

Scheduling with Speed Predictions

Clifford Stein

Eric Balkanski, Tingting Ou and Hao-Ting Wei

Dept. of IEOR

Dept. of CS

Data Science Institute

Columbia University

Motivation for Model (no predictions yet)

- Scheduling with uncertainty is well studied
 - Stochastic
 - Robust
 - Multistage
 - etc.
- Most uncertainty deals with jobs
- What about machines?
- **Application**: group jobs in data center – before knowing where they will be scheduled
- **Application**: pack items into boxes -- before knowing what container/truck they will be placed in

Scheduling with machine uncertainty

[SZ18]

- Input:
 - Upper bound M on the number of machines
 - n jobs, with processing times p_j
- Without knowing the number of machines, **partition** jobs into M **bags**
- Then learn the number of machines m
- **Schedule** the bags on the m machines to minimize makespan (or some other objective)

Scheduling with machine uncertainty

[SZ18]

- Evaluation
 - $\text{Alg}(m)$ - algorithm performance on m machines
 - $\text{OPT}(m)$ – Optimal algorithm that does not need to form bags in the first stage
 - Want to minimize $\text{Max}_m (\text{Alg}(m)/\text{OPT}(m))$
- We called it the **robust ratio**, but we will be using robust to mean something slightly different
- Need to form the bags so as to do well with all possible m

Jobs (1,1,1,1,1,1) $M=3$

Partition 1

1 1 1

m	ALG	OPT	ratio
1	6	6	1
2	4	3	4/3
3	2	2	1

Partition 2

1 1 1

m	ALG	OPT	ratio
1	6	6	1
2	3	3	1
3	3	2	3/2

Jobs (1, 1, 3, 3, 4) $M = 3$

Partition 1

1 1 4
3 3

m	ALG	OPT	ratio
1	12	12	1
2	8	6	$\frac{4}{3}$
3	4	4	1

Partition 2

4 3 3
1 1

m	ALG	OPT	ratio
1	12	12	1
2	7	6	$\frac{7}{6}$
3	5	4	$\frac{5}{4}$

Scheduling with machine uncertainty

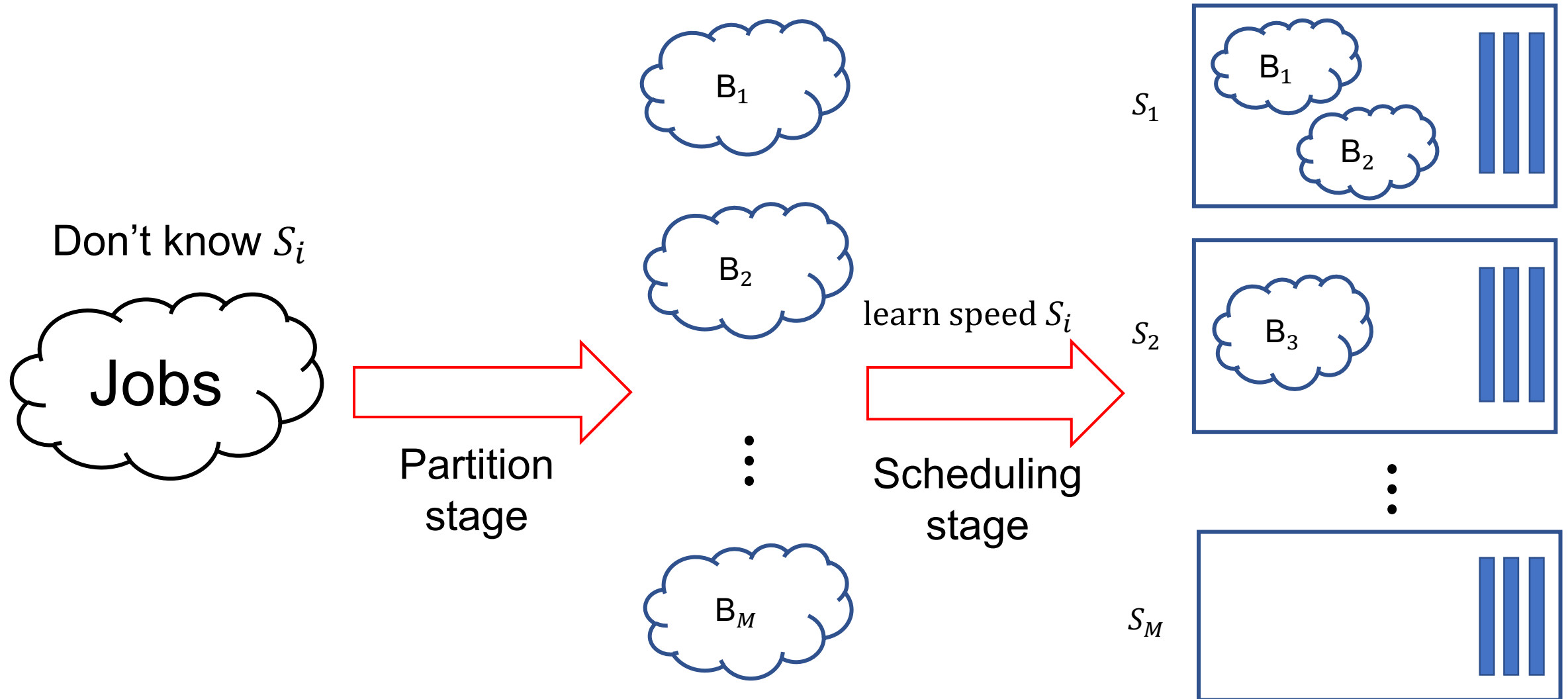
[SZ18]

