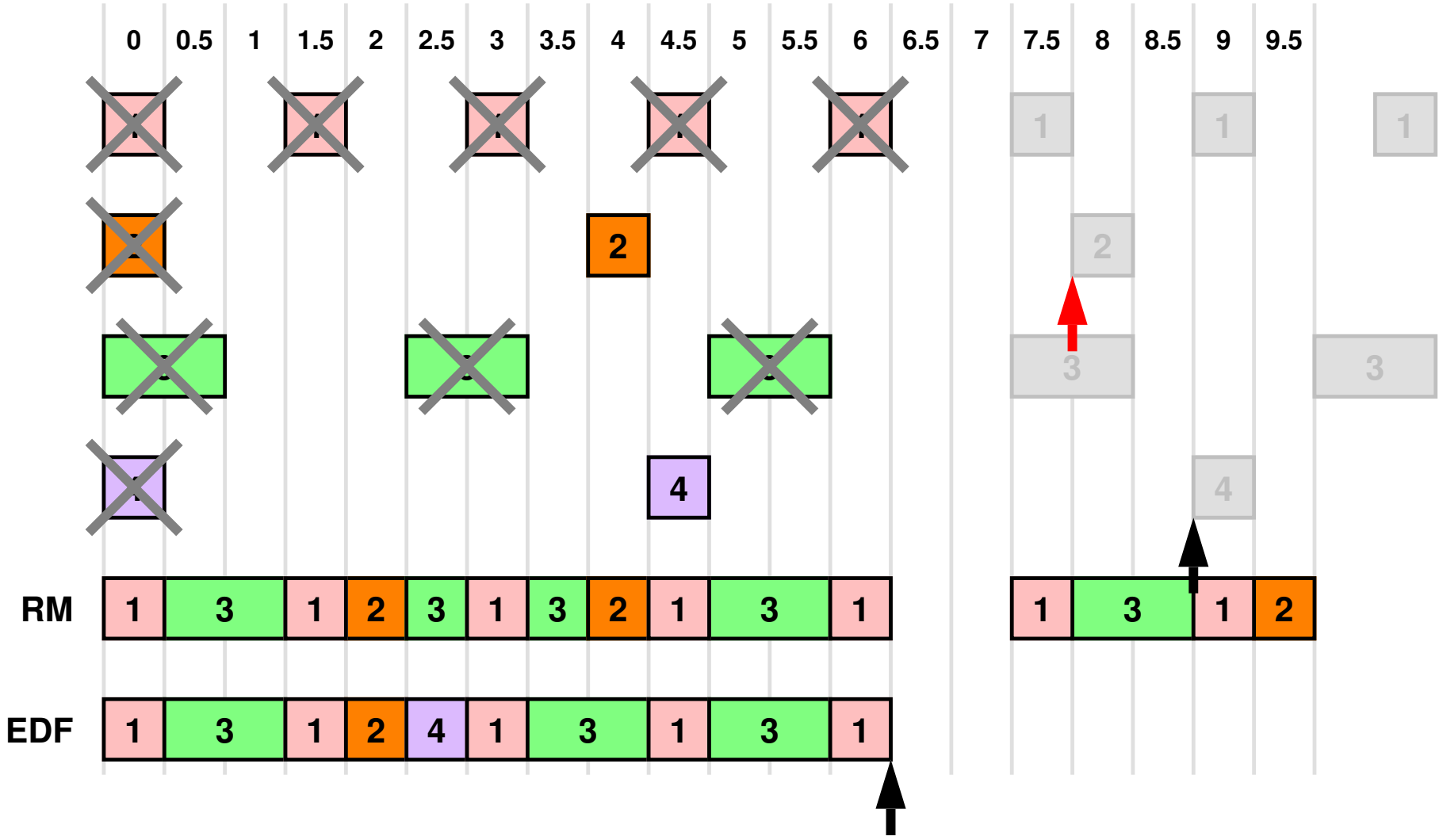
Earliest Deadline First



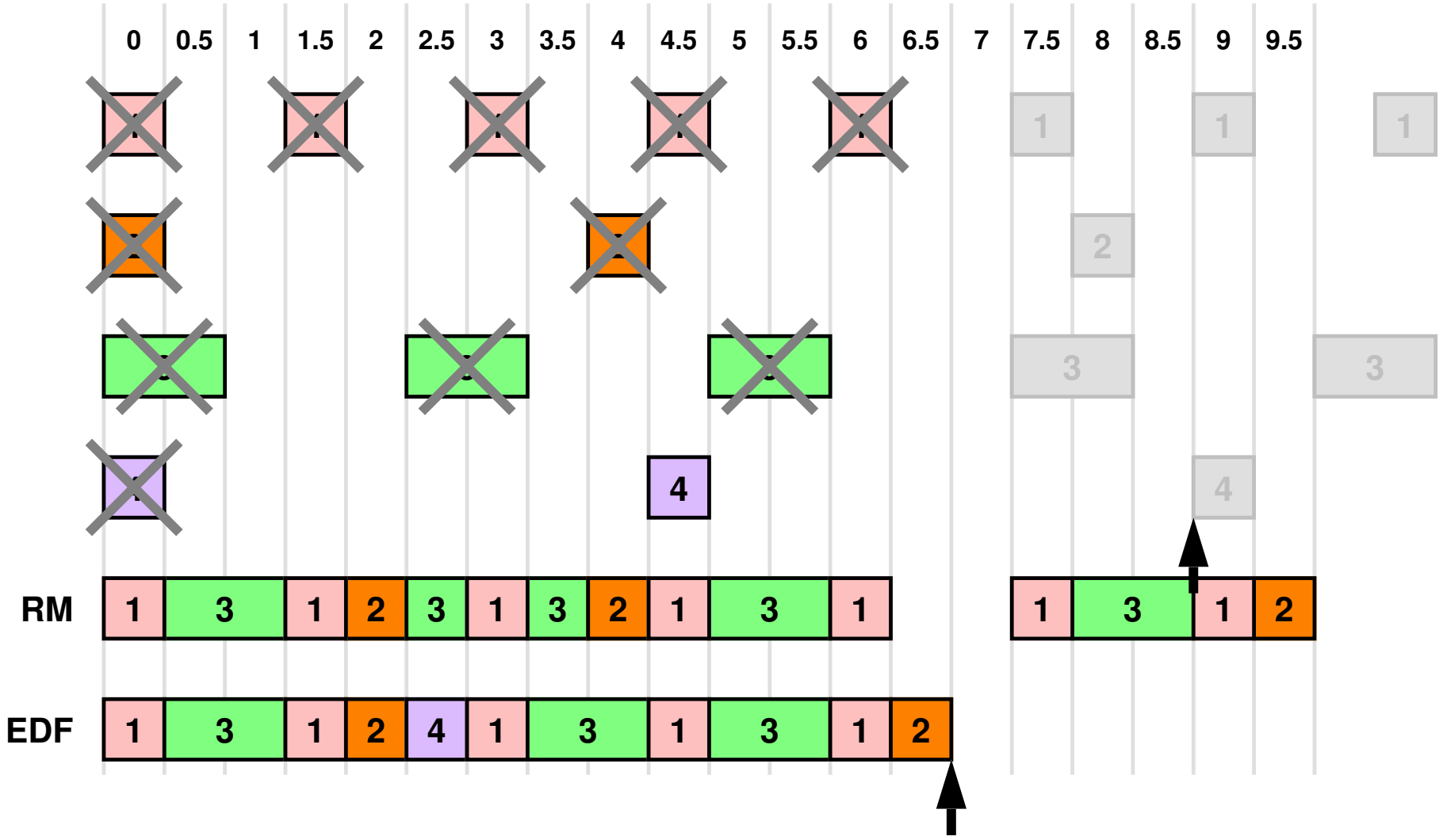
Earliest Deadline First



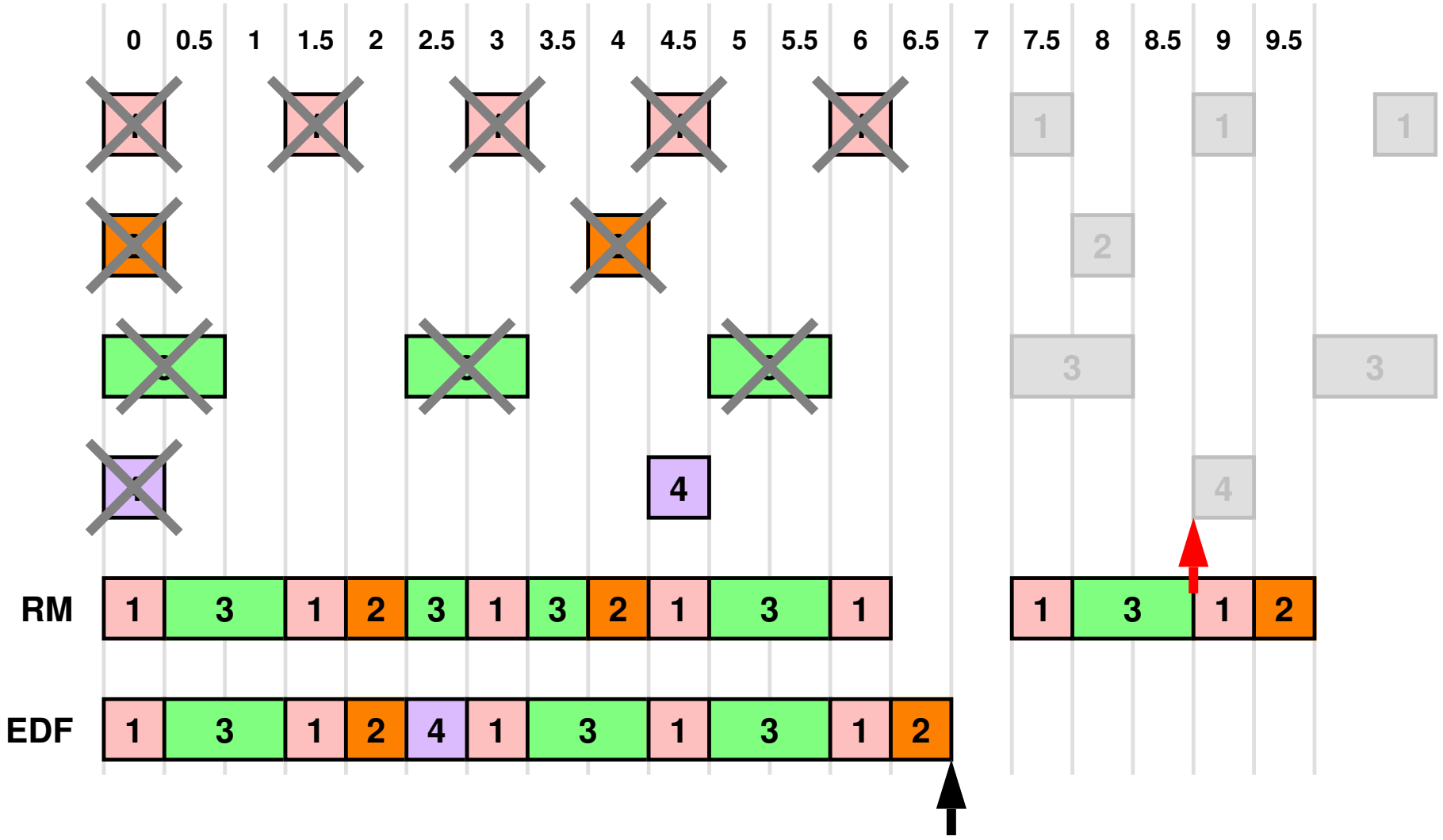
Earliest Deadline First



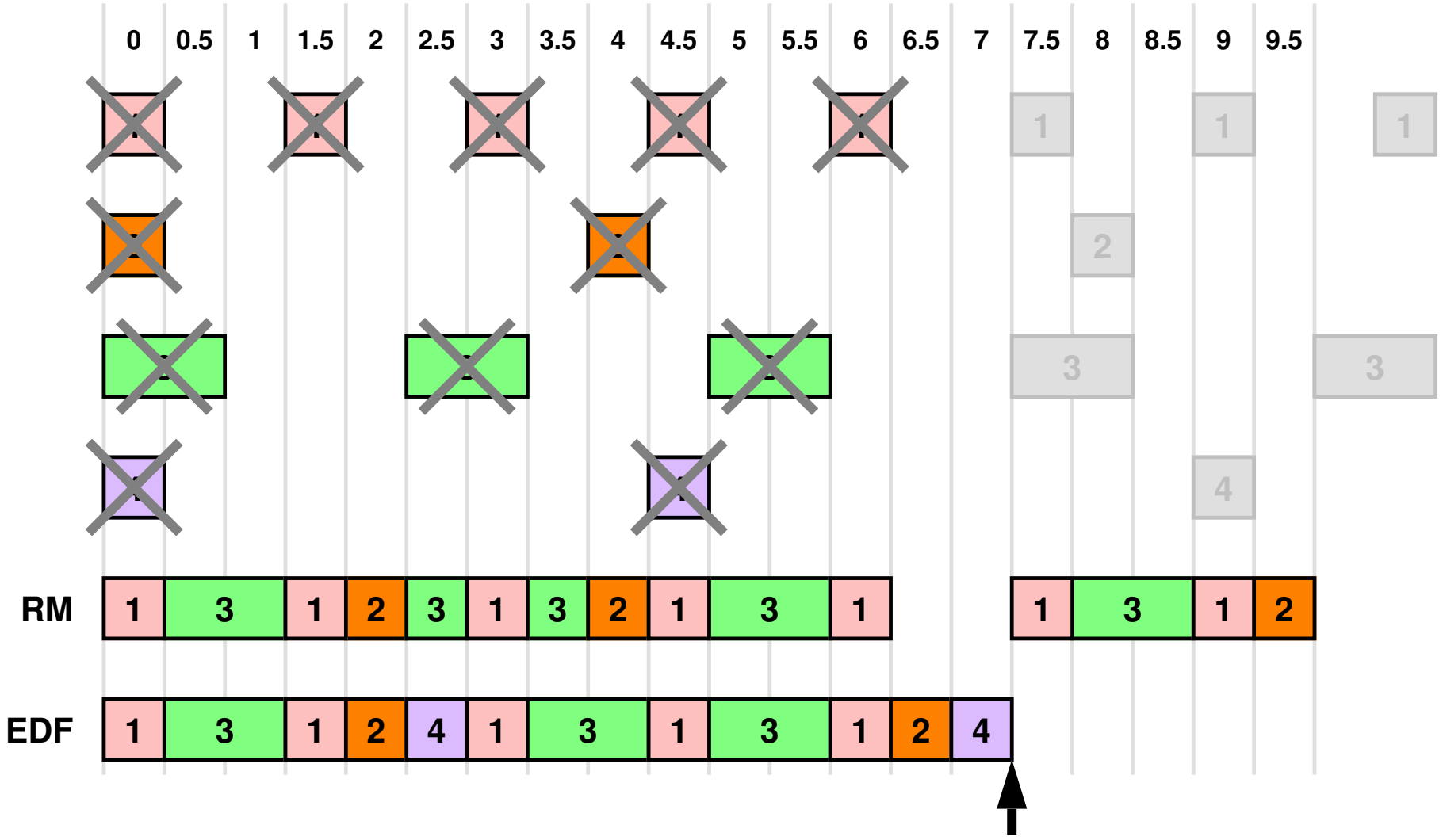
