CONSISTENCY–ROBUSTNESS TRADEOFFS FOR STRATEGYPROOF SCHEDULING WITH PREDICTIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

We study strategyproof scheduling on n unrelated machines with predictions. Each machine is controlled by an agent with privately known processing times, while the mechanism receives a public prediction of the processing-time matrix before the agents report. The objective is to minimize the makespan subject to strategyproofness. We measure performance by consistency, the approximation guarantee for correct predictions, and robustness, the worst-case guarantee for arbitrary predictions. We introduce EDGESKIP, a deterministic strategyproof member of the class of job-wise weighted mechanisms. Such mechanisms allocate each job independently using prediction-dependent weights. Using a polynomial-time 2-approximate reference schedule computed from the prediction and a standard tradeoff parameter, EDGESKIP is 4-consistent and $(2n-2)$-robust, improving on the $(6, 2n)$ guarantee of Balkanski, Gkatzelis, and Tan. Without computational restrictions, EDGESKIP with an optimal reference schedule is C-consistent and $\max\{n, (n-1)C/(C-1)\}$-robust for every $C > 1$. We prove a matching information-theoretic lower bound for all job-wise weighted mechanisms, thereby determining the exact consistency–robustness tradeoff for this class. For arbitrary deterministic strategyproof mechanisms, we establish the robustness lower bound $\max\{n, C/(C-1)\}$ for every $C > 1$. We also study an error-tolerant variant and improve upon prior guarantees. Experiments demonstrate that our mechanisms achieve good empirical performance.

1 INTRODUCTION

Scheduling on unrelated machines is a fundamental problem in approximation algorithms and algorithmic mechanism design (Lenstra et al., 1990; Nisan & Ronen, 1999). The goal is to assign m jobs to n machines to minimize the makespan, the maximum total processing time on any machine. With public processing times, the problem admits a polynomial-time 2-approximation (Lenstra et al., 1990).

In the strategic version, each machine is controlled by an agent whose processing times are private. An agent may misreport these times to obtain a more favorable assignment or payment. A mechanism chooses an assignment and payments; each agent's utility is its payment minus its total true processing time. Strategyproofness requires truthful reporting to maximize this utility regardless of the other agents' reports. Nisan & Ronen (1999) gave a deterministic strategyproof n-approximation and conjectured that this ratio is optimal even without computational restrictions. Christodoulou et al. (2023) proved the conjecture. The gap between n and 2 motivates using additional information to improve approximation while maintaining strategyproofness.

Learning-Augmented Strategyproof Scheduling. Learning-augmented algorithms use predictions to improve performance while retaining worst-case guarantees (Purohit et al., 2018; Lykouris & Vassilvitskii, 2021; Mitzenmacher & Vassilvitskii, 2022), an approach also studied in strategic settings (Agrawal et al., 2022; Xu & Lu, 2022). In scheduling, the mechanism receives an external public prediction $\widehat{\mathbf{p}}$ of the processing-time matrix $\mathbf{p}$, not controlled by the agents, before they report their types (Balkanski et al., 2023). Its *consistency* is the approximation ratio with correct predictions, and its *robustness* is the worst-case ratio with arbitrary predictions. We seek the best tradeoff subject to strategyproofness for every fixed prediction. Fixing a predicted optimal schedule with

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

report-independent payments is strategyproof and perfectly consistent, but has unbounded robustness. Ignoring predictions and using the Nisan–Ronen mechanism instead gives both consistency and robustness n.

Balkanski et al. (2023) introduced SCALEDGREEDY, the first strategyproof mechanism with constant consistency and $O(n)$ robustness for this setting. It greedily reassigns jobs from a predicted reference schedule in decreasing order of speedup, closing each machine when its incoming predicted load reaches a threshold. The resulting target schedule determines the weights used to allocate jobs after the agents report their processing times. With a κ-approximate reference schedule and parameter $\gamma > 0$, it is $(2+\gamma)\kappa$-consistent and $n(1+1/\gamma)$-robust, giving $(6, 2n)$ for the polynomial-time choices $\kappa = 2$ and $\gamma = 1$. Their variant ERRORTOLERANTSCALEDGREEDY uses a tolerance parameter to bias the allocation toward this target schedule and provide an error-dependent guarantee.

Both mechanisms belong to a natural class that we call *job-wise weighted* mechanisms. The prediction fixes positive machine–job weights, and each job is independently assigned to a machine minimizing its weighted reported processing time; report-independent weights ensure strategyproof payments exist. Can we improve the polynomial-time guarantees and determine the optimal tradeoff within this class? What lower bounds hold for arbitrary deterministic strategyproof mechanisms, without computational or structural restrictions?