- $5/3$ approximation in general
- Better for special cases, unit jobs and infinitely divisible jobs
- Consider other objectives too

Generalization to speeds [EHMNSS21]

- Input
 - m machines, with unknown speeds
 - n jobs, with processing times p_j
- Without knowing the speeds of machines, **partition** the jobs into m **bags**
- Then learn the machine speeds s_i
- **Schedule** the bags on the m machines to minimize makespan (or some other objective)

Two level scheduling



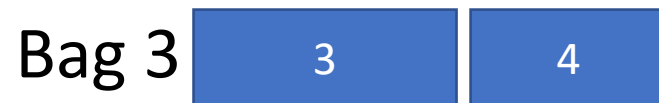
Results from Eberle et. al.

	General speeds		Speeds from $\{0, 1\}$	
	Lower bound	Upper bound	Lower bound	Upper bound
Discrete jobs (Rocks)	$\bar{\rho}(m)$ (Theorem 2.1)	$2 - \frac{1}{m}$ (Theorem 3.3)	$\frac{4}{3}$ [18]	$\frac{5}{3}$ [18]
Equal-size jobs (Bricks)	$\bar{\rho}(m)$ (Theorem 2.1)	1.8 (Theorem 4.3)	$\frac{4}{3}$ ([18], Theorem 4.6)	
Infinitesimal jobs (Sand)	$\bar{\rho}(m) \leq \frac{e}{e-1} \approx 1.58$ (Theorems 2.1 and 2.3)		$\bar{\rho}_{01}(m) \leq \frac{1+\sqrt{2}}{2} \approx 1.207$ ([18], Theorem 2.4)	

Table 1: Summary of results on speed-robust scheduling.

Discrete jobs – 2 robust

- Don't know speeds, but assume for now that $\sum s_i = \sum p_j$
- Put jobs in bags via LPT (Longest processing time)



Discrete jobs – 2 robust

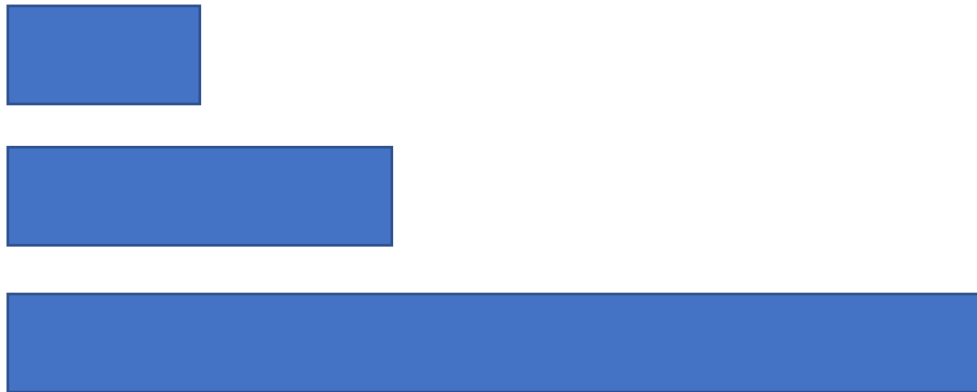
- Don't know speeds, but assume for now that $\sum s_i = \sum p_j$
- Put jobs in bags via LPT (Longest processing time)



- Large (1 per bag) jobs are annoying
- **b** is the size of the largest bag with at least 2 jobs
- Ignoring Large jobs, smallest and largest bag are within a factor of **2**

Learn machine speeds

- Say machine speeds are $1, 2, 5$
- Can think of machine of speed s as a machine of speed 1 , but with capacity s



