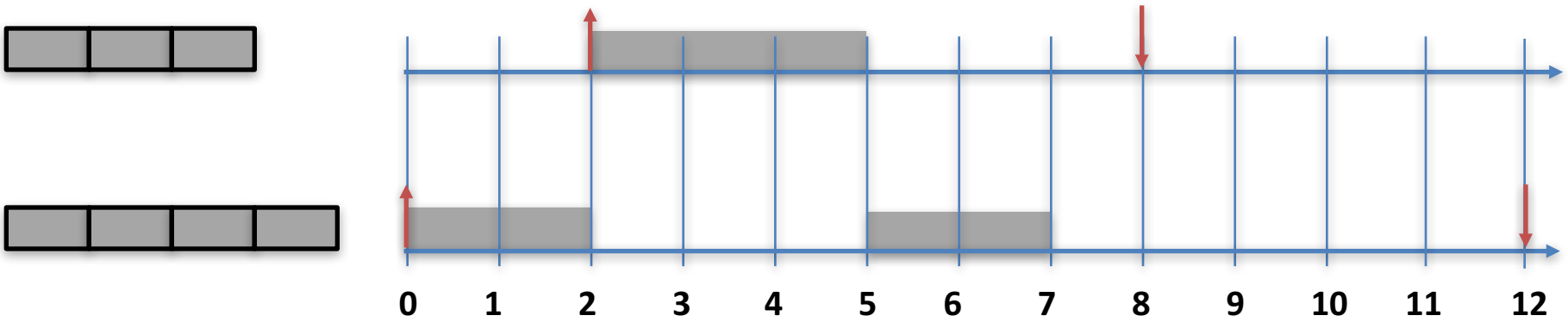
Open Problem Session

- **Sanjoy Baruah (Washington University in St. Louis):** Schedulability analysis for multiprocessor mutual exclusion
- **Antonios Antoniadis (MPI and Saarland University):** Is finding a TSP tour that visits a set of n planes in a 3-dimensional Euclidean space is NP-hard or not?
- **Ben Moseley (Carnegie Mellon University):** Minimizing total flow time on identical machines with migration and preemption
- **Samir Khuller (Northwestern University):** Active time problem

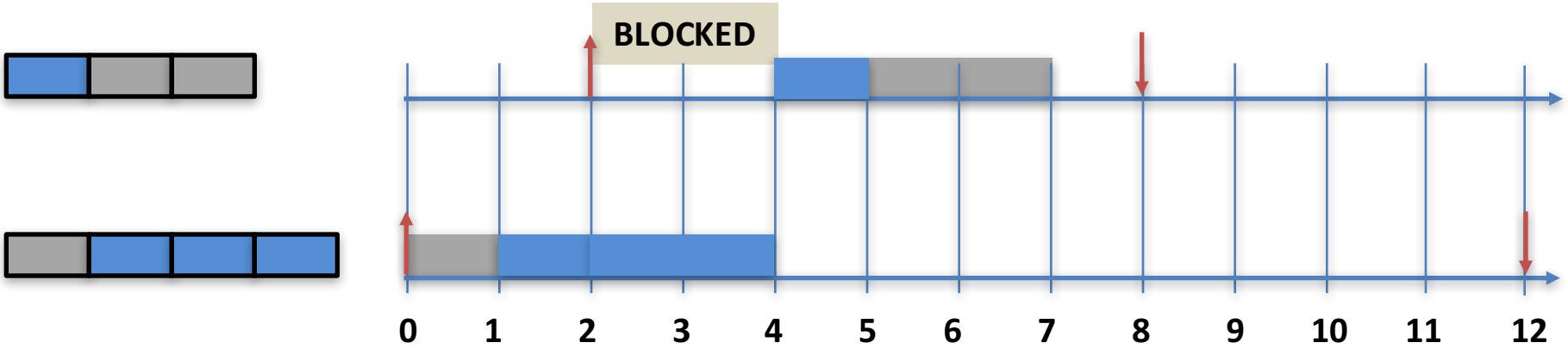
Independent jobs; Single (preemptive) machine



