PREEMPTIVE RESOURCE CONSTRAINED SCHEDULING WITH TIME-WINDOWS

Kanthi Sarpatwar

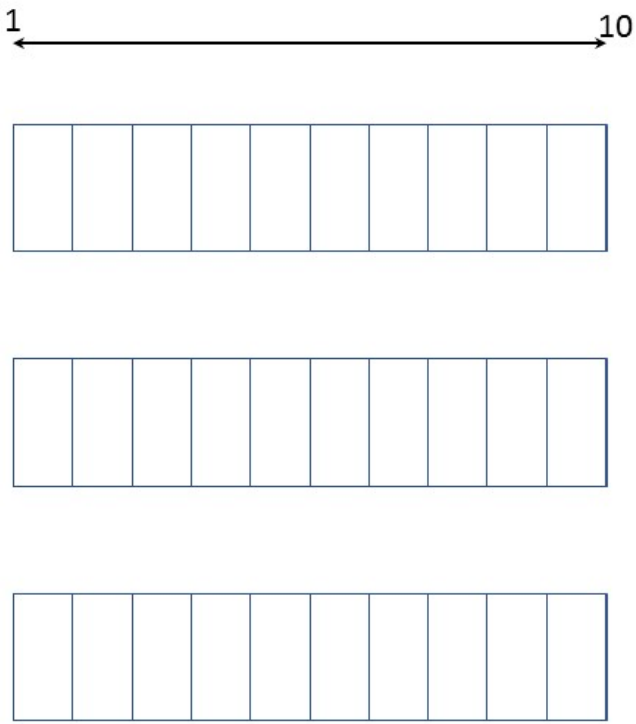
IBM Research

Joint Work With:

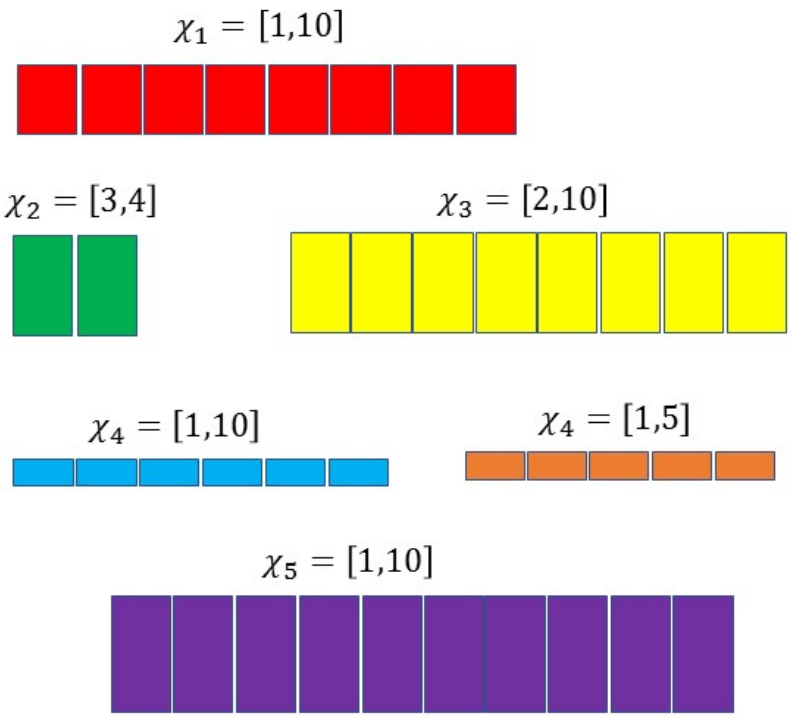
Baruch Schieber (IBM Research)

Hadas Shachnai (Technion)