- Will actually think of machines as having size $2s_i$, and will pack bags on machines using LPT

Everything fits

- Suppose the first k bags fit
- Suppose bag $k+1$ of size T does not fit
- Let smallest bag be of size w , $w \geq \frac{T}{2}$
- The $(m-k)$ unpacked bags must have size at least

$$U \geq (m - k - 1)w + T \geq (m - k - 1) \left(\frac{T}{2} \right) + T$$

- The k packed bags must have size

$$P \geq kT$$

Everything fits (cont)

- Since we inflated the machines by a factor of **2** and everything fits in aggregate, we must have at least **$2U + P$** free space.
- Plugging in

$$2U + P$$

$$\geq 2\left((m - k - 1)w + \left(\frac{T}{2}\right) + T\right) + kT$$

$$\geq (m - k - 1)T + 2T + kT$$

$$\geq (m + 1)T$$

- So at least one machine can fit the bag of size **T** .

New Topic -- Algorithms with Predictions

- General idea of getting away from worst case analysis
- For an online algorithm, you have some predictions about the data
- If the predictions are accurate, you'd like to do as well as the offline algorithm
- If the predictions are not accurate, you'd like to do as well as the online algorithm
- You don't know if the predictions are accurate
- You need one algorithm for both cases
- You can also consider incorporating measures of the prediction accuracy

(Aside) What do we want from Predictions

- Guiding principals
 - Computable based on prior job traces
 - Predictions should be reasonably sized
 - Should be robust to error or inconsequential changes to the input
- Focus on quantity to predict
 - Independent of learning algorithm used to construct the prediction
 - Focus on the worst case with access to the prediction

Some Early Papers


- Caching [Lykouris and Vassilvitskii 2018]
- Ski Rental [Purohit et al 2018]
- Non-clairvoyant scheduling [Purohit et al 2018]

Algorithms with predictions are a hot area

- Workshop at STOC
- Previous talk in this workshop by Ben Moseley, just on scheduling algorithms with predictions
- Webpage [Lindermayr, Megow]

Algorithms-with-predictions.github.io

Algorithms with Predictions [PAPER LIST](#) [FURTHER MATERIAL](#) [ABOUT](#)

 Newest first ▾ 117 papers

Private Algorithms with Private Predictions	Amin, Dick, Khodak, Vassilvitski	arXiv '22	differential privacy
Paging with Succinct Predictions	Antoniadis, Boyar, Eliás, Favrholdt, Hoeksma, Larsen, Polak, Simon	arXiv '22	caching/paging online
Proportionally Fair Online Allocation of Public Goods with Predictions	Banerjee, Gkatzelis, Hossain, Jin, Micha, Shah	arXiv '22	allocation online
Canadian Traveller Problem with Predictions	Bampis, Escoffier, Xefteris	arXiv '22	online routing
Learning-Augmented Algorithms for Online Linear and Semidefinite Programming	Grigorescu, Lin, Silwal, Song, Zhou	arXiv '22	covering problems online SDP
Strategyproof Scheduling with Predictions	Balkanski, Gkatzelis, Tan	arXiv '22	AGT scheduling
Learning-Augmented Maximum Flow	Polak, Zub	arXiv '22	max flow running time
Online Prediction in Sub-linear Space	Peng, Zhang	arXiv '22	learning online
Learning-Augmented Query Policies for Minimum Spanning Tree with Uncertainty	Erlebach, Lima, Megow, Schlöter	arXiv '22	ESA '22 explorable uncertainty network design online
Online TSP with Predictions	Hu, Wei, Li, Chung, Liao	arXiv '22	online routing
Learning-Augmented Binary Search Trees	Lin, Luo, Woodruff	arXiv '22	ICML '22 data structure search
Chasing Convex Bodies and Functions with Black-Box Advice	Christianson, Handina, Wierman	arXiv '22	COLT '22 convex body chasing online
Online Bipartite Matching with Advice: Tight Robustness-Consistency Tradeoffs for the Two-Stage Model	Jin, Ma	arXiv '22	matching online
Learning-Augmented Algorithms for Online TSP on the Line	Gouleakis, Lakis, Shahkarami	arXiv '22	online routing
A Universal Error Measure for Input Predictions Applied to Online Graph Problems	Bernardini, Lindermayr, Marchetti-Spaccamela, Megow, Stougie, Sweering	arXiv '22	network design online routing
Mechanism Design with Predictions	Xu, Lu	arXiv '22	IJCAI '22 AGT auctions scheduling
Discrete-Convex-Analysis-Based Framework for Warm-Starting Algorithms with Predictions	Sakaue, Oki	arXiv '22	matching matroid intersection running time
A Regression Approach to Learning-Augmented Online Algorithms	Anand, Ge, Kumar, Panigrahi	arXiv '22	NeurIPS '21 learning online
Customizing ML Predictions For Online Algorithms	Anand, Ge, Panigrahi	arXiv '22	ICML '20 learning online rent-or-buy
Improved Price of Anarchy via Predictions	Gkatzelis, Kollias, Sgouritsa, Tan	arXiv '22	EC '22 AGT
Online Algorithms with Multiple Predictions	Anand, Ge, Kumar, Panigrahi	arXiv '22	ICML '22 cover problems multiple predictions online
Scheduling with Speed Predictions	Balkanski, Du, Stein, Wei	arXiv '22	online scheduling