EDF

Scheduling Objective: Feasibility (all jobs should meet their deadlines)

Independent jobs; Single (preemptive) machine

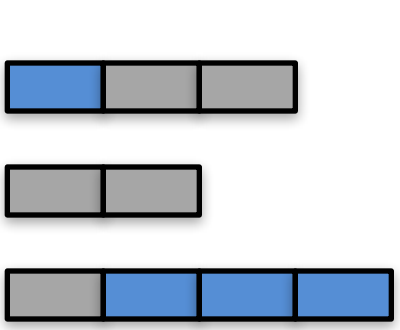


accessing a shared object

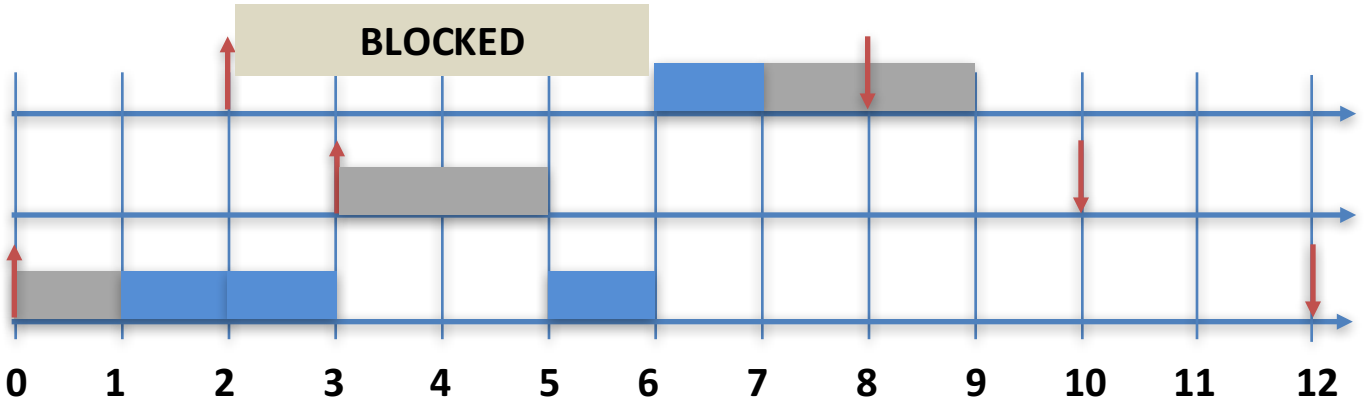
EDF

Non-preemptable; serially reusable

Independent jobs; Single (preemptive) machine

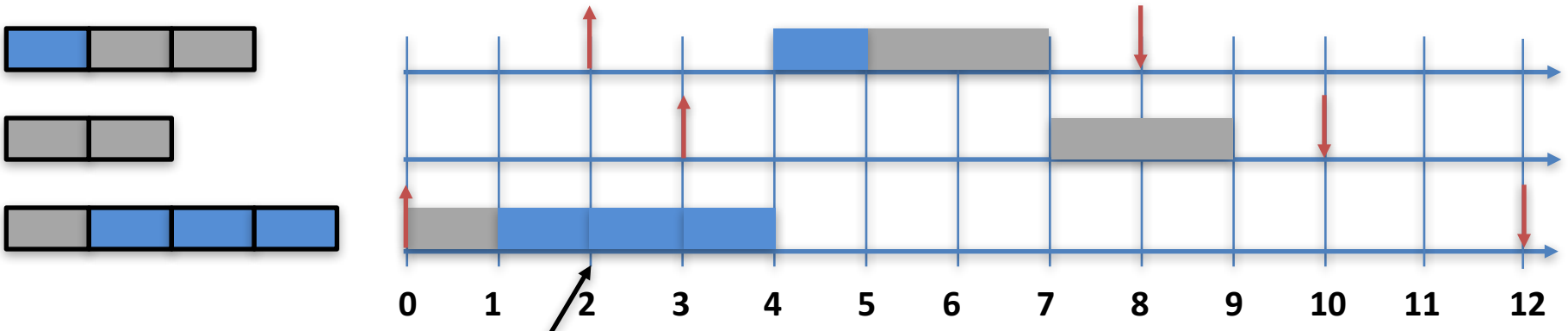


accessing a shared object