The General Problem

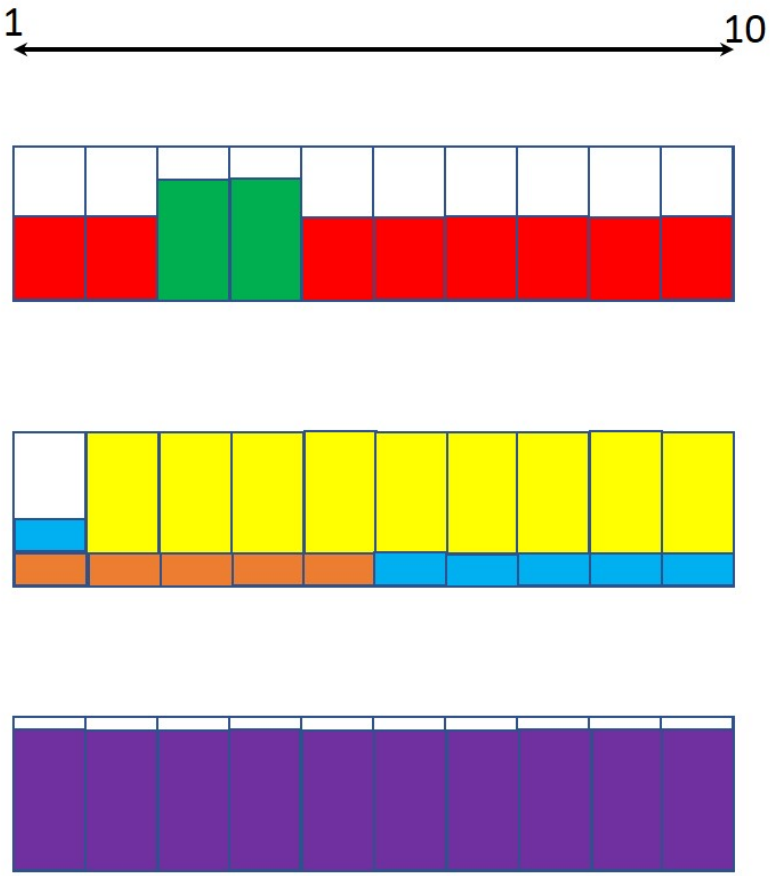


Machines: Each with a unit amount of resource. Time is slotted.



Jobs: Length represents processing time and width represents the resource requirement

The General Problem



Machines: Each with a unit amount of resource. Time is slotted.

- Jobs are non-parallel.
- Preemption and Migration are allowed.
- At any instant, the total resource utilization of jobs scheduled on any machine is at most the capacity.

Formally

Given

- an integral slotted time horizon $[T]$,
- a set of jobs $J = [n]$, a set of machines $M = [m]$ and $d \geq 1$ resources,
- each job j has a requirement vector $\bar{s}_j \in [0,1]^d$, processing time p_j, a release time r_j and deadline d_j (denote $\chi_j = [r_j, d_j]$),
- each machine has a unit capacity of every resource.

Goal and Assumptions

- Schedule (a subset of) jobs onto machines *feasibly*.
- Preemption and Migration are allowed. Jobs are non-parallel i.e., in a given time slot it can run on at most one of the machines.

Variants Considered

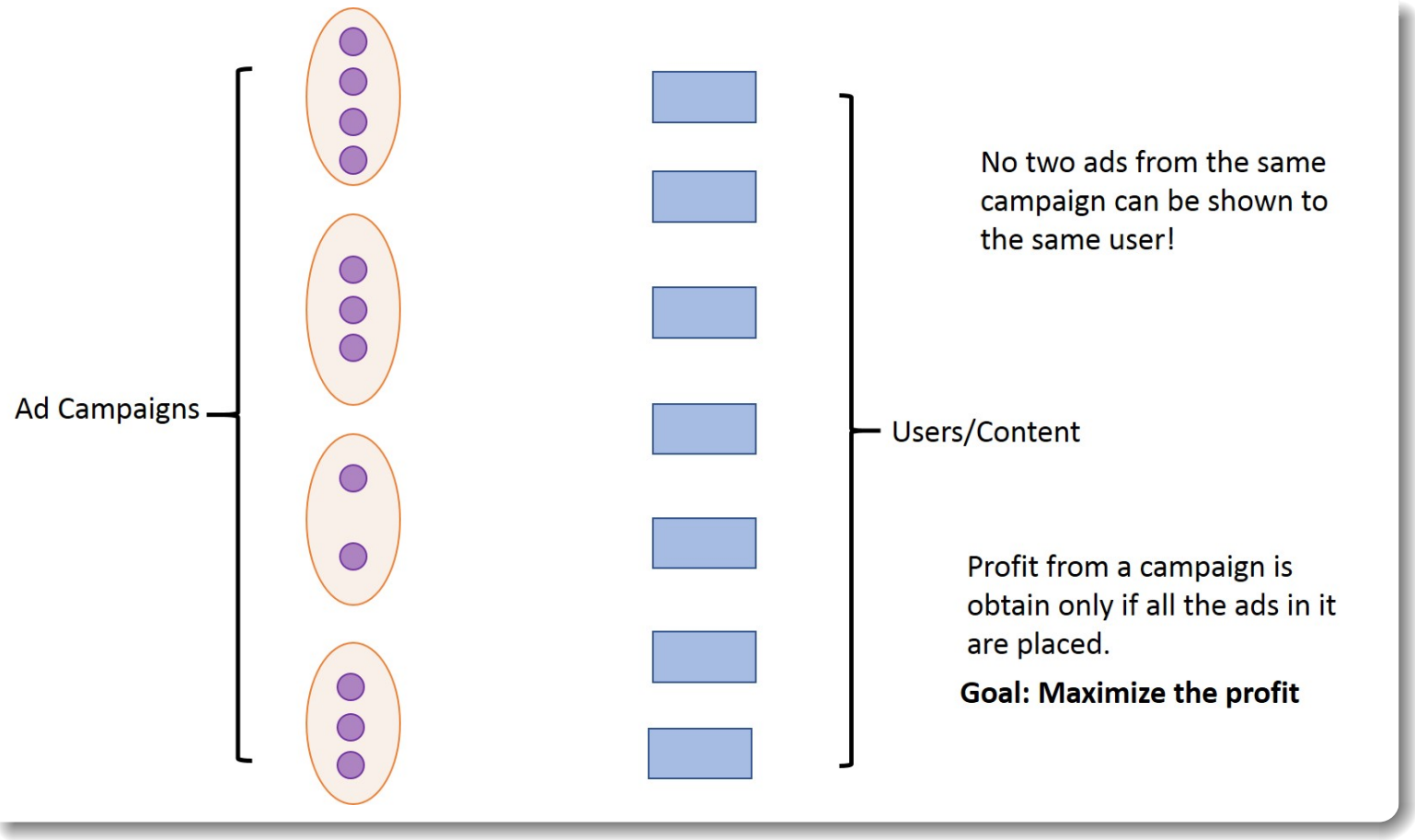
Throughput Maximization (MaxT)

Given a set of jobs J, where each job j is associated with a profit w_j, requirement s_j, processing time p_j and a time window $\chi_j = [r_j, d_j]$. Schedule a subset of jobs S with maximum profit $\sum_{j \in S} w_j$ on a given set m of machines.

