Minimalistic Predictions for Online Class Constraint Scheduling

Dorian Guyot and Alexandra Lassota

UNI FR

TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY

Context

Scheduling

- Assigning jobs to machines while minimizing some metric.
- Applications: product planning, data placement, load balancing, …

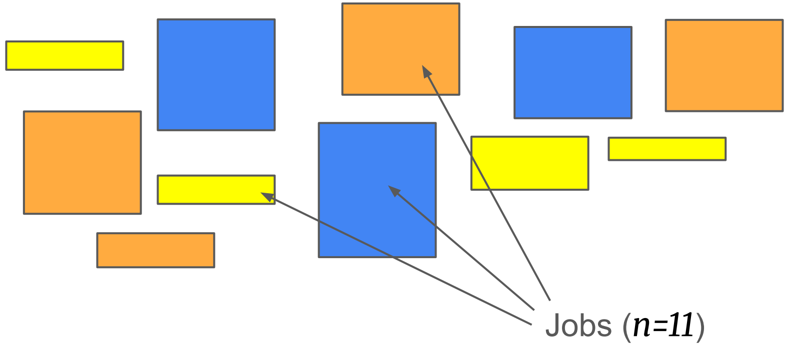
Online

- Jobs are revealed one after the other
- An irrevocable decision has to be taken before learning about the next one

Setting

Given:

Setting



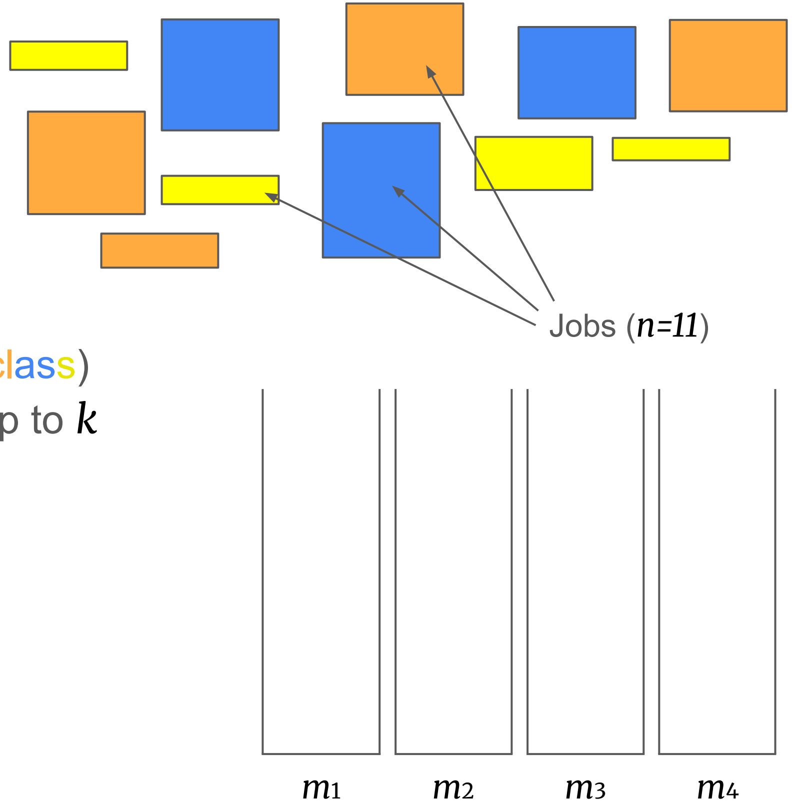
Given:

- A stream of n jobs revealed one by one (size↕, class)

Setting

Given:

- A stream of n jobs revealed one by one (size↕, class)
- A set of m identical machines, each accepting up to k different **classes** (colors) of jobs



Example schedule:
4 machines (m=4)

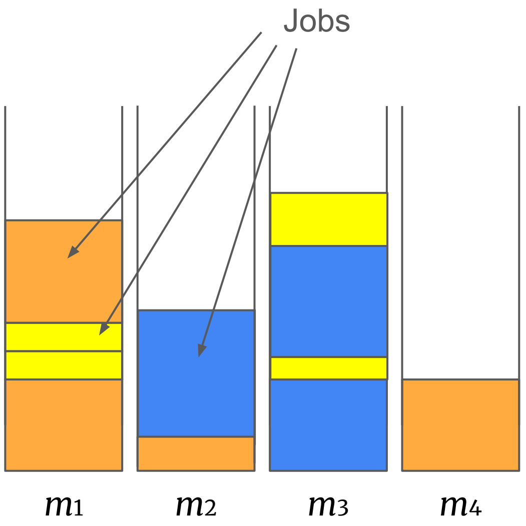
Setting

Given:

- A stream of n jobs revealed one by one (size↕, class)
- A set of m identical machines, each accepting up to k different **classes** (colors) of jobs

Goal:

- Schedule the jobs efficiently (minimize the makespan)



Example schedule:
4 machines (m=4)
11 jobs (n=11)
max 2 colors/machine (k=2)

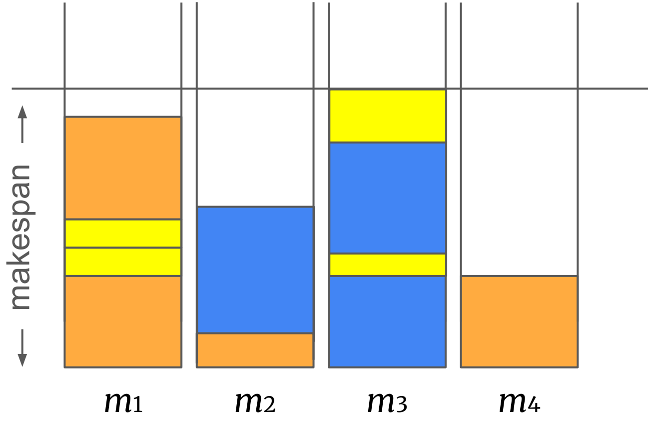
Setting

Given:

- A stream of n jobs revealed one by one (size↕, class)
- A set of m identical machines, each accepting up to k different **classes** (colors) of jobs

Goal:

- Schedule the jobs efficiently (minimize the makespan)



Example schedule:
4 machines ($m=4$)
11 jobs ($n=11$)
max 2 colors/machine ($k=2$)

Setting

Given:

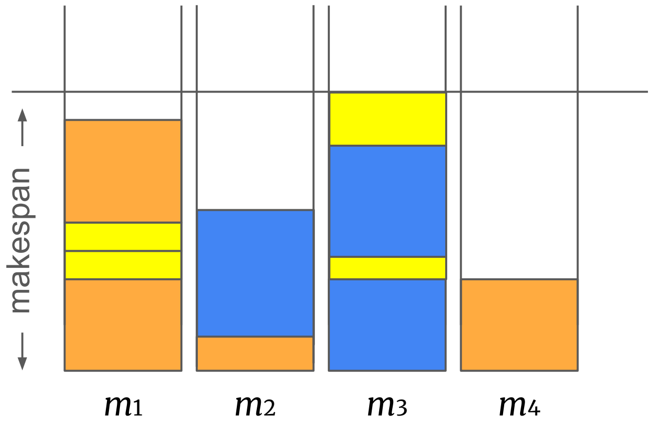
- A stream of n jobs revealed one by one (size↕, class)
- A set of m identical machines, each accepting up to k different **classes** (colors) of jobs

Goal:

- Schedule the jobs efficiently (minimize the makespan)

Hurdles:

- Hard lower bounds: can't do better than m times the optimal offline solution



Example schedule:
4 machines (m=4)
11 jobs (n=11)
max 2 colors/machine (k=2)

Learning-augmented algorithms

Predict some of the unknown information where:

- Good predictions lead to a performance close to the offline problem (*consistency*)
- Bad predictions lead to an acceptable performance (*robustness*)

What information **could** we predict ?

- Classical: input, actions (information to predict is $O(n)$)
- Specific: number of classes (and their sizes): information to predict is $O(k)$

What information is **necessary** ?

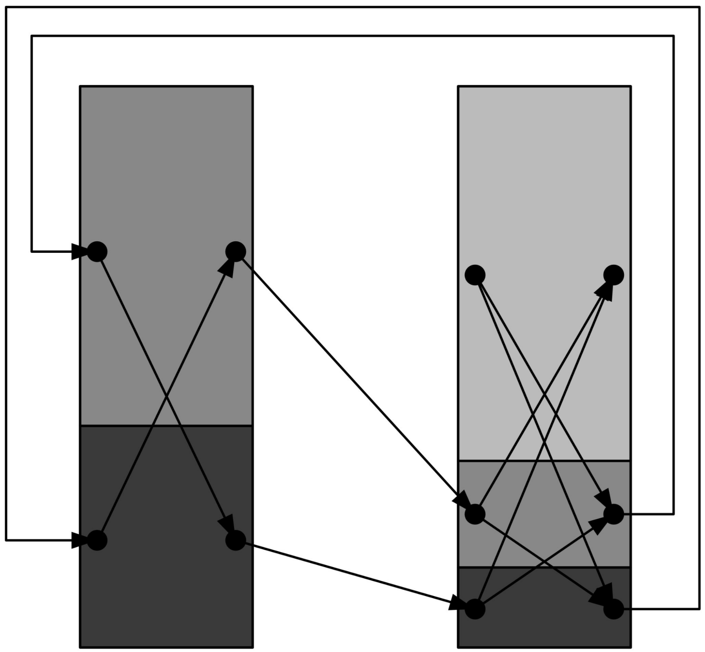
- Number of classes (has to be exact)
- The size of each class (predicted)