EDF

Independent jobs; Single (preemptive) machine



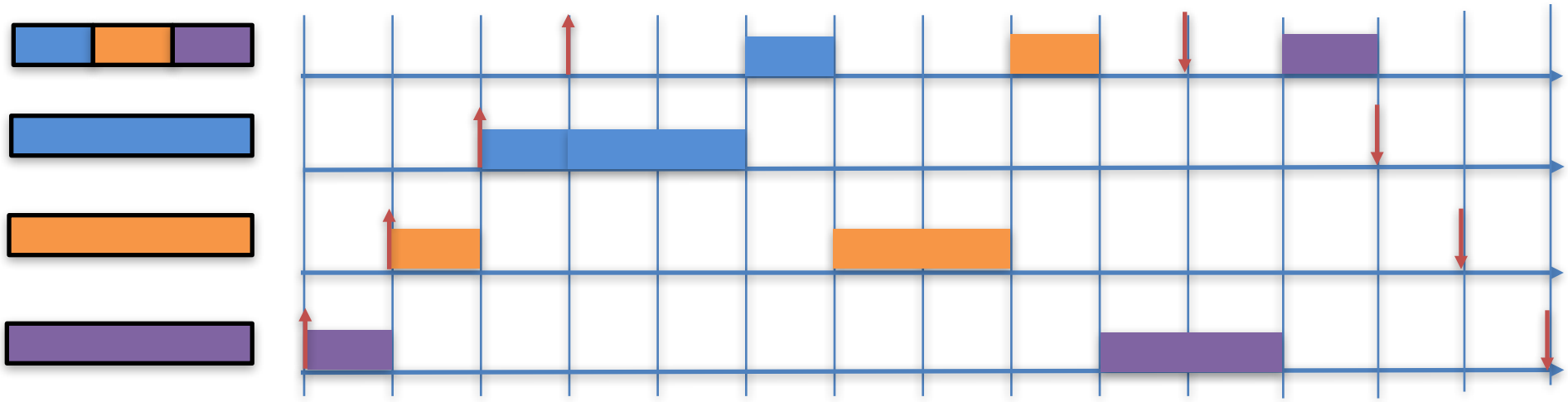
accessing a shared object

EDF + the **Priority Inheritance Protocol** (**PIP**)

Blocking job "inherits" blocked job's deadline

Independent jobs; Single (preemptive) machine

PROPERTY: Under the PIP, a job may be blocked once on each shared resource



The Stack Resource Policy (SRP)