Earliest Deadline First

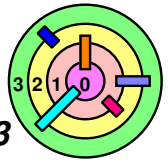


Earliest Deadline First

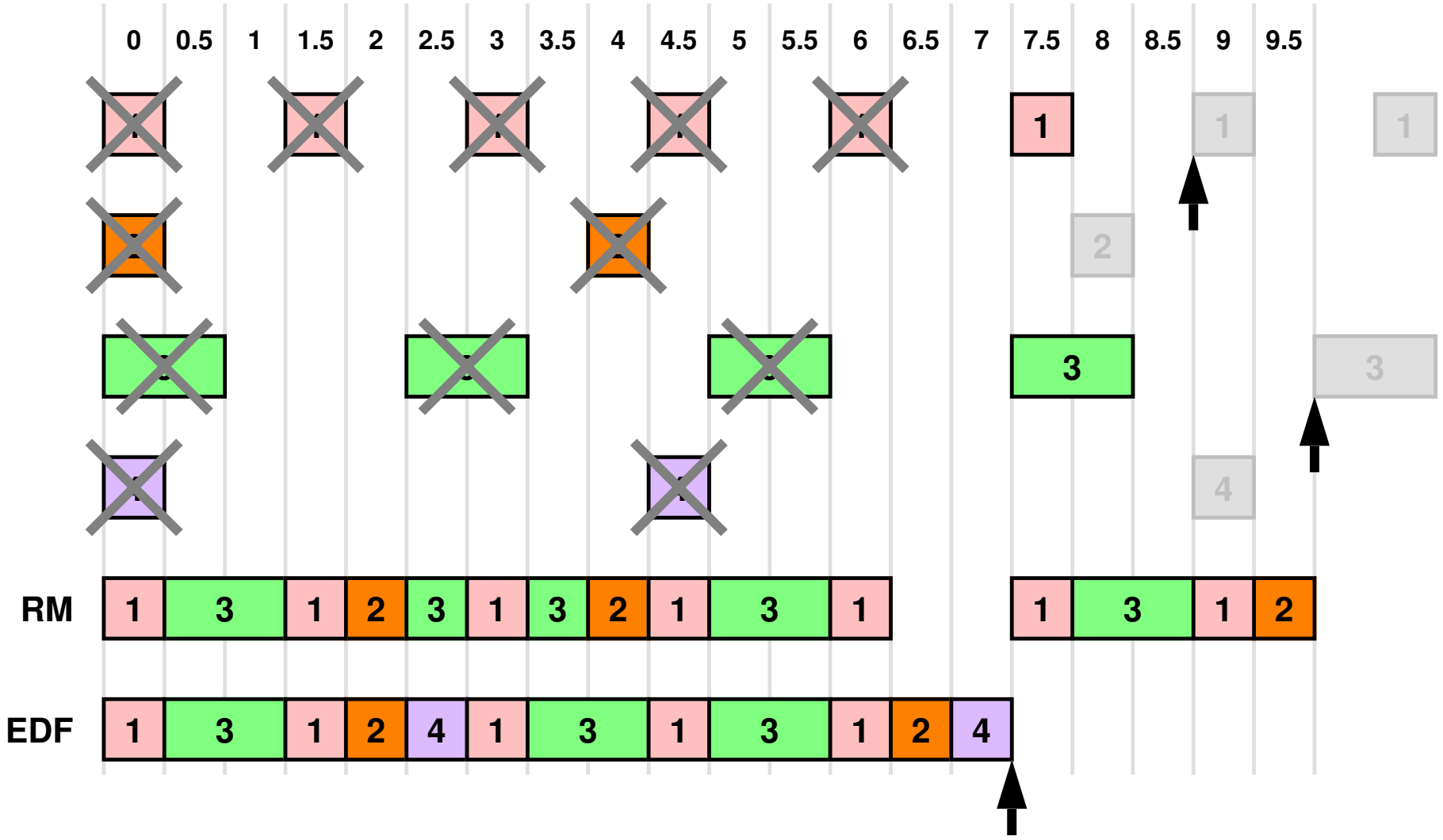


Earliest Deadline First

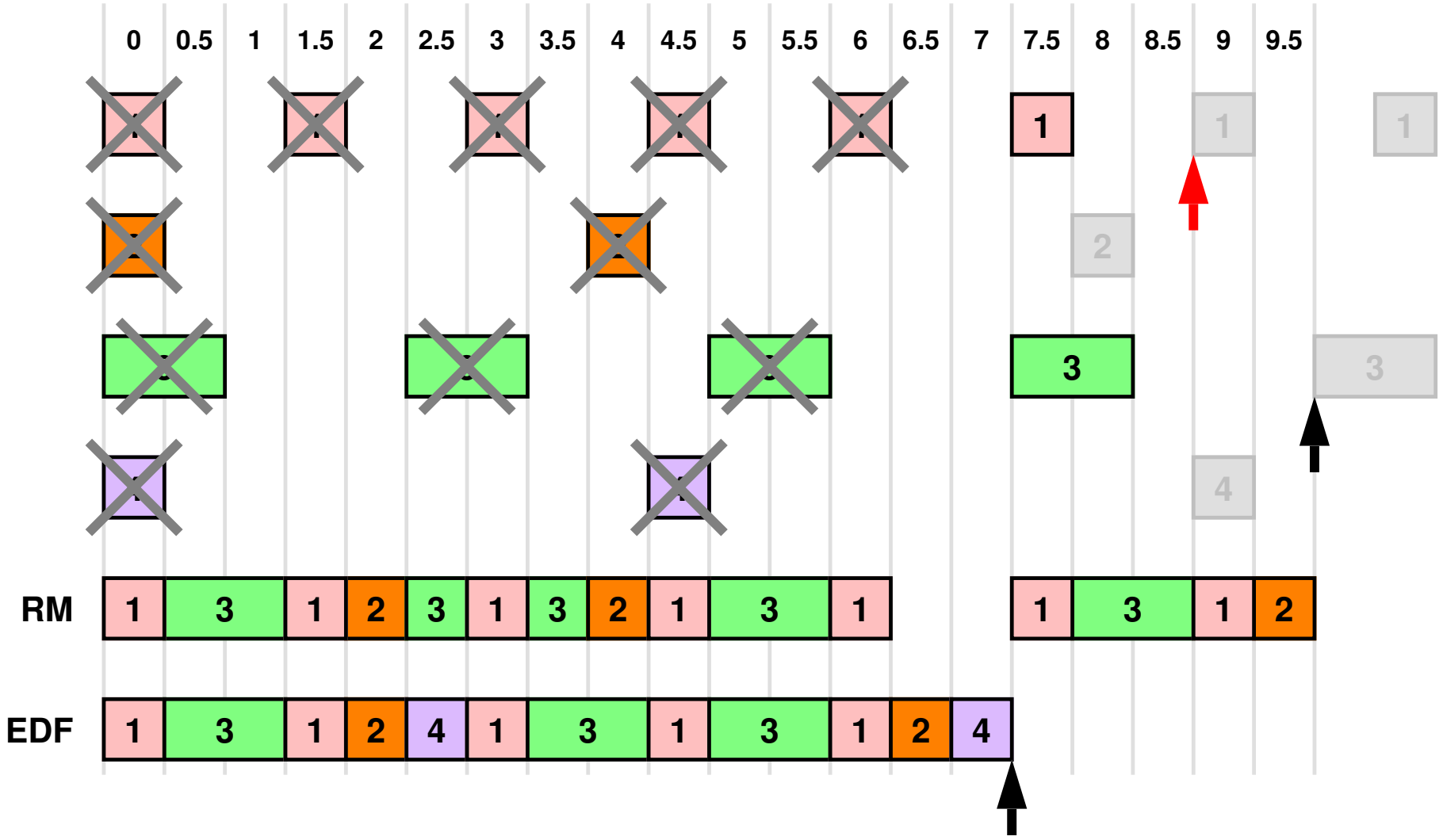




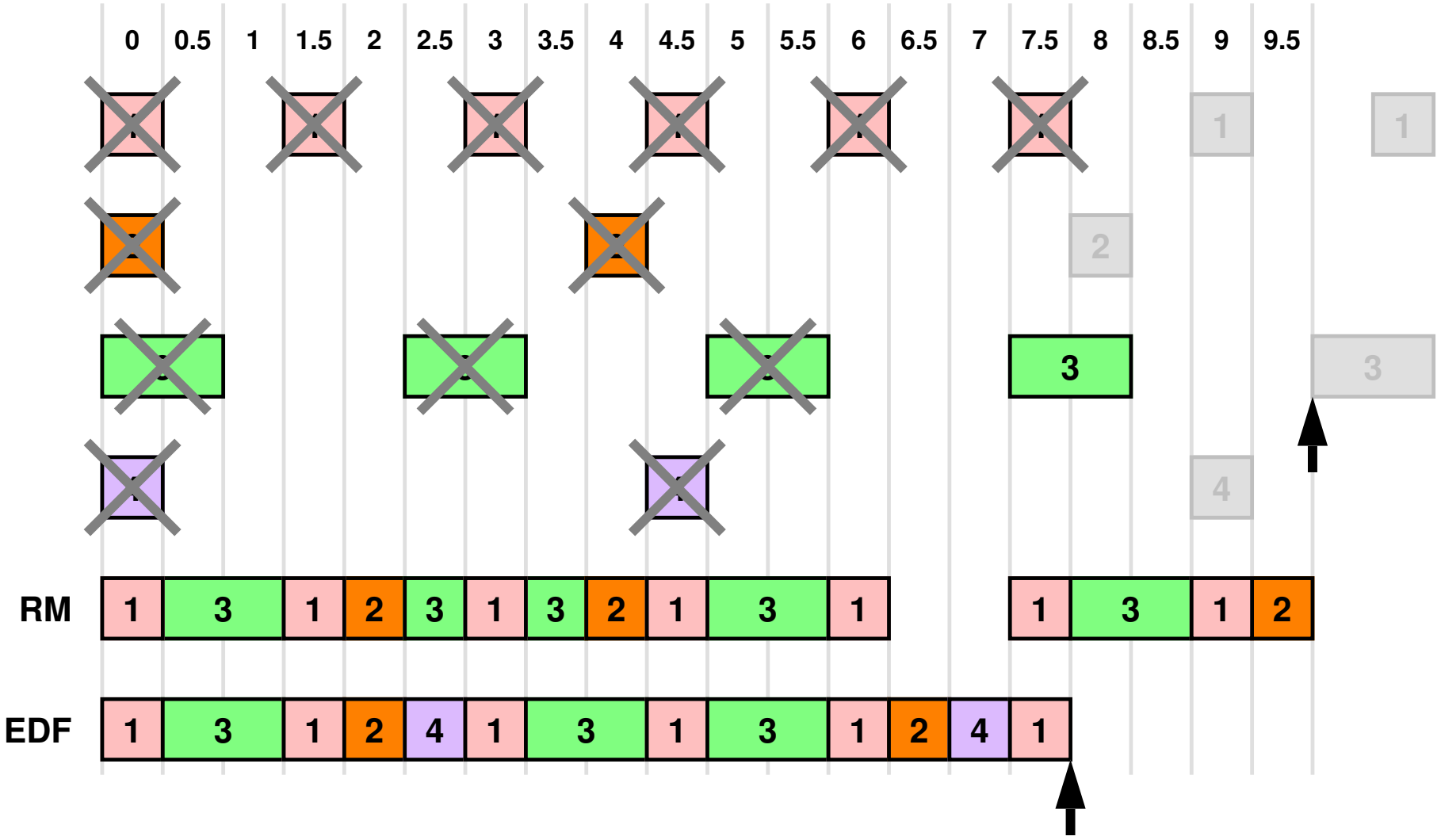