1.1 OUR CONTRIBUTIONS

We improve the polynomial-time guarantees, characterize the weighted frontier without computational restrictions, and make progress on the second question. As usual, we describe allocation rules, while the strategyproof payments are guaranteed by their incentive properties and are made explicit in the preliminaries.

- **Improved upper bounds (Section 3).** We introduce EDGESKIP, a prediction-dependent job-wise weighted mechanism. Starting from a κ-approximate reference schedule computed from the prediction, it constructs a target schedule by considering predicted speedups while skipping every proposed reassignment that would violate an incoming-load capacity. For every $C > \kappa$, EDGESKIP can be parameterized to be strategyproof and C-consistent. Its robustness guarantee, denoted by $R_\kappa(C)$, is the piecewise function stated in Theorem 1. The robustness analysis reduces the load on each output machine to the weight of a partial matching and uses a dependency forest to charge rejected reassignments without double counting. The general guarantee has two notable consequences.

  (1) With a polynomial-time 2-approximate reference schedule, EDGESKIP can be parameterized to be 4-consistent and $(2n-2)$-robust, improving the $(6, 2n)$ consistency–robustness guarantee of SCALEDGREEDY (Balkanski et al., 2023) and the $(4, 2.5n)$ guarantee of Cole et al. (2025).

  (2) With an optimal reference schedule, EDGESKIP is C-consistent with robustness at most $\max\left\{n, \frac{(n-1)C}{C-1}\right\}$ for every $C > 1$.

  We also introduce the error-tolerant variant TOLERANTEDGESKIP. For every $C > \kappa$ and tolerance $\bar{\eta} \geq 1$, its approximation ratio is at most $C\eta^2$ whenever the multiplicative prediction error η is at most $\bar{\eta}$, and at most $1 + \bar{\eta}^2\big(R_\kappa(C) - 1\big)$ for arbitrary predictions.
- **Lower bounds (Section 4).** We prove that the C-consistency guarantee for any job-wise weighted mechanism requires robustness at least $R \geq \max\left\{n, \frac{(n-1)C}{C-1}\right\}$, for every $C > 1$. This exactly matches EDGESKIP with an optimal reference schedule and determines the consistency–robustness frontier for the entire weighted class. Beyond the weighted class, for arbitrary deterministic strategyproof mechanisms with no computational restriction, we show the lower bound $R \geq \max\left\{n, \frac{C}{C-1}\right\}$ for every $C > 1$, while $C = 1$ forces unbounded robustness.
- **Empirical evaluation (Section 5).** We evaluate the polynomial-time mechanisms on the Heterogeneous Computing Scheduling Problem (HCSP) benchmark instances of Braun et al. (2001), using predictions obtained by synthetically perturbing the true processing-time matrices. Across the 21 synthetic-noise settings, EDGESKIP improves on SCALED-

GREEDY in 17, while TOLERANTEDGESKIP improves on ERRORTOLERANTSCALED-GREEDY in 16; both comparisons favor our mechanisms at every sparse-error level.

1.2 ADDITIONAL RELATED WORK

For deterministic strategyproof scheduling, the initial lower bound of two (Nisan & Ronen, 1999) was strengthened in successive works (Christodoulou et al., 2009; Koutsoupias & Vidali, 2013; Giannakopoulos et al., 2021; Dobzinski & Shaulker, 2020), eventually yielding a bound growing with n (Christodoulou et al., 2021) and the tight bound n (Christodoulou et al., 2023). For anonymous mechanisms, Ashlagi et al. (2012) had already proved the tight bound. Christodoulou et al. (2025) studied graph- and hypergraph-restricted scheduling domains.

Early learning-augmented results address caching and general online problems (Purohit et al., 2018; Lykouris & Vassilvitskii, 2021); see Mitzenmacher & Vassilvitskii (2022) for an overview. Applications to nonstrategic scheduling and online variants include predicted job sizes, learned weights, unrelated-machine load balancing, and untrusted predictions (Mitzenmacher, 2020; Lattanzi et al., 2020; Li & Xian, 2021; Bampis et al., 2022).