Machine Minimization

Given a set of jobs J, where each job j is associated with requirement vector $\bar{s}_j$, processing time p_j and a time window $\chi_j = [r_j, d_j]$. Compute the minimum number of machines needed to scheduled all the jobs successfully.

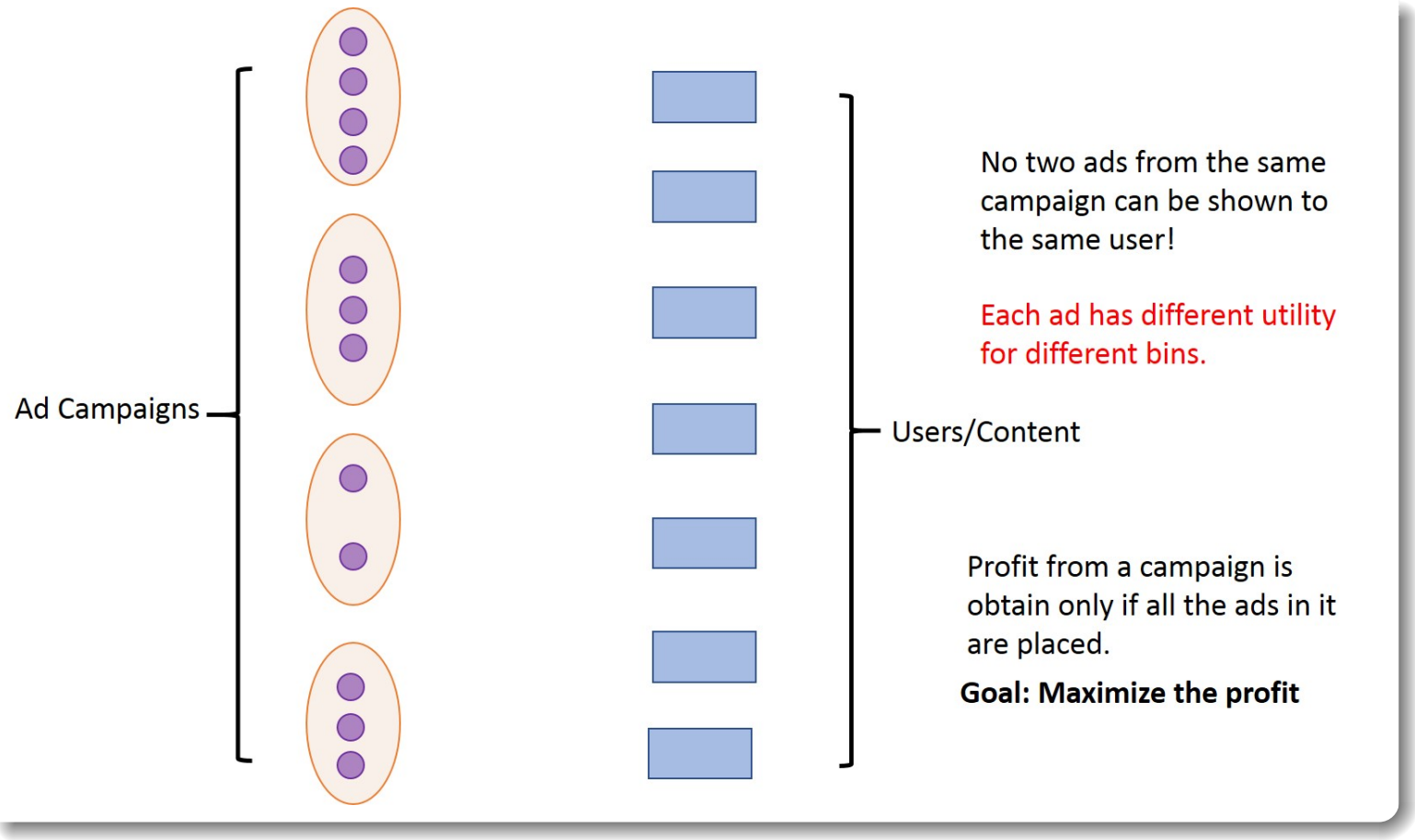
Ad Campaign Scheduling



Freund and Naor (IPCO'02)

Originally obtained a $3+\varepsilon$ approximation guarantee.