- data structure
- online
- running time
- AGT
- differential privacy
- prior/related work
- allocation
- auctions
- beyond NP hardness
- bidding
- caching/paging
- clustering
- convex body chasing
- cover problems
- covering problems
- data-driven
- experiments
- explorable uncertainty
- exploration
- k-server
- learning
- linear quadratic control
- load balancing
- locality sensitive hashing
- matching
- matroid intersection
- max flow

Scheduling with speed predictions

- Input:
 - n jobs with processing time p_j
 - m machines with unknown speed $s_i > 0$
 - A prediction of machine speeds $\hat{s}_i > 0$
- Reminder:
 - The processing time for job j on machine i is p_j/s_i
- Model
 - **Partition** the jobs into m (possibly empty) bags only knowing speed predictions \hat{s}_i
 - Learn the speeds \hat{s}_i . Schedule bags on machines
 - Objective function: Minimize the makespan

Evaluation of Algorithms

- For any speed realization S ,
 - ALG_S is makespan of our algorithm
 - OPT_S is optimal makespan that knows S in advance (no bags)
- Define
 - $\alpha = \frac{ALG_{S=\hat{S}}}{OPT_{S=\hat{S}}}$ is the consistency (performance with good predictions)
 - $\beta = \max_S \frac{ALG_S}{OPT_S}$ is the robustness (performance with bad predictions)

Our results

Job sizes	Speeds	Lower bound	Upper bound
General	General	$1 + (1 - \alpha)/2\alpha - O(1/m)$ (Theorem 3.2)	$2 + 2/\alpha$ (Theorem 4.9)
Equal-size	General	$1 + (1 - \alpha)/2\alpha - O(1/m)$ (Theorem 3.2)	$2 + 1/\alpha$ (Theorem 5.1)
Infinitesimal	General	$1 + (1 - \alpha)^2/4\alpha - O(1/m)$ (Theorem 3.2)	$1 + 1/\alpha$ (Theorem 5.5)
General	$\{0,1\}$	$(4 - 2\alpha)/3$ (Theorem 5.13)	2 (Theorem 5.11)

Table 1: Robustness guarantees of deterministic $(1 + \alpha)$ -consistent algorithms, for any $\alpha \in (0, 1/2)$.

Consider prediction error

- We define the prediction error $\eta = \max_{i \in [m]} \frac{\max\{\hat{s}_i, s_i\}}{\min\{\hat{s}_i, s_i\}}$ to be the maximum ratio between the true speed and the predicted speed, or vice versa.
- We propose an algorithm that achieves the following result:

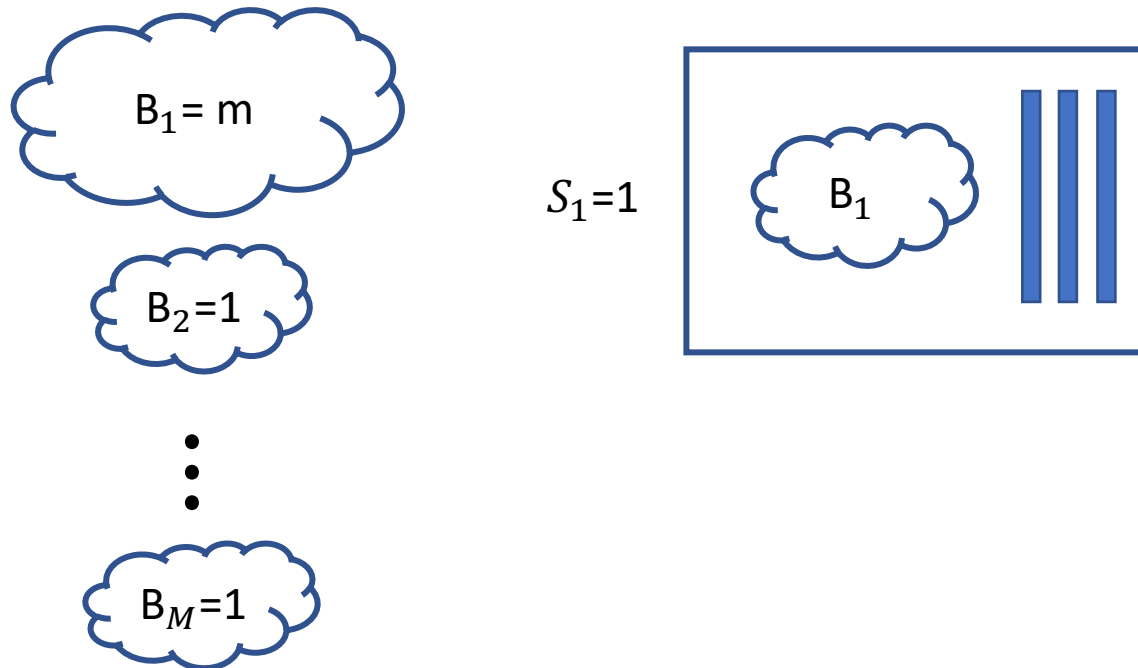
$\min\{\eta^2 (1+\alpha), (2 + 2/\alpha)\}$ -approximation