Strategic applications include facility location and general mechanism design (Agrawal et al., 2022; Xu & Lu, 2022). Our upper bounds build on the weighted mechanisms and error-tolerant construction of Balkanski et al. (2023). Cole et al. (2025) use $1 + n + m$ predictions instead of the full matrix: the optimal makespan, a dual-derived value per machine, and a recommended machine per job. Their deterministic strategyproof mechanism is C-consistent and $n\big(2 + 1/(C-2)\big)$-robust for $C > 2$, giving $(C, R) = (4, 2.5n)$.

2 PRELIMINARIES

There is a set $N = [n]$ of machines, with $n \geq 2$, and a set $M = [m]$ of jobs. For fixed n, all consistency and robustness guarantees are uniform over every finite job count $m \geq 1$. The processing time of job j on machine i is $p_{ij} > 0$, and we write $\mathbf{p} = (p_{ij})_{i \in N, j \in M}$ for the resulting instance. An allocation $\mathbf{x} \in \{0,1\}^{n \times m}$ assigns every job to exactly one machine, subject to $\sum_{i \in N} x_{ij} = 1$ for each $j \in M$. We use the terms allocation and schedule interchangeably. The load of machine i and the makespan of the allocation are

$$L_i(\mathbf{p}, \mathbf{x}) = \sum_{j \in M} p_{ij} x_{ij}, \qquad \mathrm{MS}(\mathbf{p}, \mathbf{x}) = \max_{i \in N} L_i(\mathbf{p}, \mathbf{x}).$$

We use $\mathbf{x}^*(\mathbf{p})$ for a fixed optimal allocation that minimizes the makespan and write

$$\mathrm{OPT}(\mathbf{p}) = \min_{\mathbf{x}} \mathrm{MS}(\mathbf{p}, \mathbf{x}) = \mathrm{MS}(\mathbf{p}, \mathbf{x}^*(\mathbf{p})).$$

In the strategic setting, each machine i is controlled by an agent whose private type is the vector $\mathbf{p}_i = (p_{ij})_{j \in M}$. A deterministic direct-revelation mechanism consists of an allocation rule $\mathbf{x}$ and payments $\boldsymbol{\pi} = (\pi_i)_{i \in N}$. If the true type of machine i is $\mathbf{p}_i$, its utility is its payment minus the true processing cost of the jobs assigned to it. The mechanism is *strategyproof*, or dominant-strategy incentive compatible (DSIC), if truthful reporting maximizes this utility for every machine and every profile of reports by the other machines.

We recall weak monotonicity, used in the deferred proofs, in Appendix A.1. Thus, as is standard in strategyproof scheduling (Nisan & Ronen, 1999), we usually describe and analyze the allocation rule, while the corresponding payments are implicit.

Predictions, consistency, and robustness. In the learning-augmented framework, before the machines report their types, the mechanism receives a positive public prediction $\widehat{\mathbf{p}} = (\widehat{p}_{ij})_{i \in N, j \in M}$ of the processing-time matrix. The prediction is not controlled by the agents. We write $\mathbf{x}(\mathbf{p}; \widehat{\mathbf{p}})$ for the allocation on reports $\mathbf{p}$ under prediction $\widehat{\mathbf{p}}$. For every fixed prediction $\widehat{\mathbf{p}}$, the map $\mathbf{p} \mapsto \mathbf{x}(\mathbf{p}; \widehat{\mathbf{p}})$ is required to be a DSIC allocation rule.

The approximation ratio on the pair $(\mathbf{p}, \widehat{\mathbf{p}})$ is

$$\rho(\mathbf{p}, \widehat{\mathbf{p}}) = \frac{\mathrm{MS}(\mathbf{p}, \mathbf{x}(\mathbf{p}; \widehat{\mathbf{p}}))}{\mathrm{OPT}(\mathbf{p})}.$$

A mechanism is C-*consistent* if its approximation ratio is at most C when the prediction is correct (i.e., $\mathbf{p} = \widehat{\mathbf{p}}$), and it is R-*robust* if the same guarantee R holds for arbitrary predictions (Balkanski et al., 2023). Formally,

$$\sup_{\mathbf{p}} \rho(\mathbf{p}, \mathbf{p}) \le C, \qquad \sup_{\mathbf{p}, \widehat{\mathbf{p}}} \rho(\mathbf{p}, \widehat{\mathbf{p}}) \le R.$$

The case $C = 1$ is called perfect consistency. For error-dependent guarantees, we use the standard symmetric multiplicative error:

$$\eta(\mathbf{p}, \widehat{\mathbf{p}}) = \max_{i \in N, j \in M} \left\{ \frac{p_{ij}}{\widehat{p}_{ij}}, \frac{\widehat{p}_{ij}}{p_{ij}} \right\}.$$

Thus $\eta \ge 1$, with $\eta = 1$ exactly when the prediction is correct. An error-tolerant mechanism is also given a tolerance parameter $\bar{\eta} \ge 1$; its error-dependent guarantee applies when $\eta(\mathbf{p}, \widehat{\mathbf{p}}) \le \bar{\eta}$.