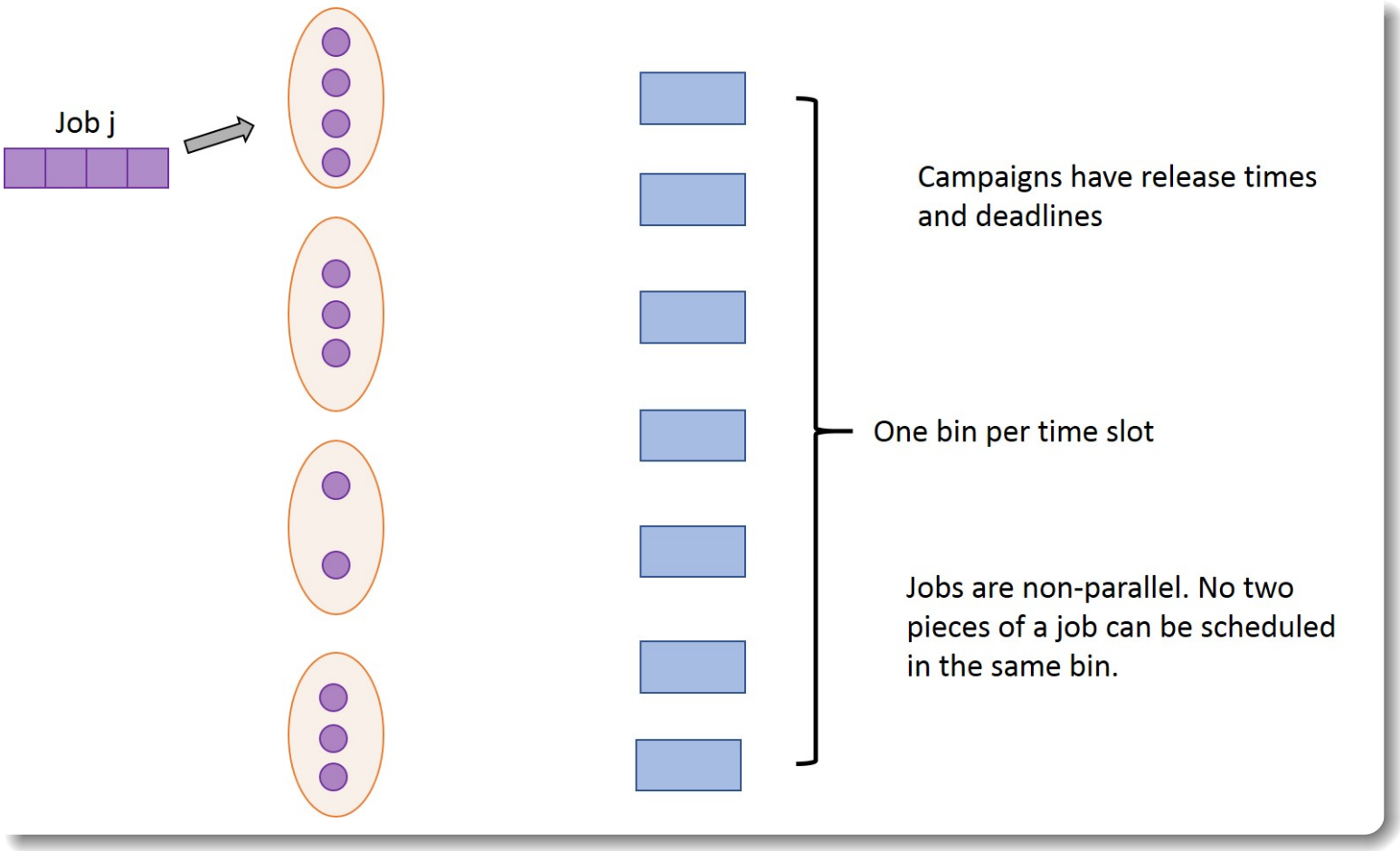
All or Nothing Generalized Assignment (AGAP)



Adany et. al. (IPCO'11)