for any constants $\alpha \in (0,1)$, where $\eta \geq 1$ is the prediction error.

- $(1+\alpha)$ -consistent , $(2 + 2/\alpha)$ -robust

Cannot trust the predictions blindly

- Consider $2m-1$ equal size jobs
 - $p_1 = p_2 = \dots = p_{2m-1} = 1$.
- Given the prediction where $\hat{s}_1 = m, \hat{s}_2 = \dots = \hat{s}_m = 1$.

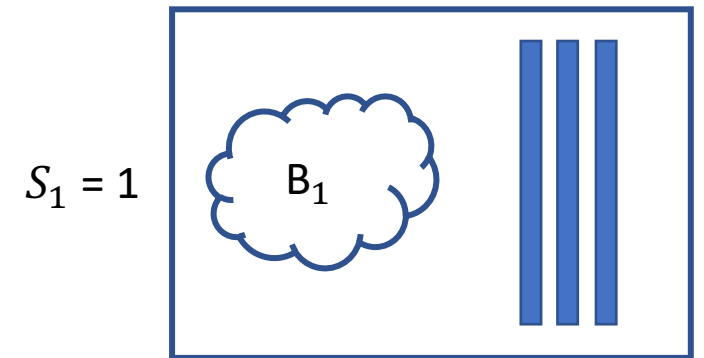
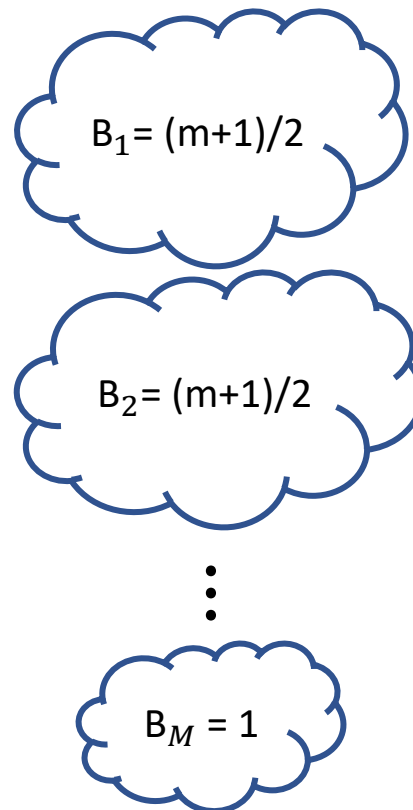


If $s_1 = s_2 = \dots = s_m = 1$ then the optimal makespan is 2 but trust the prediction has makespan at least m .

Trade-off between consistency and robustness

Theorem: Any $(1+\alpha)$ -consistent algorithm has $\sim 1/\alpha$ rate of increase of the robustness.

E.g. If the required consistency is $(m+1)/m$ then it is feasible to create two bags with total processing time $(m+1)/2$.



Makespan decreases to $(m+1)/2$

Trade-off between consistency and robustness

Theorem: Any $(1+\alpha)$ -consistent algorithm has $\sim 1/\alpha$ rate of increase of the robustness.