Earliest Deadline First

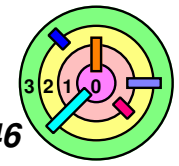


Earliest Deadline First

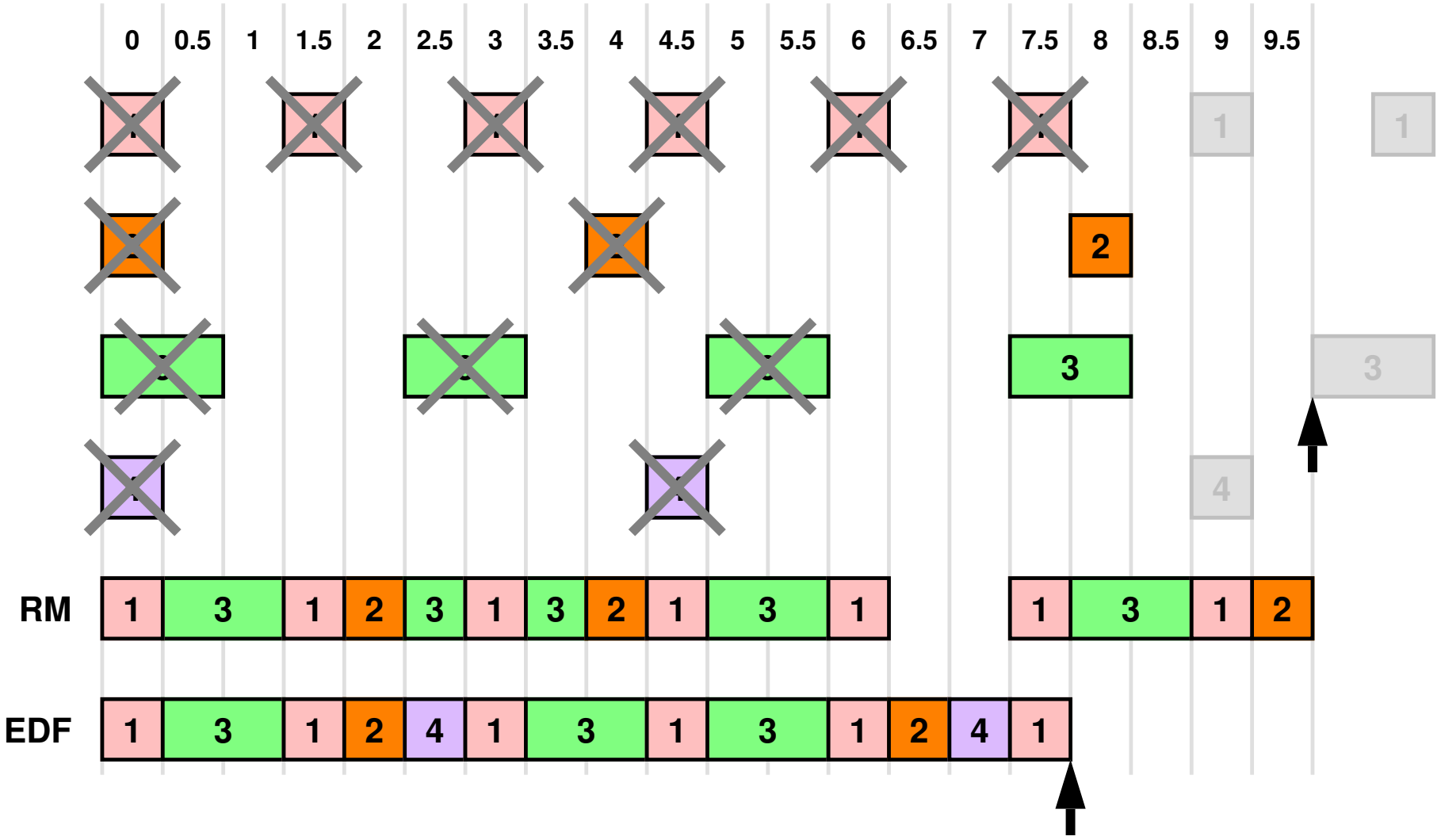


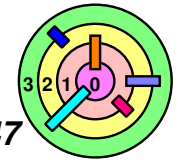
Earliest Deadline First



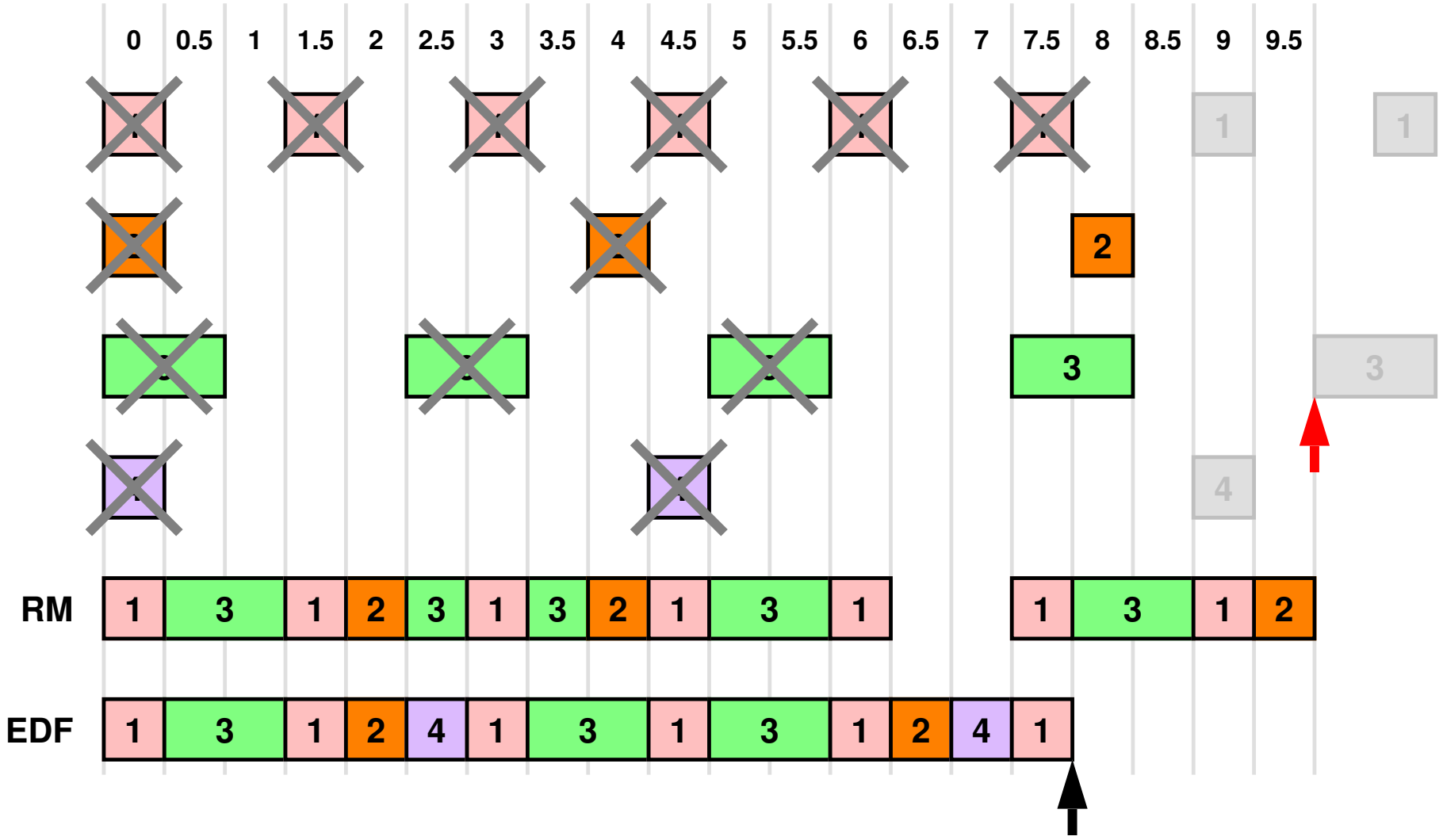