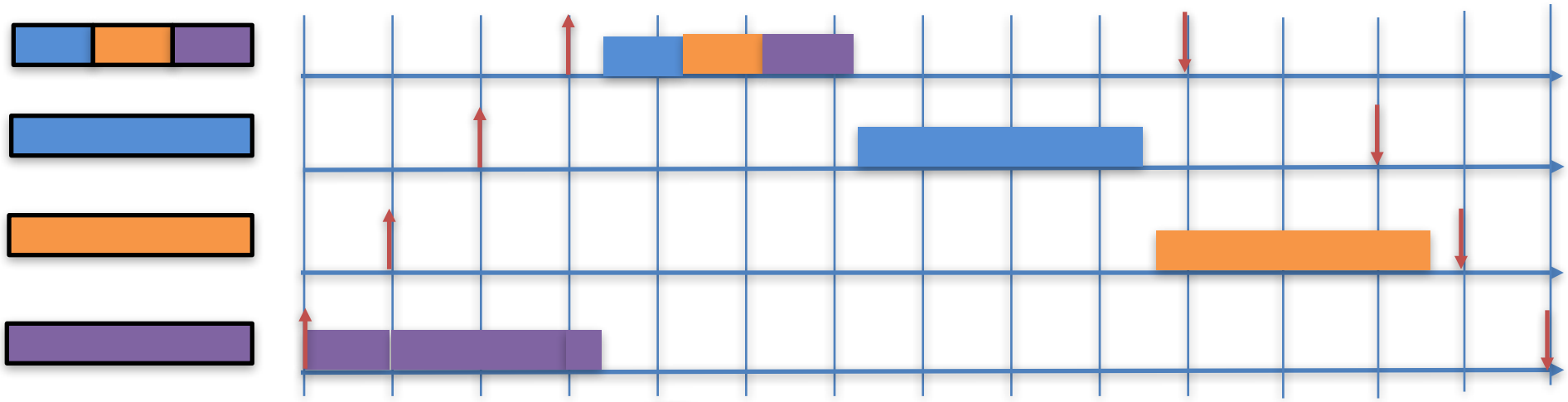
For each resource, a **resource ceiling**: earliest deadline of any job that uses it

At run-time, the **system ceiling** = minimum resource ceiling of any locked resource

A job may begin execution only if its deadline is less than the system ceiling

Independent jobs; Single (preemptive) machine

Scheduling Objective: Feasibility (all jobs should meet their deadlines)



The Stack Resource Policy (SRP)

For each resource, a **resource ceiling**: earliest deadline of any job that uses it

At run-time, the **system ceiling** = minimum resource ceiling of any locked resource

A job may begin execution only if its deadline is less than the system ceiling

Property: No job is blocked for more than one critical section

Property: **Optimality**

An **exact** feasibility test

Multiple Machines: The Open Problem

Design a **processor-scheduling algorithm** + **resource-access protocol**

with a **resource-augmentation bound**

Computational Complexity of TSP with Hyperplane Neighborhoods

- **Given:** n hyperplanes in d-dimensional Euclidean space (think planes in 3D).
- **Output:** A tour of minimal length that visits all hyperplanes, i.e., the tour must contain at least one point from each hyperplane.

Computational Complexity of TSP with Hyperplane Neighborhoods

- **Given:** n hyperplanes in d-dimensional Euclidean space (think planes in 3D).
- **Output:** A tour of minimal length that visits all hyperplanes, i.e., the tour must contain at least one point from each hyperplane.

Poly-time solvable? NP-hard?

Computational Complexity of TSP with Hyperplane Neighborhoods

- **Given:** n hyperplanes in d-dimensional Euclidean space (think planes in 3D).
- **Output:** A tour of minimal length that visits all hyperplanes, i.e., the tour must contain at least one point from each hyperplane.

Poly-time solvable? NP-hard?

What is known?

- Lines in 2D: Solvable in polynomial time [Jonsson 2002]
- $d \geq 3$: EPTAS, [Antoniadis, Fleszar, Hoeksma, Schewior 2019]

Computational Complexity of TSP with Hyperplane Neighborhoods

- **Given:** n hyperplanes in d-dimensional Euclidean space (think planes in 3D).
- **Output:** A tour of minimal length that visits all hyperplanes, i.e., the tour must contain at least one point from each hyperplane.

Poly-time solvable? NP-hard?

What is known?

- Lines in 2D: Solvable in polynomial time [Jonsson 2002]
- $d \geq 3$: EPTAS, [Antoniadis, Fleszar, Hoeksma, Schewior 2019]
  Ruben's talk, tommorow 10:45, Session B!