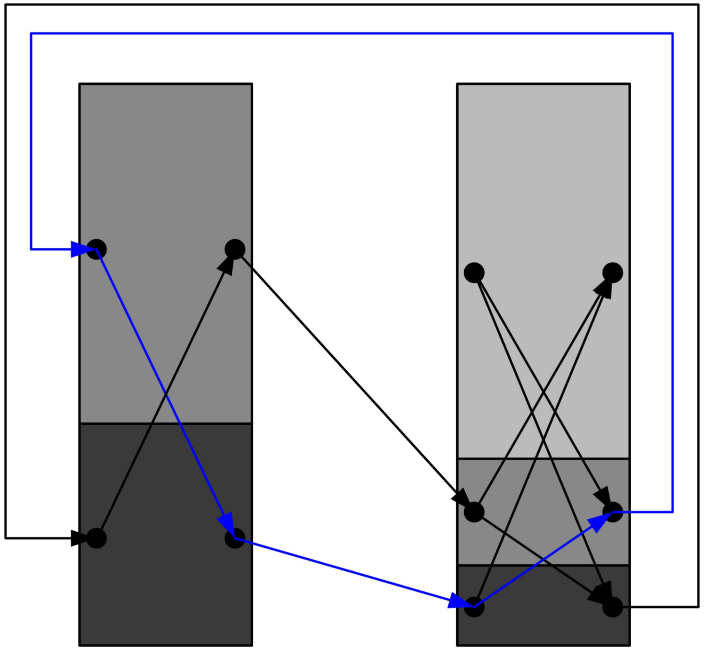
Our approach

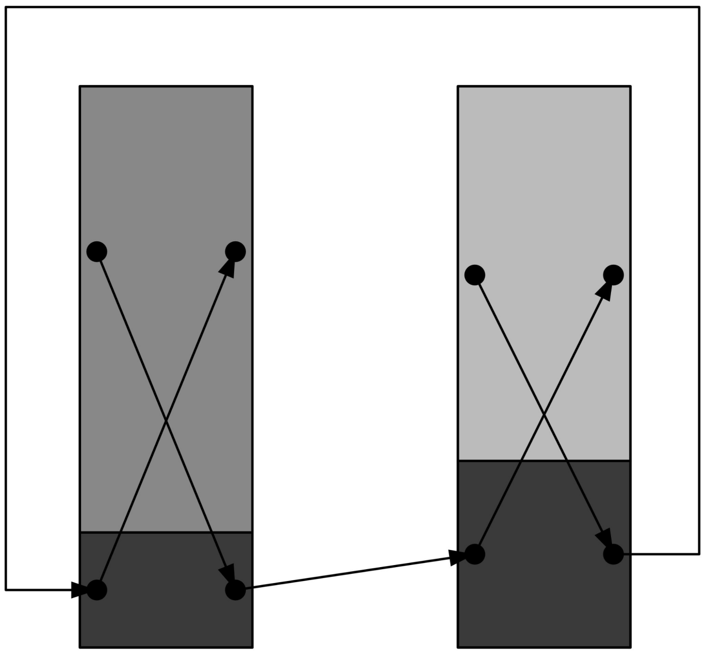
1. Predict the class sizes (with some error L)
2. Compute a schedule based on the predictions
3. Manipulate/structure the schedule to minimize the impact of L
4. Place the incoming jobs according to the resulting schedule

Our approach

1. Predict the class sizes (with some error L)
2. Compute a schedule based on the predictions
3. Manipulate/structure the schedule to minimize the impact of L
4. Place the incoming jobs according to the resulting schedule







Result

Competitive ratio: $2 + \varepsilon + L/OPT$

- Does not depend on m
- Scales gracefully with L

Thank you !