Reference schedules. Our mechanisms first compute a schedule from the prediction $\widehat{\mathbf{p}}$. Let $\widehat{\sigma} : M \to N$ be such a reference schedule. We call $\widehat{\sigma}$ a κ-approximate reference schedule if its predicted makespan satisfies $\max_{i \in N} \sum_{j : \widehat{\sigma}(j) = i} \widehat{p}_{ij} \le \kappa \, \mathrm{OPT}(\widehat{\mathbf{p}})$. For each job j, we call $\widehat{\sigma}(j)$ its reference machine. The case $\kappa = 1$ is useful for stating the tradeoff without a polynomial-time requirement. For polynomial-time mechanisms, we use a schedule returned by the classical 2-approximation algorithm, so $\kappa = 2$ (Lenstra et al., 1990).

Job-wise weighted mechanisms. We call the following class *job-wise weighted mechanisms*; it includes the mechanisms of Balkanski et al. (2023). Let $\mathbf{r} = (r_{ij})_{i \in N, j \in M}$ be a positive weight matrix that may depend on the prediction and on public mechanism parameters, but not on the machines' reports. The associated weighted mechanism $M_{\mathbf{r}}$ assigns each job independently according to

$$M_{\mathbf{r}}(\mathbf{p})(j) \in \arg\min_{i \in N} \{ r_{ij} p_{ij} \}.$$

We call $r_{ij} p_{ij}$ the *weighted score* of machine i for job j. All ties are resolved by a deterministic rule fixed before the machines report their types; the mechanisms developed below give priority to a machine selected from the prediction. For a fixed prediction and job, and fixed reports by the other machines, there is a threshold below which machine i wins the job; equality is resolved by the fixed tie-breaking rule. Paying this threshold for each allocated job and summing the payments over jobs gives a DSIC implementation.

3 Improved Upper Bounds

We introduce EdgeSkip and outline its analysis through a partial-matching reduction and a forest-charging argument. We then give an error-tolerant variant and its guarantees; full proofs are in Appendix A.

3.1 The EdgeSkip Mechanism

We introduce a new job-wise weighted mechanism, called EdgeSkip. Fix $\gamma > 0$ and a κ-approximate reference schedule $\widehat{\sigma}$, and let $T = \max_{i \in N} \sum_{j : \widehat{\sigma}(j) = i} \widehat{p}_{ij}$ be its predicted makespan. Our mechanism first constructs a target machine $y(j)$ for every job. Initially, the target is the reference machine $\widehat{\sigma}(j)$. For each machine i, a counter I_i starts at zero and records the predicted load of the jobs moved into i. The mechanism considers every pair (i, j) with $\widehat{p}_{ij} < \widehat{p}_{\widehat{\sigma}(j), j}$, in nonincreasing order of the predicted speedup $\frac{\widehat{p}_{\widehat{\sigma}(j), j}}{\widehat{p}_{ij}}$. It moves j to i only if j has not moved before and the new incoming load would remain at most γT. We call the inequality $I_i + \widehat{p}_{ij} \le \gamma T$ the *capacity test* for pair (i, j).

The resulting assignment y is the *target schedule*, and $\mathbf{r}^0$ is the corresponding *base weight matrix*. For each job j, a machine with $\widehat{p}_{ij} \ge \widehat{p}_{y(j), j}$ receives weight one, while a machine with $\widehat{p}_{ij} < \widehat{p}_{y(j), j}$ receives weight $\frac{\widehat{p}_{y(j), j}}{\widehat{p}_{ij}}$. Thus $r^0_{ij} \widehat{p}_{ij} = \max\{\widehat{p}_{ij}, \widehat{p}_{y(j), j}\} \ge \widehat{p}_{y(j), j}$. For $i = y(j)$, equality holds,