Earliest Deadline First

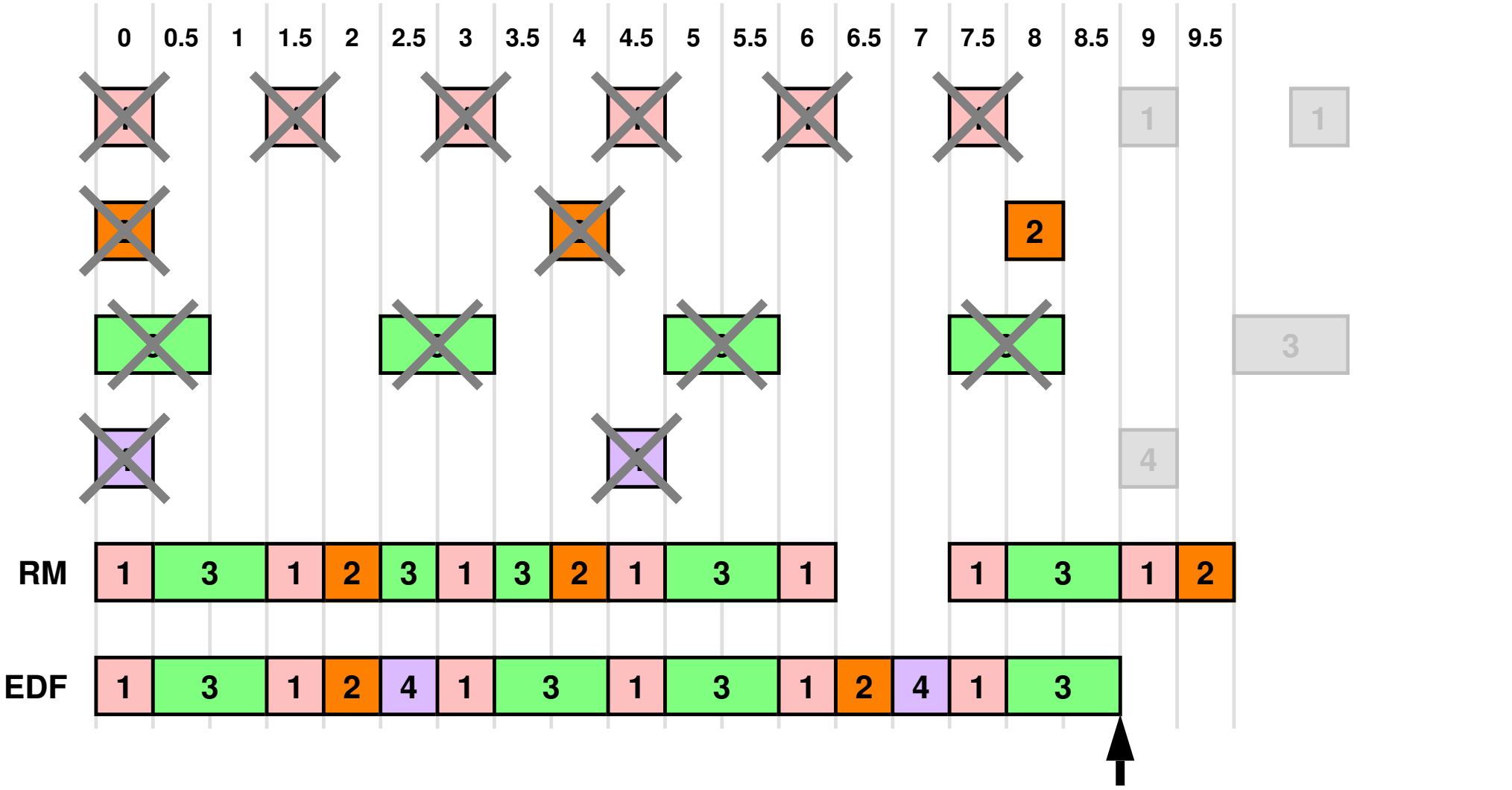




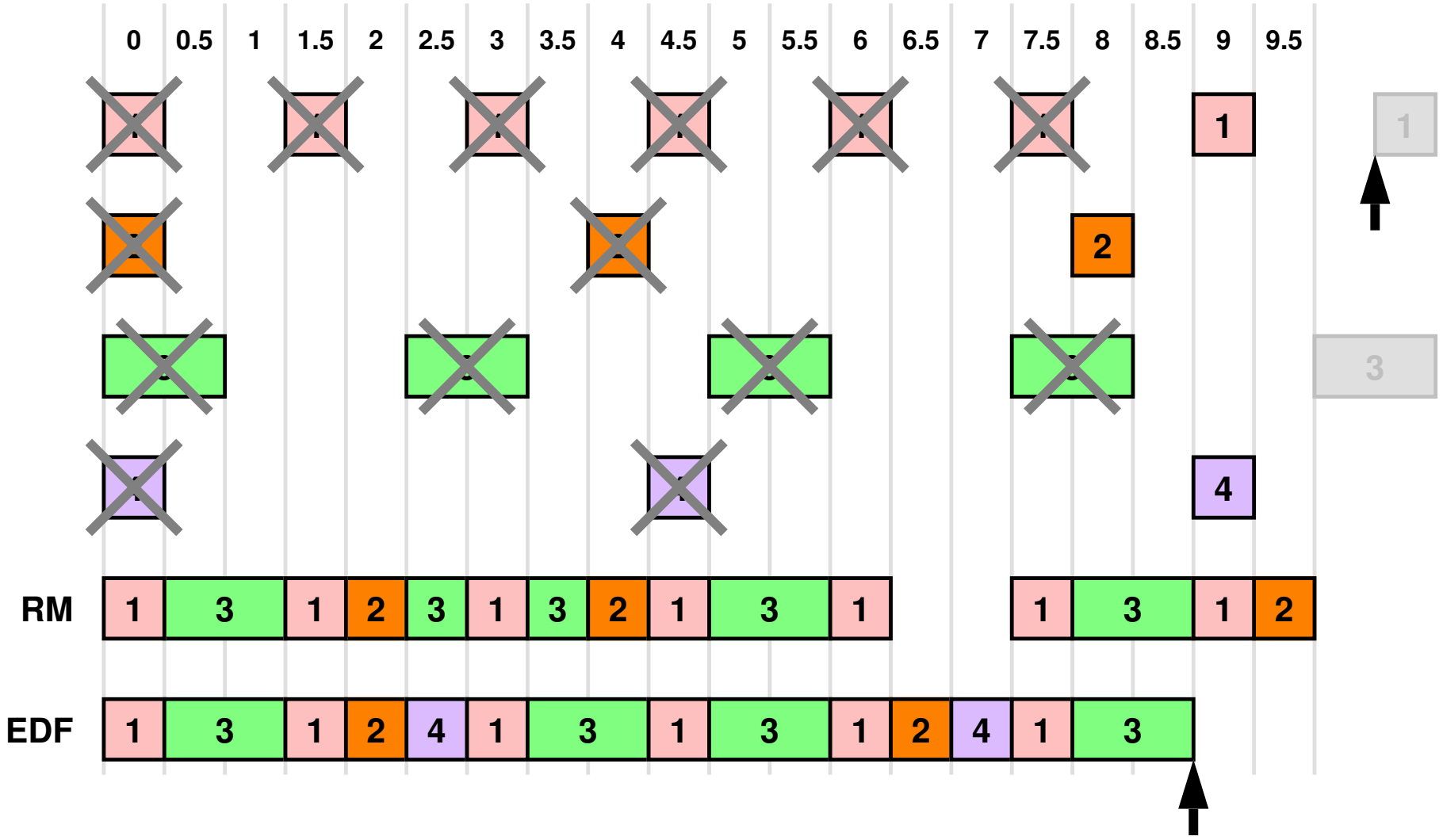
Earliest Deadline First



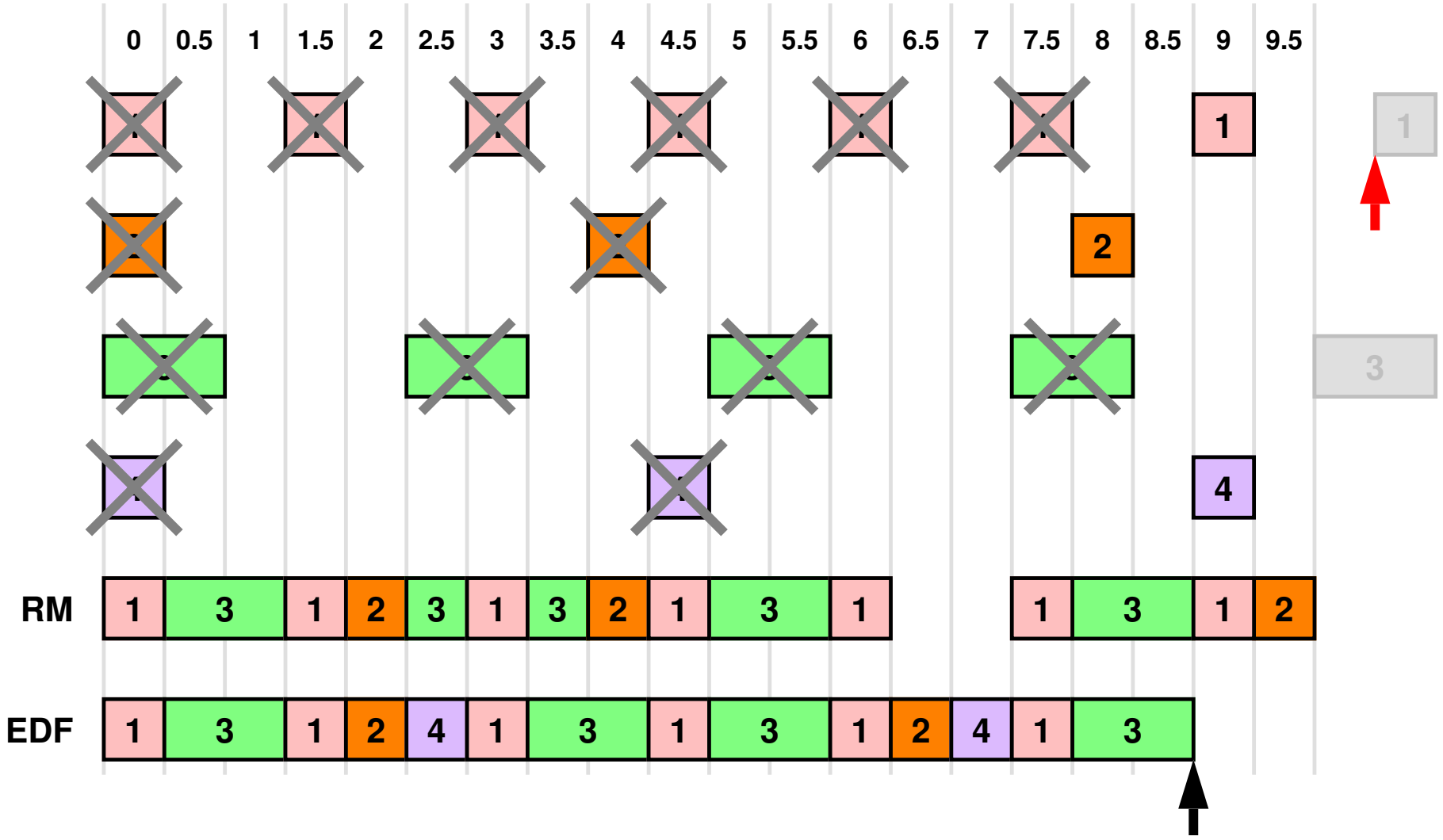
Earliest Deadline First