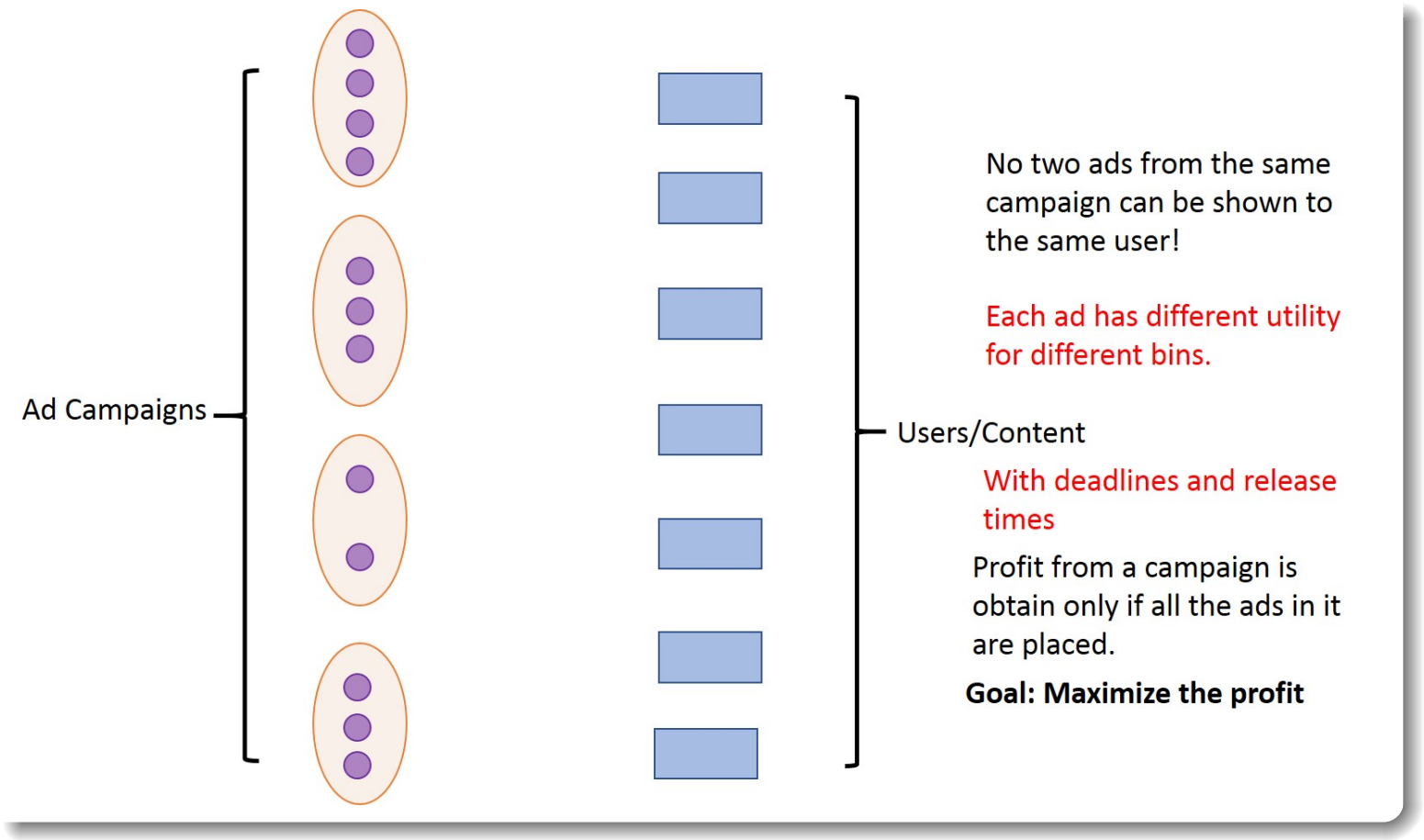
Obtained a constant approximation guarantee.

MaxT : A Generalization of Ad Campaign Scheduling



Each job is a collection of unit sub-tasks. Each machine is a collection of T bins. Release times and deadlines translate directly.

χ-AGAP : A Generalization of AGAP



The Non-Preemptive Version

Resource allocation problem (RAP) is the non-preemptive variant of our throughput maximization problem (MaxT). RAP is a well-studied problem:

- Phillips, Uma and Wein (SODA'00) obtained a $1/6$-approximation algorithm.
- Bar-Noy, Bar-Yehuda, Freund, Naor and Schieber (STOC'00) improved it to $1/3$-approximation guarantee.
- Calinescu, Chakrabarti, Karloff and Rabani (IPCO'02) finally improved it to $1/2 - \varepsilon$ (for any $\varepsilon > 0$).

Machine Minimization

Continuous Model

Jansen and Porkolab (IPCO'02) studied a *continuous* variant of the problem without time-windows where the objective is to minimize the makespan and obtained a *polynomial time approximation scheme* for any constant number of resources $d > 0$ and a single machine.

Vector Packing Problem

In the slotted time model, the machine minimization problem generalizes the vector packing problem.

- Chekuri and Khanna (J. Comp 2005) obtained the first $O(\log d)$ approximation algorithm.
- Bansal, Caprara and Sviridenko (J. Comp 2009) improved this guarantee to $1 + \ln d + \varepsilon$.

Contributions: Throughput variant

Theorem (Laminar MaxT)

For any $\lambda < \frac{1}{3}$, there exists a $(\frac{1-3\lambda}{2})$-approximation algorithm for the laminar MaxT *problem, assuming that $p_j \leq \lambda |\chi_j|$.*

Theorem (Non-Laminar MaxT)

For any $\lambda < \frac{1}{12}$, there exists a $(\frac{1-12\lambda}{8})$-approximation algorithm for the MaxT *problem, assuming that $p_j \leq \lambda |\chi_j|$.*

Theorem (χ-AGAP)

*For any $\lambda < \frac{1}{20}$, there exists a polynomial time algorithm for the χ-*AGAP *problem with an $\Omega(1)$-approximation guarantee. For the case where the time-windows form a laminar family the condition can be relaxed to $\lambda < \frac{1}{5}$.*

Contributions: Machine Minimization

Theorem (MinM)

For any $\varepsilon > 0$ and $\lambda \in (0, \frac{1}{4})$, given an instance of the MinM $(J, \mathscr{W})$ *and sufficiently large constant θ, assuming that $|\chi_j| \geq \theta d^2 \log d \log(T\varepsilon^{-\frac{1}{2}})$ for each $j \in J$, there is a poly-time algorithm that guarantees $O(\log d)$ approximation with probability at least $1 - \varepsilon$.*

The assumption $|\chi_j| \geq \theta d^2 \log d \log(T\varepsilon^{-\frac{1}{2}})$ can be relaxed to $|\chi_j| \geq \theta d^2 \log d \log\log \ldots (\beta \text{ times}) \log(T\varepsilon^{-\frac{1}{2}})$ at a loss of $O(\beta \log d)$ approximation factor.