Mechanism 1: EDGESKIP

Input: Prediction $\widehat{\mathbf{p}}$, reports $\mathbf{p}$, reference schedule $\widehat{\sigma}$, and parameter $\gamma > 0$
Set $T \leftarrow \max_{i \in N} \sum_{j:\widehat{\sigma}(j)=i} \widehat{p}_{ij}$ to be the predicted makespan of $\widehat{\sigma}$;
Set $y(j) \leftarrow \widehat{\sigma}(j)$ for every $j \in M$, and $I_i \leftarrow 0$ for every $i \in N$;
Let $E = \{(i,j) : \widehat{p}_{ij} < \widehat{p}_{\widehat{\sigma}(j),j}\}$;
Order E by nonincreasing $\frac{\widehat{p}_{\widehat{\sigma}(j),j}}{\widehat{p}_{ij}}$, with ties fixed in advance;
for $(i,j) \in E$, in this order **do**
  if $y(j) = \widehat{\sigma}(j)$ and $I_i + \widehat{p}_{ij} \leq \gamma T$ **then**
    $y(j) \leftarrow i$ and $I_i \leftarrow I_i + \widehat{p}_{ij}$;
Set $r^0_{ij} \leftarrow \max\{1, \frac{\widehat{p}_{y(j),j}}{\widehat{p}_{ij}}\}$ for every i, j;
Assign j to a machine minimizing weighted score $r^0_{ij} p_{ij}$, giving $y(j)$ priority in a tie;

and the tie-breaking rule assigns j to $y(j)$ when $\mathbf{p} = \widehat{\mathbf{p}}$; under other reports, a different machine may minimize $r^0_{ij} p_{ij}$.

EDGESKIP follows the predicted-speedup ordering of SCALEDGREEDY (Balkanski et al., 2023), but changes both capacity handling and weights. SCALEDGREEDY accepts jobs while a machine's incoming load is below γT, allowing the last job to exceed this threshold before closing the machine. EDGESKIP instead rejects each pair that would exceed capacity and continues scanning, leaving the job available and the machine open. For a moved job j, SCALEDGREEDY retains reference-based weights $\widehat{p}_{\widehat{\sigma}(j),j}/\widehat{p}_{ij}$ on machines faster than its target, whereas EDGESKIP uses target-based weights $\widehat{p}_{y(j),j}/\widehat{p}_{ij}$.

However, this change makes the robustness analysis less direct. A failed capacity test implies only $I_i + \widehat{p}_{ij} > \gamma T$, not $I_i \geq \gamma T$: the rejected job itself may account for most of the excess. Thus the accepted incoming load on i cannot be charged by itself. The robustness proof below instead accounts for the accepted jobs together with the rejected job and uses a dependency forest to control repetitions across different rejected pairs.

Theorem 1. *For every κ-approximate reference schedule and every $C > \kappa$, run* EDGESKIP *with $\gamma = (C-\kappa)/\kappa$. The resulting mechanism is strategyproof, C-consistent, and $R_\kappa(C)$-robust, where*

$$R_\kappa(C) = \begin{cases} \max\left\{n, n-1+\dfrac{\kappa n - 1}{C-\kappa}\right\}, & \kappa < C \leq \kappa+1, \\ \max\left\{n, n-2+\dfrac{\kappa n}{C-\kappa}\right\}, & \kappa+1 < C \leq 2\kappa, \\ \max\left\{n, n-1+\dfrac{\kappa(n-1)}{C-\kappa}\right\}, & C > 2\kappa. \end{cases} \tag{3.1}$$

The two relevant reference schedules are an optimal schedule ($\kappa = 1$) and the best polynomial-time schedule used in our results, namely the standard 2-approximate schedule ($\kappa = 2$).

Corollary 1. *With $\kappa = 1$ and $\gamma = C - 1$,* EDGESKIP *is C-consistent and $\max\{n, \frac{(n-1)C}{C-1}\}$-robust for every $C > 1$. With $\kappa = 2$ and $\gamma = 1$, it is polynomial-time, 4-consistent, and $(2n-2)$-robust.*

The $\kappa = 1$ guarantee matches the information-theoretic lower bound in Section 4.2 for every $C > 1$, and therefore gives the exact consistency–robustness frontier for job-wise weighted mechanisms.

The $\kappa = 2$ guarantee is our main polynomial-time point $(C, R) = (4, 2n-2)$, which improves the $(6, 2n)$ guarantee of SCALEDGREEDY (Balkanski et al., 2023). At the same consistency level, it also improves the $(4, 2.5n)$ consistency–robustness guarantee of Cole et al. (2025) (CGJ). This comparison concerns guarantees under different prediction interfaces: the CGJ mechanism is polynomial-time once its predicted value of OPT, machine-specific dual values, and one predicted machine per job are supplied, whereas EDGESKIP computes its $\kappa = 2$ reference schedule in polynomial time from the full predicted processing-time matrix.