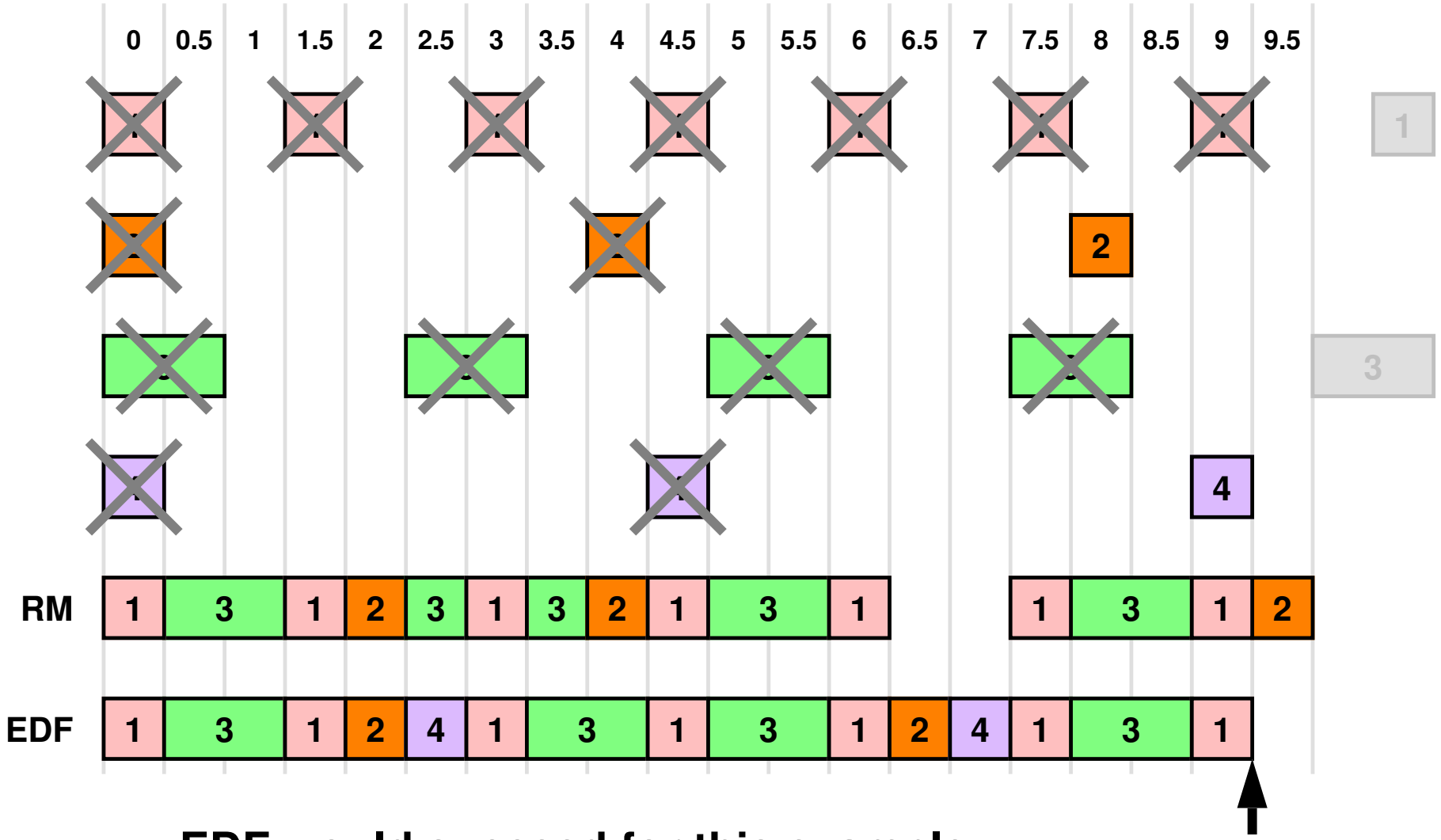
Earliest Deadline First



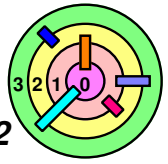
Earliest Deadline First



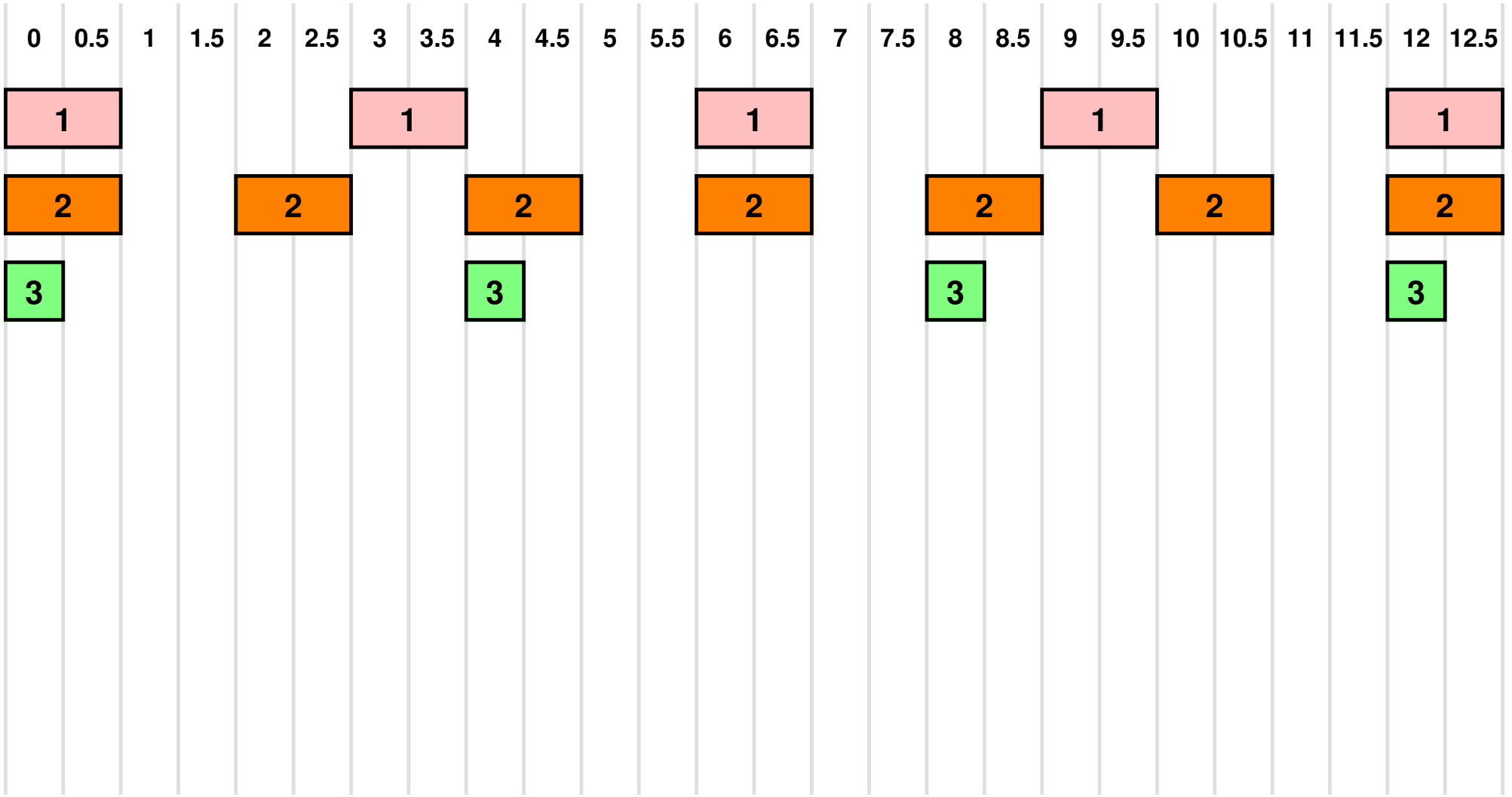
Earliest Deadline First



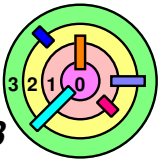