The General Case: High Level Idea

Step I

Find a maximum weight subset of jobs S, such that, for any interval $\chi \in \mathscr{W}$:

$$\sum_{j \in S : \chi_j \subseteq \chi} s_j p_j \leq \eta m |\chi|$$

Can we show that:

$$\sum_{j \in S} w_j \geq \eta' OPT$$

Step II

Is there a small enough η such that jobs in S can be feasibly scheduled onto the machines.

The Laminar Case

Let $a_j = p_j s_j$ and $\omega \in (0,1)$ be some parameter.

Linear Program

Maximize

$$\sum_{j \in J} w_j x_j$$

Subject to

$$\sum_{j:\chi_j \subseteq \chi} a_j x_j \leq \omega m |\chi| \qquad \forall \chi \in \mathscr{L}$$

$$0 \leq x_j \leq 1 \qquad \forall j \in J$$

Rounding

Assuming $p_j \leq \lambda |\chi_j|$, we construct a rounded solution $\hat{x}_j : j \in J$ and $S = \{j \in J : \hat{x}_j = 1\}$ such that:

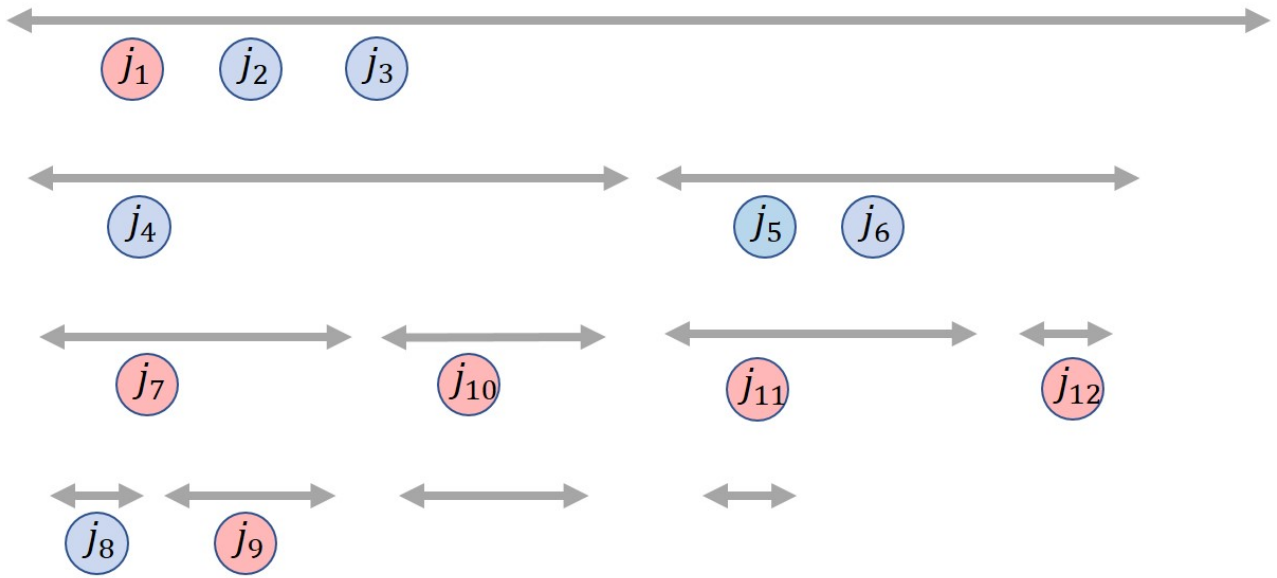
$$\sum_{j \in S} w_j \geq \omega OPT$$

for any $\chi \in \mathscr{L}$,

$$\sum_{j \in S:\chi_j \subseteq \chi} a_j \leq (\omega + \lambda) m |\chi|$$

Rounding Algorithm

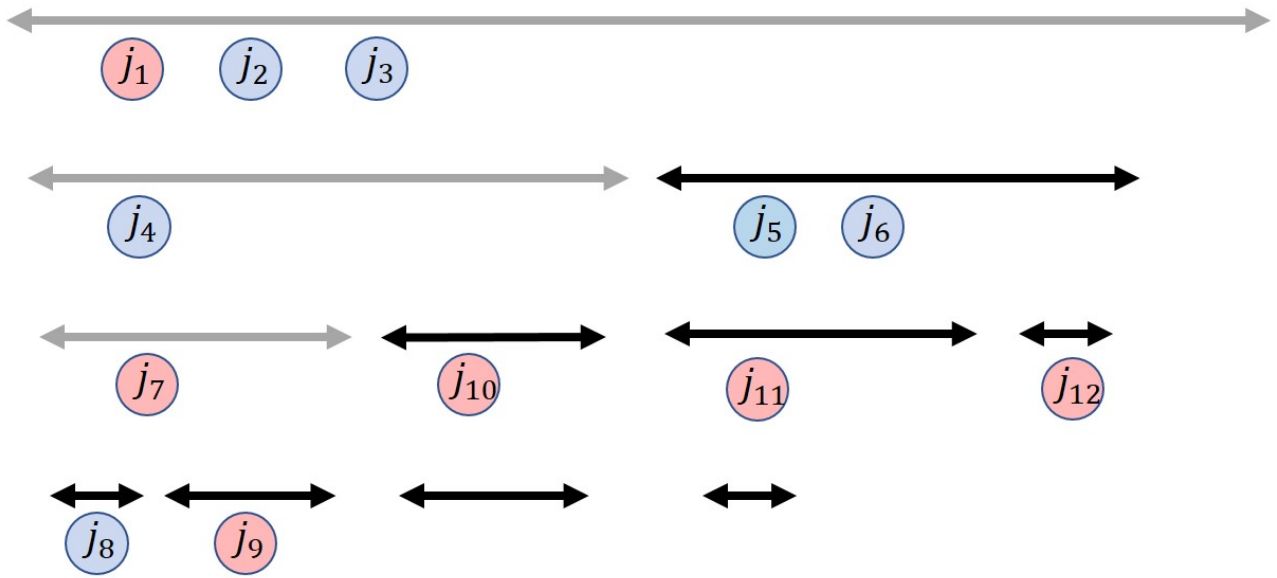
Laminar Windows



- Firstly, any fractional solution, x_j^*: $j \in J$, satisfies $\sum_{j \in J} x_j^* w_j \geq \omega OPT$.
- Jobs shown satisfy $x_j^* > 0$. Red jobs are fractional and blue ones are integral. Initially all the time-windows are colored *gray*.

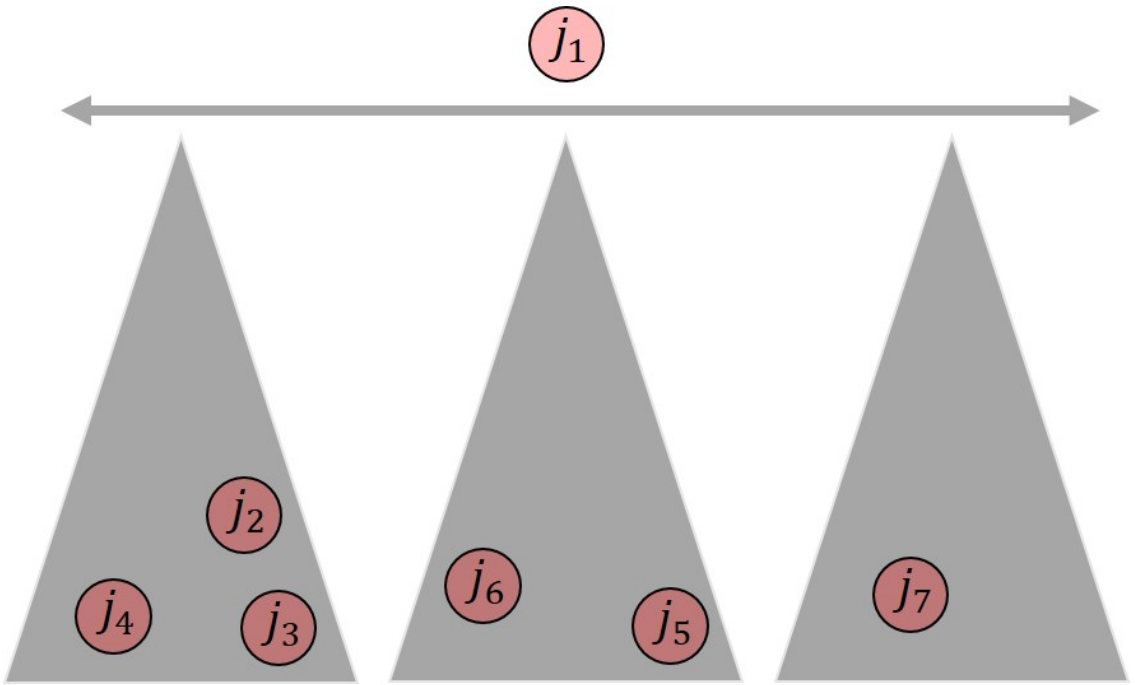
Rounding Algorithm

Laminar Windows



Color a time-window χ *black* if the following property satisfies: for any path $\mathscr{P}(\chi, \chi_l)$ from χ to any leaf χ_l there is at most one fractional job j such that χ_j lies on $\mathscr{P}(\chi, \chi_l)$.