3.2 ANALYSIS OF EDGESKIP

The strategyproofness and consistency claims follow from Lemma 2, whose statement and proof are given in Appendix A.2.

The consistency of SCALEDGREEDY (Balkanski et al., 2023) is $(2+\gamma)\kappa$. Before each acceptance in SCALEDGREEDY, machine i's incoming load is below γT, and the accepted job j has predicted processing time less than $\widehat{p}_{\widehat{\sigma}(j),j} \le T$. Thus its incoming load can reach $(1+\gamma)T$, and adding the remaining reference load gives a target load of at most $(2+\gamma)T$. In contrast, EDGESKIP keeps the incoming load at most γT, so its target load is at most $(1+\gamma)T$. Both mechanisms recover their respective target schedules under correct predictions, so the tighter incoming-load bound explains the improvement in consistency. For the standard 2-approximate reference schedule and $\gamma = 1$, this changes the consistency factor from 6 to 4.

For robustness, fix an output machine h and group its assigned jobs by their machines in an optimal allocation. From each nonempty group associated with machine i, retain a job maximizing r^0_{ij}/r^0_{hj}. These representatives form a partial matching Q_h, and the weighted allocation rule gives

$$L_h(\mathbf{p},\mathbf{x}) \le w(Q_h)\,\mathrm{OPT}(\mathbf{p}), \qquad w(Q) = \sum_{(i,j)\in Q} \frac{r^0_{ij}}{r^0_{hj}}.$$

The construction and derivation are given in Appendix A.5.

The technical statement and proof of Lemma 3 are given in Appendix A.3. The set $\mathcal{L}$ consists of the leaves of a dependency forest that records overlaps between rejected jobs and previously accepted jobs. The path products P_e account for these overlaps, leaving disjoint groups of jobs at the leaves. The refined budget (A.3) follows by reassigning each leaf group to its associated machine and using the fact that every feasible predicted schedule has makespan at least T/κ.

To conclude the robustness analysis, Lemma 3 bounds every partial matching with a high-weight edge by

$$w(Q) \le |Q| + \sum_{e\in\mathcal{L}} (P_e - 1) \le n - |\mathcal{L}| + \sum_{e\in\mathcal{L}} P_e.$$

The one-leaf and refined budgets handle $|\mathcal{L}| = 1$ and $|\mathcal{L}| \ge 2$, respectively; if there is no high-weight edge, then $w(Q) \le n$. Comparing these bounds yields $w(Q) \le R_\kappa((1+\gamma)\kappa)$, which bounds every machine's load through the matching reduction above. The full derivation is given in Appendix A.5.

3.3 ERROR-DEPENDENT GUARANTEES

Fix a tolerance parameter $\bar{\eta} \ge 1$, chosen in advance. It specifies the largest multiplicative prediction error for which the mechanism is designed to retain an error-dependent guarantee: this guarantee applies when $\eta(\mathbf{p},\widehat{\mathbf{p}}) \le \bar{\eta}$, whereas the unconditional robustness guarantee must hold for arbitrary predictions. Let $\mathbf{r}^0$ and y be the weights and target schedule produced in EDGESKIP, and define

$$\widetilde{r}_{ij} = \begin{cases} \frac{1}{\bar{\eta}^2}, & i = y(j), \\ r^0_{ij}, & i \ne y(j). \end{cases}$$

We call the resulting weighted mechanism TOLERANTEDGESKIP. It assigns each job j to a machine minimizing $\widetilde{r}_{ij}p_{ij}$, giving $y(j)$ priority in a tie. This follows the tolerance-based biasing idea of ERRORTOLERANTSCALEDGREEDY (Balkanski et al., 2023), but discounts only the target machine's weight; for a moved job, the original mechanism discounts every machine no faster than its target.

Theorem 2. TOLERANTEDGESKIP *is strategyproof. If* $\eta(\mathbf{p},\widehat{\mathbf{p}}) \le \bar{\eta}$*, its approximation ratio is at most* $(1+\gamma)\kappa\eta^2$*. For arbitrary predictions, its approximation ratio is at most* $1 + \bar{\eta}^2\big(R_\kappa((1+\gamma)\kappa) - 1\big)$.

The proof is given in Appendix A.4.