Open Problems on Approximate Scheduling on Multiple Identical Machines

Benjamin Moseley
Tepper School of Business, Carnegie Mellon University
Relational-AI

Carnegie Mellon University

rAI

Identical Machine Environment

- n jobs arrive over time offline
  - Each job has processing time p_j
  - Each job j must be scheduled
    - Preemption and migration is allowed
- m identical machines

Open Question

- Is there a o(log n) approximation for total flow time?
  - Logarithmic is the best possible if migration not allowed.
    - Upper bound $O(\min\{\log n/m, \log P\})$
    - Lower bound $\Omega(\sqrt{\frac{\log P}{\log\log P}})$
  - My opinion: It is interesting to resolve complexity with migration for such a basic problem

Open Question on a Generalization

- Each job has j has a cost function $g_j(t) : \mathbb{R}^+ \to \mathbb{R}^+$
  - Cost function is monotonically increasing
- Given a completion C_j for each job j
- Goal is to minimize $\min \sum_j g_j(C_j)$

- Any nontrivial approximation?
  - With speed augmentation this is easy
  - O(log log nP) possible if jobs arrive at time 0 or when m=1
    - Interesting to get an O(1) approximation

Thank you!

Questions?

A Greedy 2 Apx for the Active Time Problem

Saurabh Kumar & Samir Khuller

Active Time Problem (ATP)

Set of n preemptible jobs J = {1, 2, ..., n} where each job j has length p_j and window $[r_j, d_j]$.

Time is divided into unit length slots

We can schedule g distinct job units in a single time slot.

Objective - Schedule all of the jobs in a feasible way while minimizing the number of **active time slots**.

Background

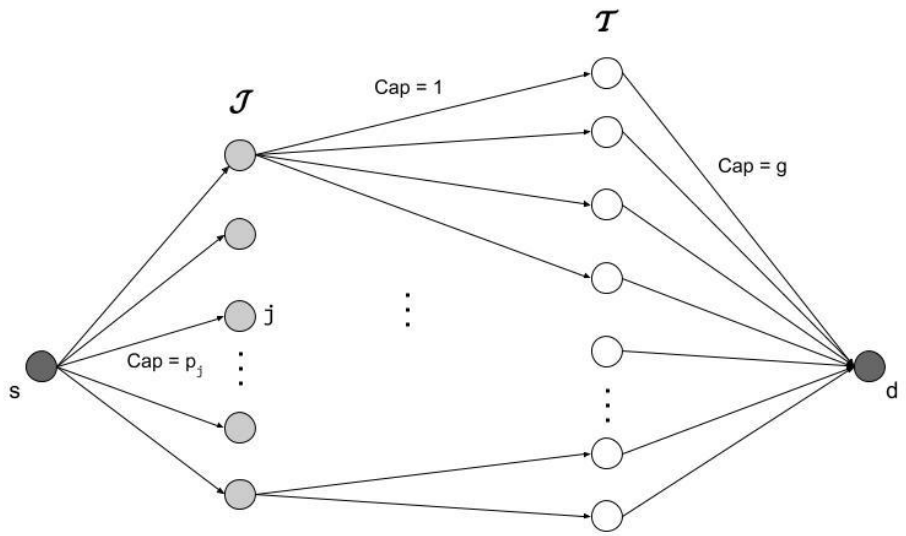
ATP with unit length jobs can be solved exactly.

For non-unit length jobs, the complexity status is still open.

What we have so far:

1. A minimal feasible solution gives a 3 apx.
2. An LP rounding scheme and now this greedy algorithm give a 2 apx.

Connection to Network Flow

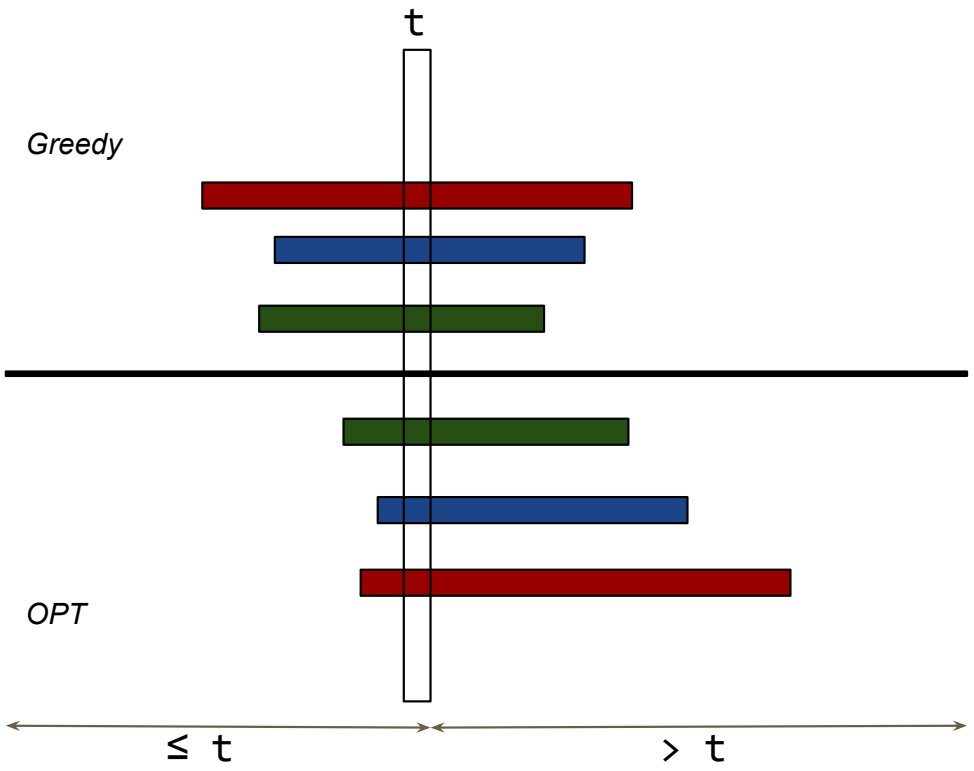


Given a set of open time slots, we can determine whether a feasible schedule is possible using max flow.