- 2 consistent algorithm can be $O(1)$ -robust
- Partition equally
 - 2 consistent (makespan = 2, vs. 1)
 - Can show is $O(1)$ robust
- Want to be between follow predictions and partition equally

Preliminaries to Algorithm

- There is a PTAS [HS 88] for scheduling with speeds to minimize makespan, so we will focus on the partitioning stage.
- Conceptually, we'd like to think about equal-sized jobs.
- Single large jobs get in the way of that thinking
- They can basically be handled by observing that both the algorithm and OPT has to put them alone (similar to earlier proof)

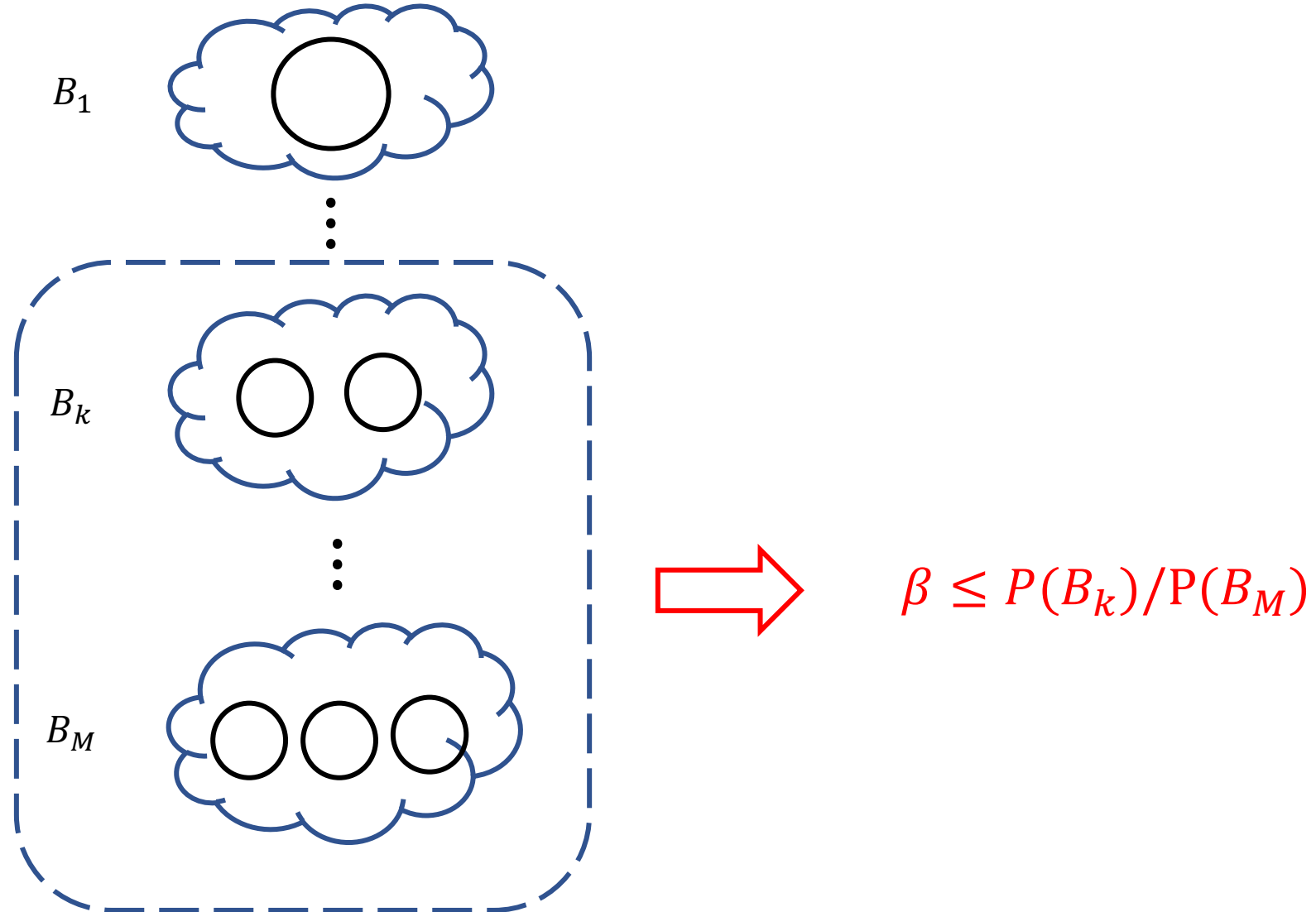
Useful fact about robustness

- For any partition $\mathcal{B} = \{B_1, \dots, B_M\}$,
 - let $p(B)$ be the total processing time of a bag $B \in \mathcal{B}$.
 - Let $|B|$ be the number of jobs in a bag $B \in \mathcal{B}$
- Theorem : The robustness of partition \mathcal{B} is at most $\max\{\beta, 2\}$, where

$$\beta = \frac{\max_{B \in \mathcal{B}, |B| \geq 2} P(B)}{\min_{B \in \mathcal{B}} P(B)},$$

is ratio the maximum total processing time of a bag containing at least two jobs to the minimum total processing time of a bag.