For general κ and γ, the corresponding guarantees of ERRORTOLERANTSCALEDGREEDY are $(2+\gamma)\kappa\eta^2$ when $\eta \le \bar{\eta}$ and $n(1+\frac{1}{\gamma})\bar{\eta}^2$ for arbitrary predictions (Balkanski et al., 2023). TOLERANTEDGESKIP strictly improves the first bound to $(1+\gamma)\kappa\eta^2$ and the second to $1+\bar{\eta}^2\big(R_\kappa((1+\gamma)\kappa)-1\big)$.

With a 2-approximate reference schedule and $\gamma = 1$, TOLERANTEDGESKIP is a $4\eta^2$-approximation whenever $\eta \leq \bar{\eta}$, and an unconditional $1 + (2n-3)\bar{\eta}^2$-approximation. The corresponding bounds for ERRORTOLERANTSCALEDGREEDY are $6\eta^2$ and $2n\bar{\eta}^2$ (Balkanski et al., 2023).

4 INFORMATION-THEORETIC LOWER BOUNDS

This section presents two lower bounds that do not impose any computational restriction. We first use a simple instance with three jobs to prove a lower bound for all deterministic strategyproof mechanisms. We then use a replicated-job construction to prove that every C-consistent job-wise weighted mechanism has robustness at least $\max\{n, (n-1)C/(C-1)\}$, exactly matching the upper bound achieved by EDGESKIP.

4.1 A GENERAL LOWER BOUND

There are two independent obstacles to the robustness guarantee R. First, fixing any prediction $\widehat{\mathbf{p}}$ leaves an ordinary deterministic strategyproof scheduling rule $\mathbf{x}_{\widehat{\mathbf{p}}}(\mathbf{p}) := \mathbf{x}(\mathbf{p}; \widehat{\mathbf{p}})$. For n unrelated machines, no deterministic strategyproof scheduling mechanism can guarantee a makespan approximation ratio better than n (Christodoulou et al., 2023). Therefore, $R \geq n$. The next theorem gives the second obstacle using only two machines and three jobs.

Theorem 3. *Let a deterministic strategyproof mechanism be C-consistent and R-robust on $n \geq 2$ unrelated machines, uniformly over all finite job counts. If $C > 1$, then $R \geq \max\left\{n, \frac{C}{C-1}\right\}$. If $C = 1$, then its robustness is unbounded.*

The proof is given in Appendix A.6. The construction first uses consistency to force a job onto one machine, then applies weak monotonicity to preserve that assignment through two changes of reports, yielding the factor $C/(C-1)$.

For fixed n, the term $\frac{C}{C-1}$ exceeds the prediction-independent bound n exactly when $1 < C < \frac{n}{n-1}$. Outside this range, the theorem gives $R \geq n$.

4.2 A LOWER BOUND FOR JOB-WISE WEIGHTED MECHANISMS

The preceding lower bound applies without any restriction on how the jobs are allocated. For job-wise weighted mechanisms, consistency can force many distinct jobs to satisfy the same large lower bound on the weight ratio between a common pair of machines. The following lemma converts these repeated ratios directly into a worst-case instance.

Lemma 1. *Let $\mathbf{r}$ be a positive weight matrix. Suppose that two distinct machines s, h, a set J of $n-1$ distinct jobs, and a value $K \geq 1$ satisfy $\frac{r_{sj}}{r_{hj}} \geq K$ for every $j \in J$. If there is an additional job $a \notin J$, then the worst-case approximation ratio of $M_{\mathbf{r}}$ is at least $n - 1 + K$.*

The key step is to define $g_i = (\prod_{j \in J} r_{ij})^{1/(n-1)}$ and average over matchings that pair machine i with a and the other machines with J. The arithmetic–geometric mean inequality bounds the average matching weight from below by $(\sum_{\ell \in N} g_\ell)/g_i$. Choosing i with minimum g_i gives at least $n - 1 + K$, since $g_s/g_h \geq K$. Appendix A.7 gives the full argument and constructs reports whose approximation ratio approaches this matching weight.

Theorem 4. *Let a prediction-dependent job-wise weighted mechanism be C-consistent and R-robust on $n \geq 2$ unrelated machines, uniformly over all finite job counts. If $C > 1$, then $R \geq \max\left\{n, \frac{(n-1)C}{C-1}\right\}$. If $C = 1$, then its robustness is unbounded.*

The proof is given in Appendix A.8. The construction uses an anchor job and many small jobs for each leaf machine. Consistency and the pigeonhole principle force at least $n-1$ jobs onto the same leaf, yielding the repeated ratios needed for Lemma 1.