- EDF would succeed for this example
  - need to simulate till time = 180 to be certain



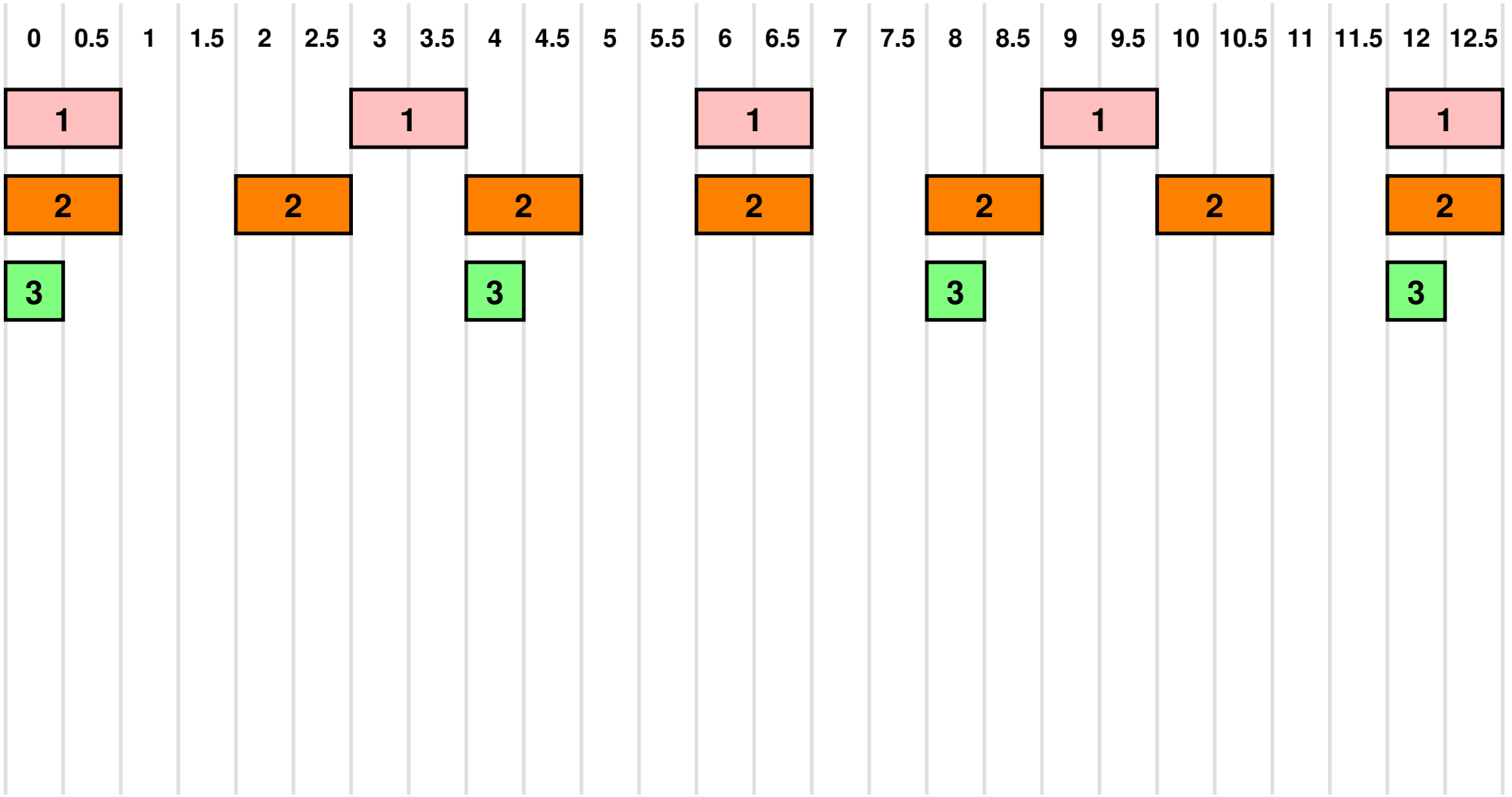
Phase Problems



- if jobs do not start in-phase, rate-monotonic scheduling still may work



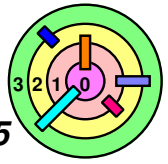
Phase Problems



see "extra slides" at the end

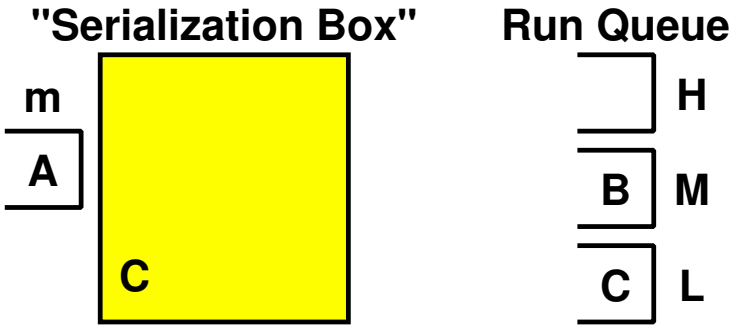
5.3 Scheduling

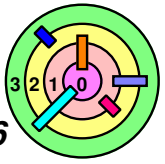
- Goals
- Scheduling Algorithms
- *Implementation Issues*
- Case Studies