Bound the robustness



Algorithm: Iterative-Partial-Rebalancing (IPR)

Partitioning Step:

1. Trust the prediction \hat{S} and partition all of jobs into m bags ($\sim m$ machines)
2. “Remove” singletons
3. While $(\beta \geq 4)$
 1. Add a bag with the minimum total processing time to the “machine” that contains the largest non-singleton bag.
 2. Rebalance this machine into 2 bags using LPT
4. Output the m bags

Algorithm: Iterative-Partial-Rebalancing (IPR)

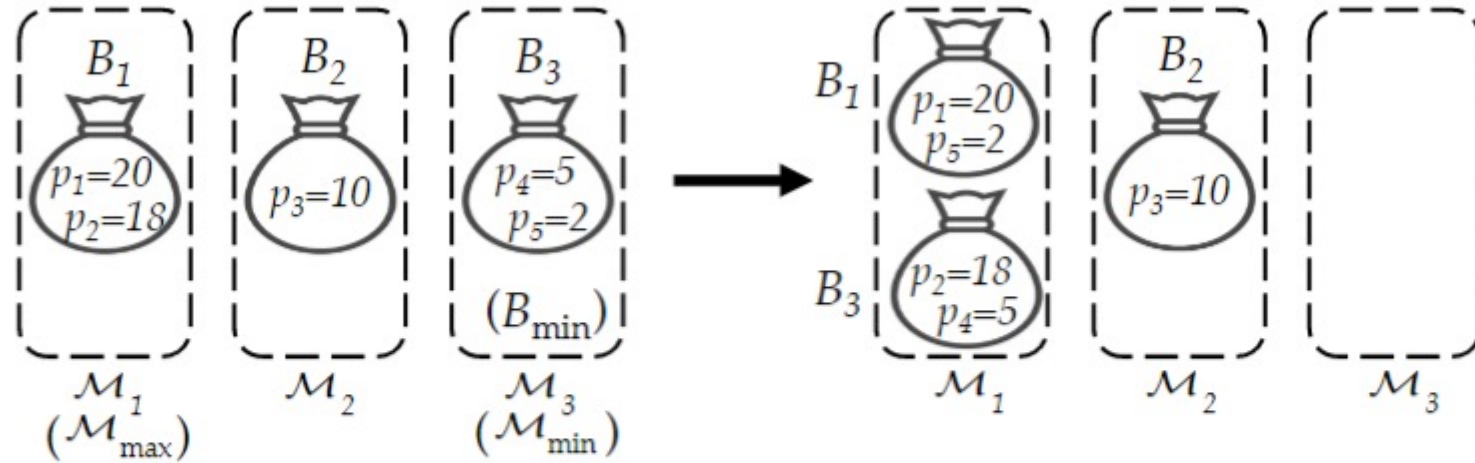


Figure 1: Illustration of one iteration of the IPR algorithm on an example with $m = 3$ bags and machines and $n = 5$ jobs.