A Left to Right Greedy Solution is a 2 Apx

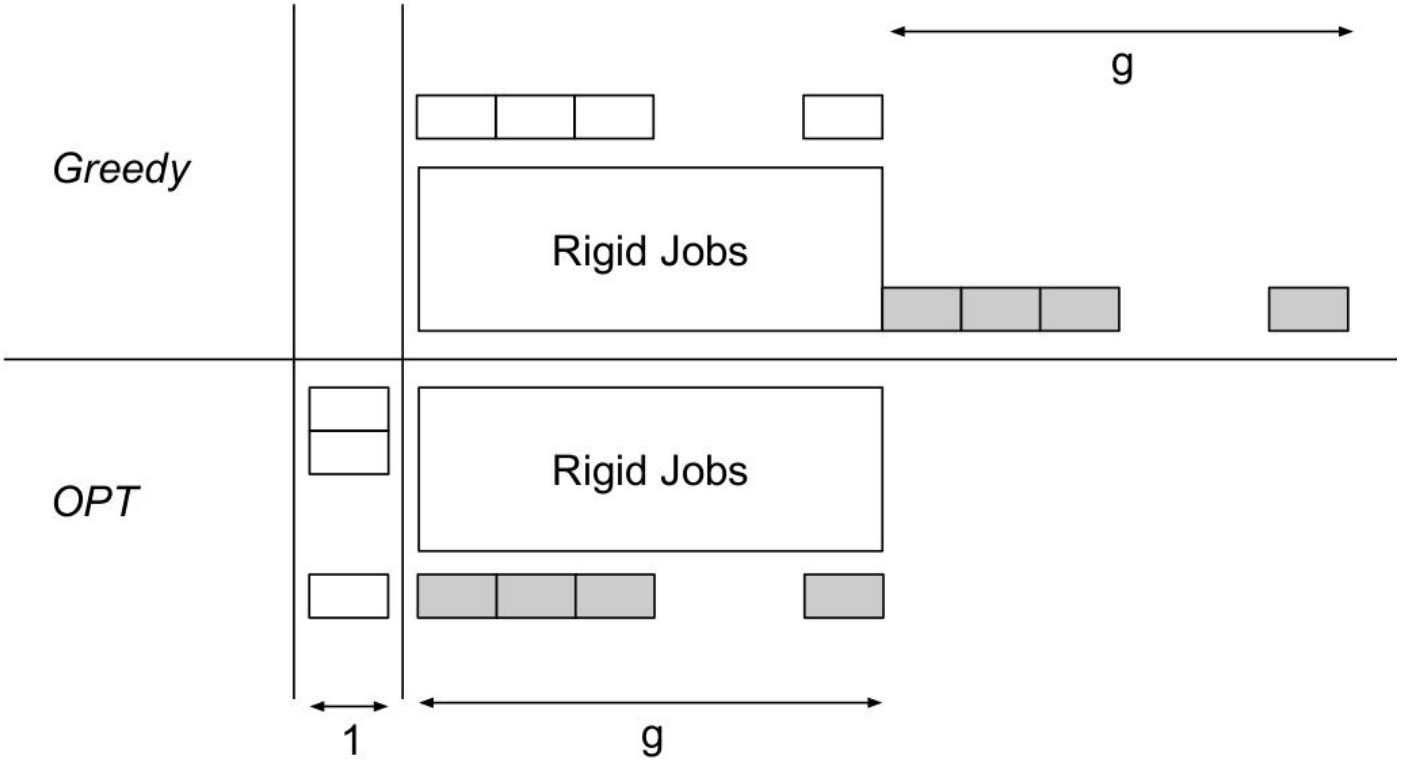
```
t = 0
While t < D:
        Close t
        If feasible schedule no longer possible:
                Open t
        t++
```

Analysis - OPT Comparison



At any non-full slot t in our greedy schedule, there exists at least one job j s.t OPT schedules at least as much of j as ALG does in $[r_j, t]$.

Tight Example



Conclusion

2 Apx matches the best known bound (= complex LP rounding scheme).

The complexity status of this problem is still unknown.

Local Search (close X slots open < X) may help us break the 2 barrier.

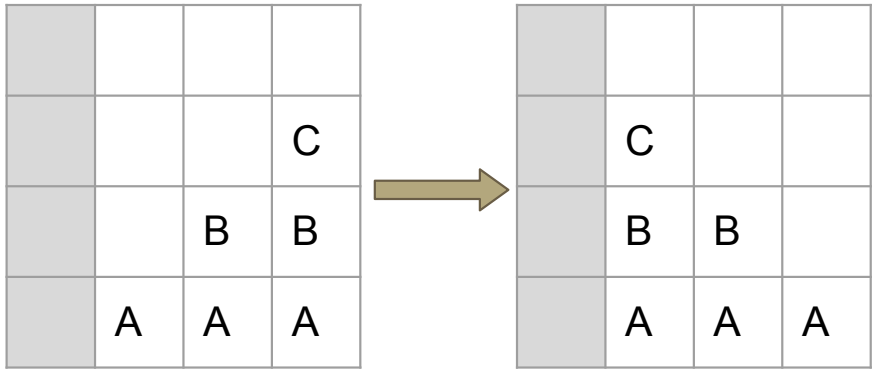
r_a r_b r_c d

A	A	A	B	B	B	C	C	C

						C	C	C
						B	B	B
						A	A	A

We focus solely on the #slots used.

Analysis - Left Shifting



For any job j in non-full slot t, j must be present in every non-full slot in $[r_j, t]$.

Analysis - Putting it Together

From the previous two steps, the number of non-full slots <= OPT.

From a mass bound, the number of full slots <= OPT

So final solution < = 2 * OPT