Priority Problem

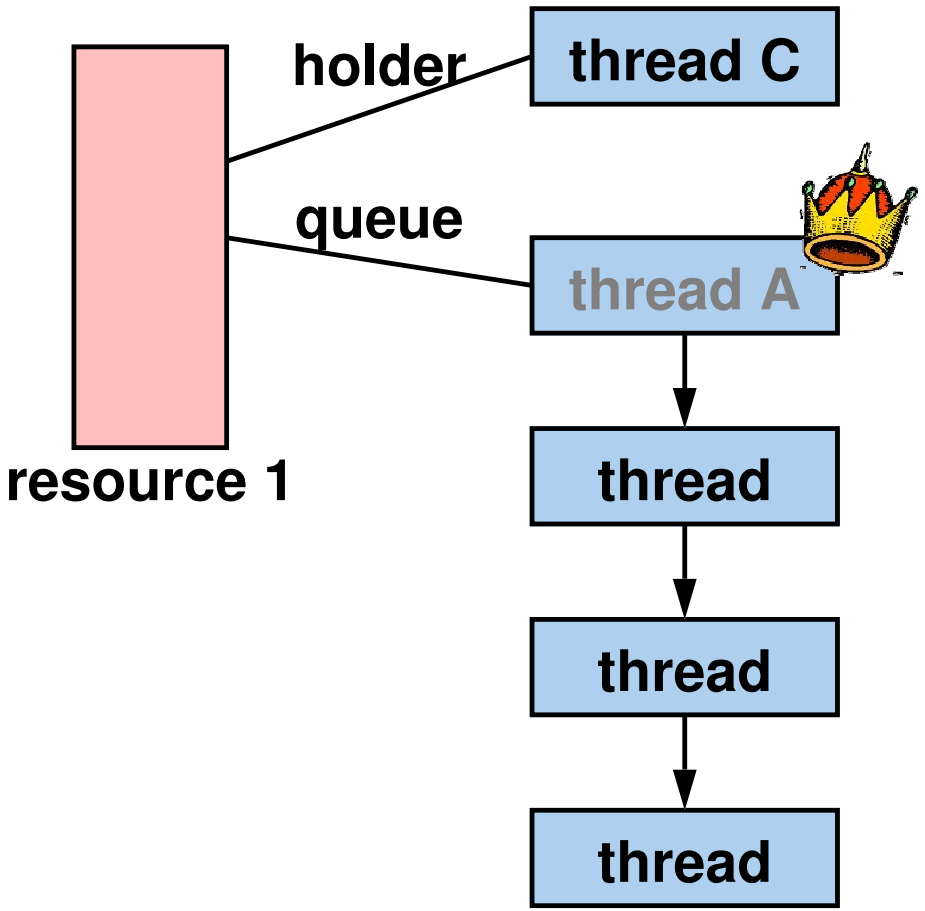
- High-priority thread A blocks on mutex 1
- Low-priority thread C holds mutex 1
- Thread C cannot run because medium-priority thread B is running
- A is effectively waiting at C's priority
  - *priority inversion*





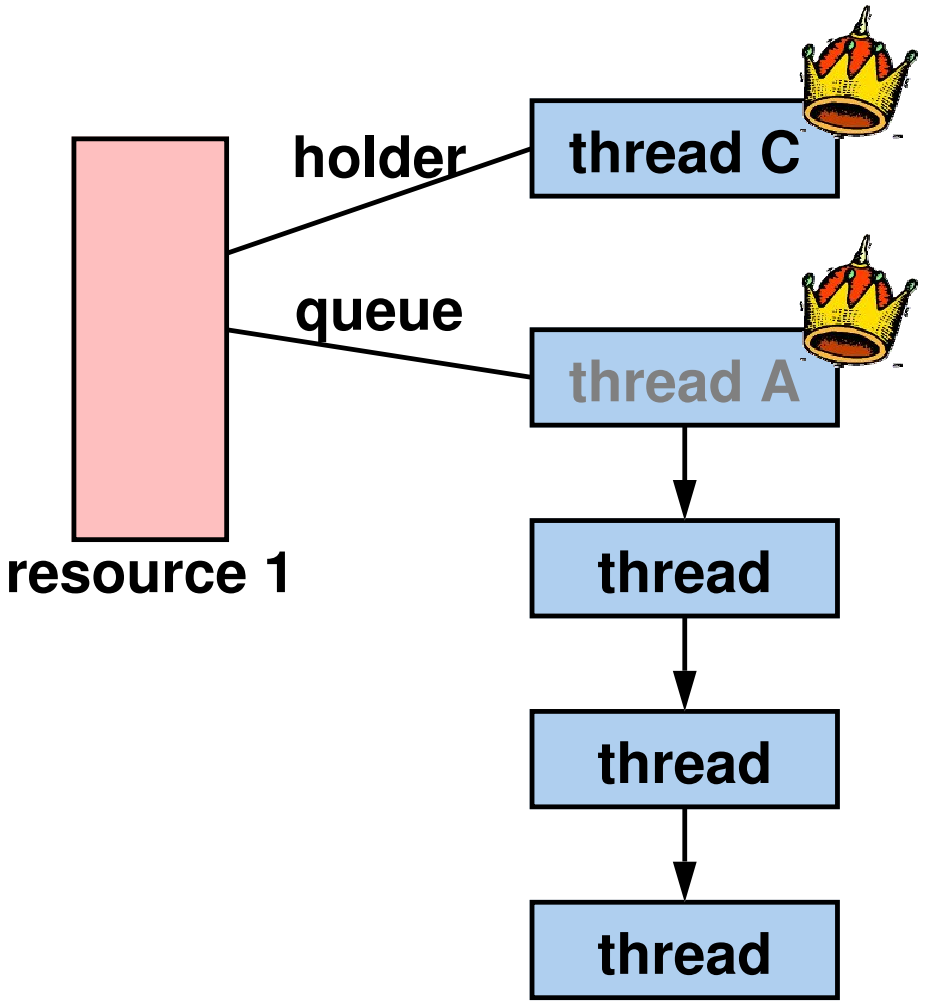
Priority Inheritance

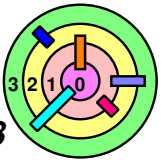
- While A is waiting for resource held by C, it gives C its priority



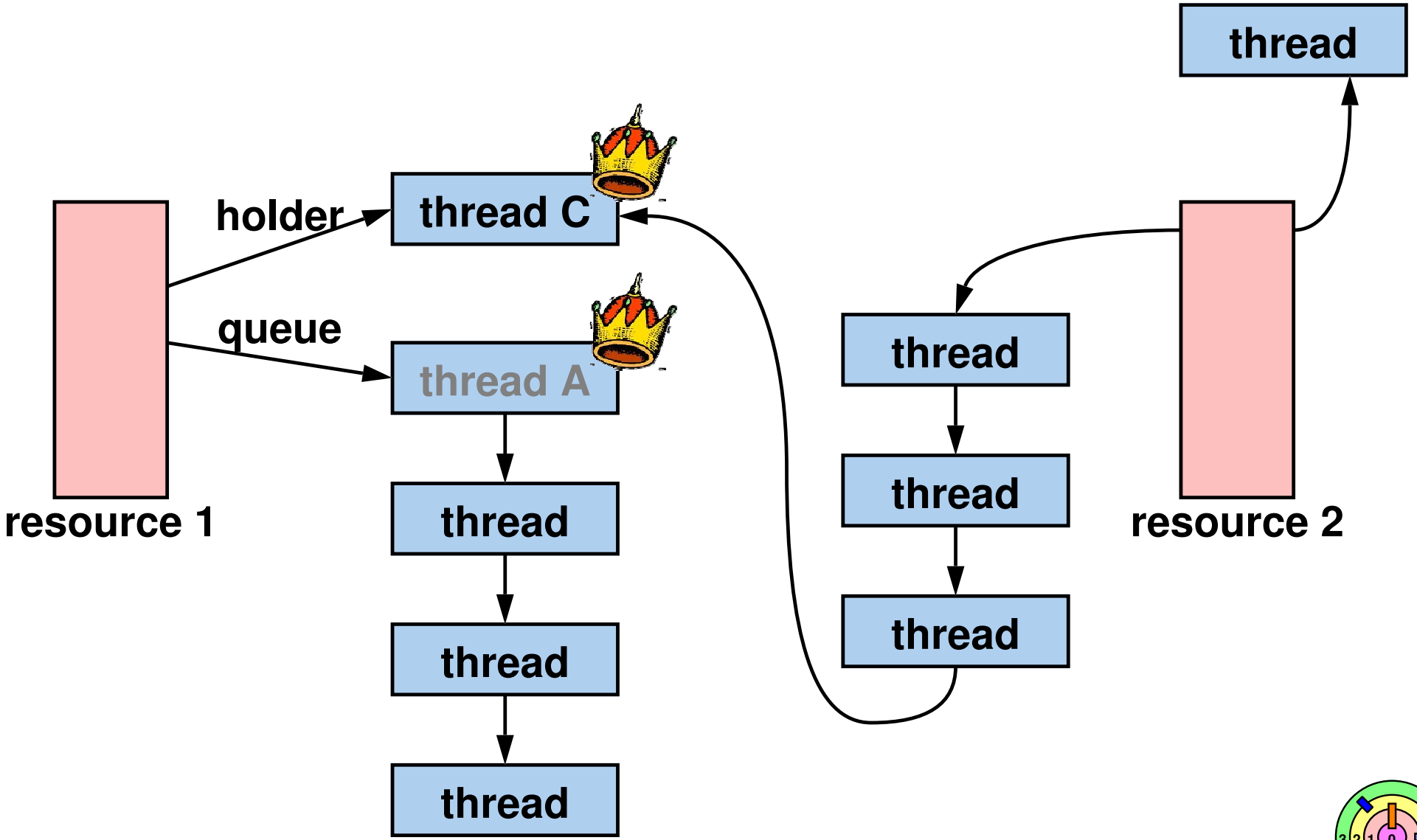
Priority Inheritance

- While A is waiting for resource held by C, it gives C its priority





Cacading Inheritance



Cacading Inheritance