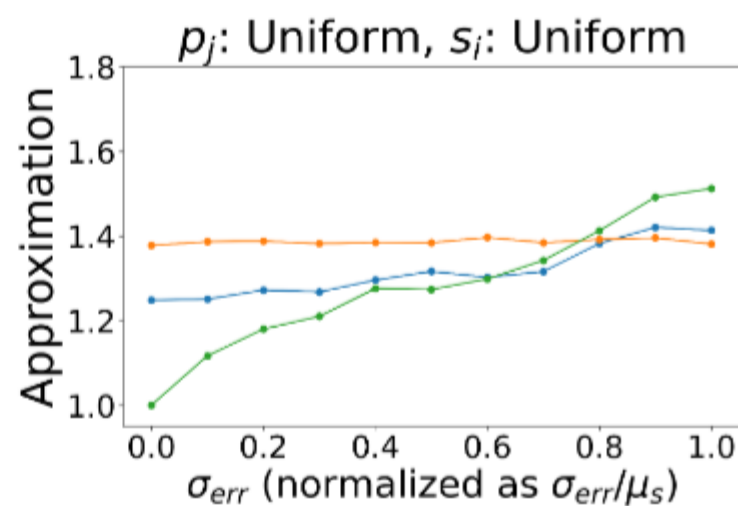
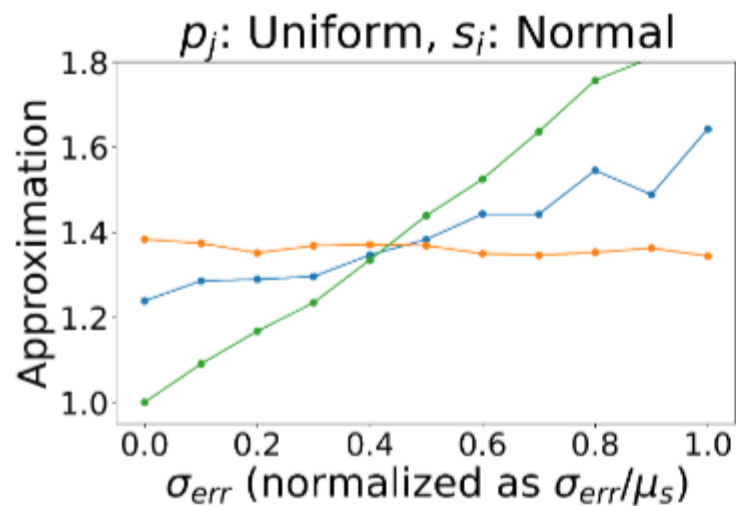
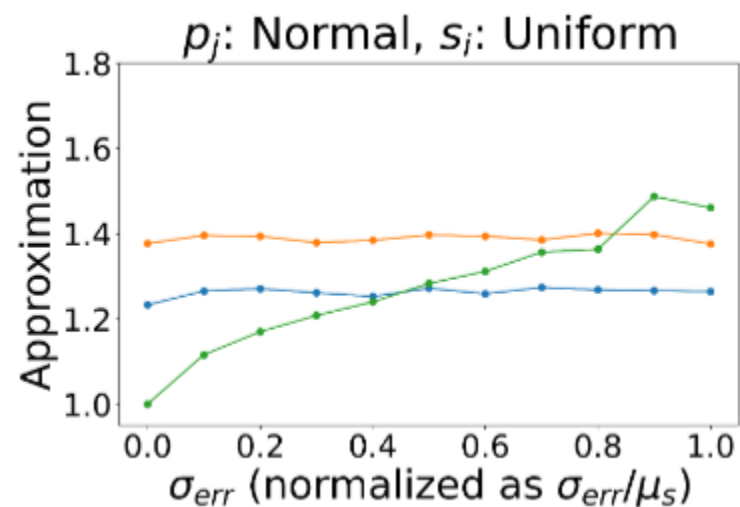
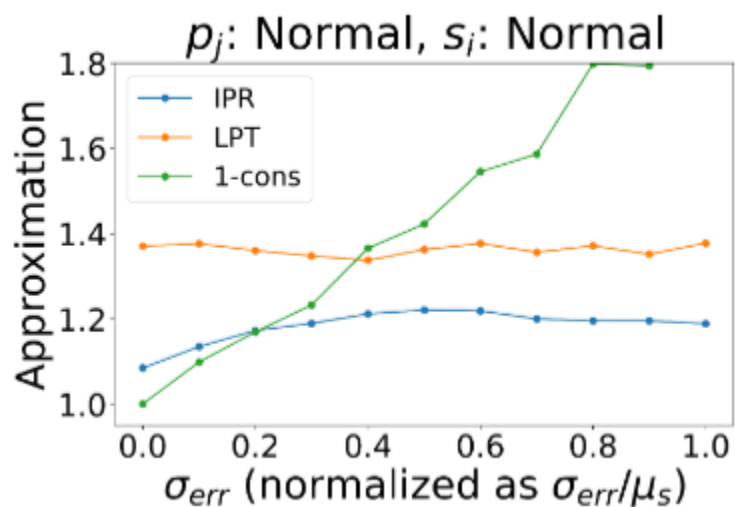
Analysis Sketch

- Lemma: During the algorithm, the processing time of the bag with minimum total processing cannot decrease.
- Lemma: β does decrease, using features of LPT (Similar to analysis in [EHMNSS21])
- So we gradually increase consistency and decrease robustness
- until...
- we show that if we require our partition to be $(1 + \alpha)$ -consistent then the robustness is at most $(2 + 2 / \alpha)$.

Consider Prediction Error

- Recall the prediction error $\eta = \max_{i \in [m]} \frac{\max\{\hat{s}_i, s_i\}}{\min\{\hat{s}_i, s_i\}}$
- Theorem: the impact of error is at most η^2 .
- $\eta^2(1+\alpha)$ -consistent , $(2 + 2/\alpha)$ -robust

Experimental Results



Conclusions

- Incorporate predictions into problem with machine uncertainty
- Other results for special cases
- Algorithm starts with predictions and moves to robustify (in partitioning phase)
- Possible paradigm for other algorithms with predictions