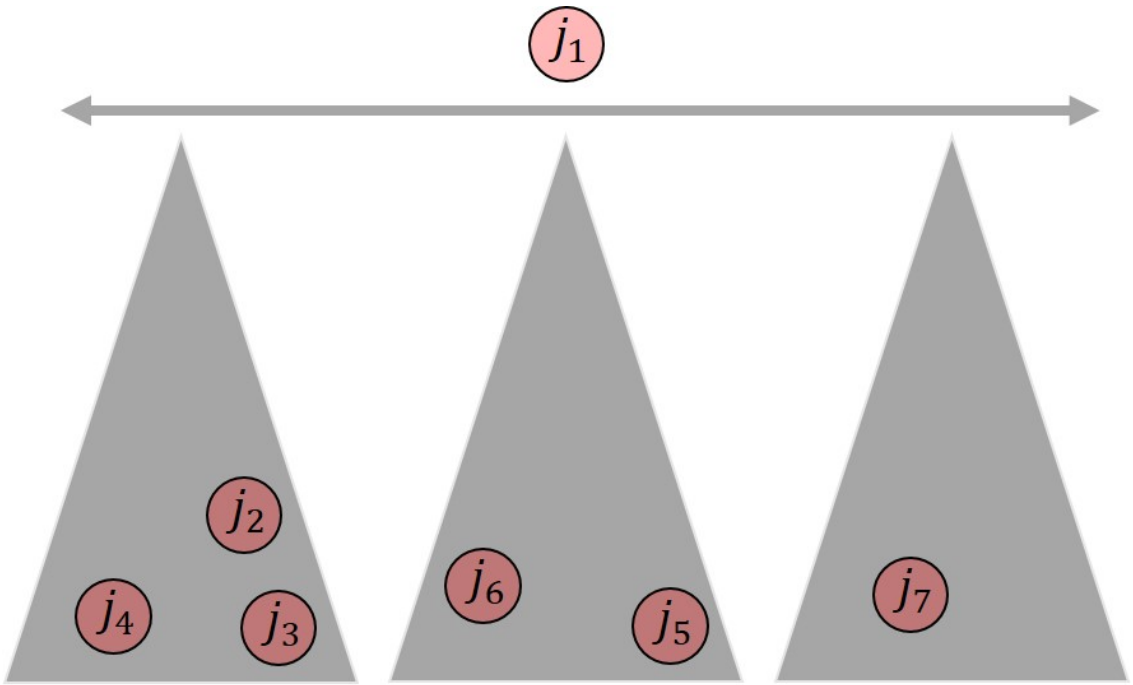
Rounding Algorithm



Pick a minimal gray time-window χ. The following must hold:

- $\exists$ a fractional job j such that $\chi_j = \chi$.
- $\exists$ a non-empty set of fractional jobs $\{j_1, j_2, \ldots, j_l\}$ such that $\chi_{j_i} \subset \chi$.
- $\frac{w_j}{a_j} \leq \frac{w_{j_i}}{a_{j_i}}$ for all $i \in [l]$

Rounding Algorithm



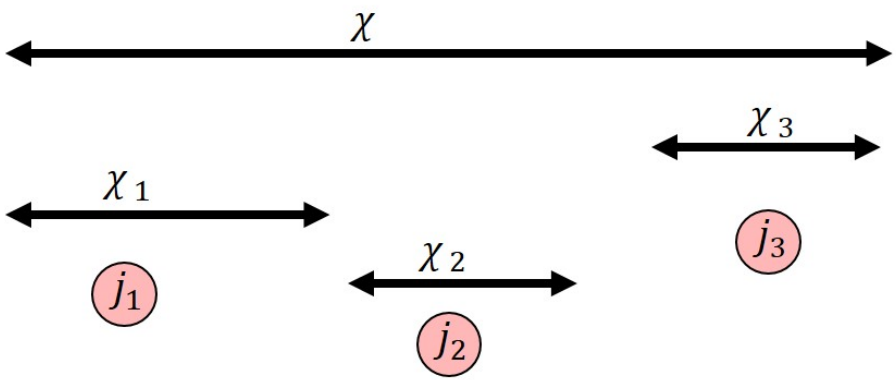
We decrease the fractional value of job j by Δ and increase that of each of the jobs $j_i : i \in [l]$ by Δ_i such that:

- $\Delta a_j = \sum_{i \in [l]} \Delta_i a_{j_i}$ (transferred volume is conserved),
- either $\hat{x}_j = 0$, or $\hat{x}_{j_i} = 1$, for all $i \in [l]$.

Rounding Algorithm

What could go wrong?

- The total volume of jobs packed into a gray interval is conserved!
- What about the black intervals - clearly the volume bound could be violated but by how much?



- We clearly do not create any new fractional jobs.
- Consider the iteration where an interval χ is colored black
- Clearly at the end of this iteration the volume is still conserved.

- The total size increase (ever) in volume is contributed by the fractional jobs in this iteration.
- Increase in volume = $p_{j_1} + p_{j_2} + p_{j_3} \leq \lambda(|\chi_1| + |\chi_2| + \chi_3) \leq \lambda|\chi|$.

Phase II

Summarizing

We can compute a subset S of jobs such that

$$\sum_{j \in S} w_j \geq \omega OPT$$

$$\sum_{j \in S : \chi_j \subseteq \chi} a_j \leq (\omega + \lambda) m |\chi|$$

Next Step

We show that for any $\lambda \in (0, \frac{1}{3})$ and $\omega = \frac{1-3\lambda}{2}$, we can schedule all the jobs in S feasibly. Thus we obtain a constant approximation algorithm for any such λ.

Open Problems

- Throughput variant for d-resources. Is there a $O(\log d)$ approximation?
- Can we obtain a constant approximation for the throughput variant without the slack assumptions?
- For the machine minimization variant can we remove the assumptions on the minimum window size?