The lower bound in Theorem 4 matches the upper bound of EDGESKIP with an optimal reference schedule: setting $\kappa = 1$ in Theorem 1 gives $R_1(C) = \max\{n, \frac{(n-1)C}{C-1}\}$. Therefore, over job-wise

weighted mechanisms that work uniformly for every finite number of jobs, the optimal robustness of a C-consistent mechanism is $\max\left\{n, \frac{(n-1)C}{C-1}\right\}$ for every $C > 1$.

5 EXPERIMENTS

We evaluate the polynomial-time mechanisms under truthful reports on the HCSP benchmark instances of Braun et al. (2001), using predictions obtained by synthetically perturbing the true processing-time matrices. The sole performance measure is the ratio

$$\rho(\mathbf{p}, \widehat{\mathbf{p}}) = \frac{\mathrm{MS}(\mathbf{p}, \mathbf{x}(\mathbf{p}; \widehat{\mathbf{p}}))}{\mathrm{OPT}(\mathbf{p})},$$

where smaller values are better. All averages below are averages of ρ.

5.1 SETUP

We compare EDGESKIP and TOLERANTEDGESKIP with SCALEDGREEDY, ERRORTOLERANTSCALEDGREEDY, and SIMPLESCALEDGREEDY (Balkanski et al., 2023). For each job, SIMPLESCALEDGREEDY scales every machine's reported processing time by the predicted speedup of that machine relative to the reference machine, truncated to the interval $[1, n]$, and selects a machine with minimum weighted reported processing time. We also include GREEDY, which sends each job to a fastest machine under the reported processing times, and the report-independent schedule computed by the Lenstra–Shmoys–Tardos (LST) 2-approximation algorithm from $\widehat{\mathbf{p}}$, denoted by PREDICTIONONLY (Lenstra et al., 1990). The five prediction-dependent job-wise weighted mechanisms use this same LST reference schedule; PREDICTIONONLY returns the schedule itself. We set $\gamma = 1$ and $\bar{\eta} = 2$ throughout and use fixed deterministic tie-breaking rules.

We use all 12 HCSP expected-time-to-compute matrices, each with 512 jobs and 16 machines. For each of three fixed random seeds used to construct instances, we select eight machines from every matrix and partition all 512 jobs into 16 disjoint batches of 32 jobs. For each resulting instance, we generate

$$\widehat{p}_{ij} = p_{ij}\eta^{z_{ij}}, \qquad \eta \in \{1, 1.1, 1.25, 1.5, 2, 3, 5, 10\}.$$

Each exponent matrix is rescaled so that $\max_{i,j} |z_{ij}| = 1$, making its realized symmetric multiplicative error exactly η. We use three error structures: entrywise independent and identically distributed (i.i.d.) exponents uniform on $[-1, 1]$; machine–job error with correlated exponents $z_{ij} = a_i + b_j + 0.15\epsilon_{ij}$, where a_i, b_j, and ϵ_{ij} are independent standard normal random variables representing machine, job, and residual effects, respectively; and sparse error, in which only 5% of the entries have nonzero exponents drawn uniformly from $[-1, 1]$. For every $\eta > 1$, error structure, and batch, we use three fixed perturbation seeds. The case $\eta = 1$ is the common perfect prediction.

For evaluation only, we use a mixed-integer program to certify $\mathrm{OPT}(\mathbf{p})$. This value is used solely in the denominator of ρ; it is never supplied to a mechanism and is never used to construct the reference schedule. For each of the seven reported methods, we use the same 36,864 HCSP trial inputs. We first average over perturbation seeds, then over batches, and then over the three constructions. Finally, we give equal weight to each of the 12 source matrices; we call the resulting quantity the macro average.

5.2 EXPERIMENTAL RESULTS

Figure 1 presents the direct HCSP comparisons between each proposed mechanism and its SCALEDGREEDY counterpart.

Across the 21 noisy settings, EDGESKIP has lower macro-averaged ρ than SCALEDGREEDY in 17, while TOLERANTEDGESKIP improves on ERRORTOLERANTSCALEDGREEDY in 16. Both paired comparisons favor our mechanisms at all seven noisy values of η under sparse error. The advantage of our tolerant mechanism is not uniform: under machine–job error at $\eta = 10$, TOLERANTEDGESKIP reaches 3.8601, compared with 2.5948 for ERRORTOLERANTSCALEDGREEDY, and is the worst of all seven methods in that setting.

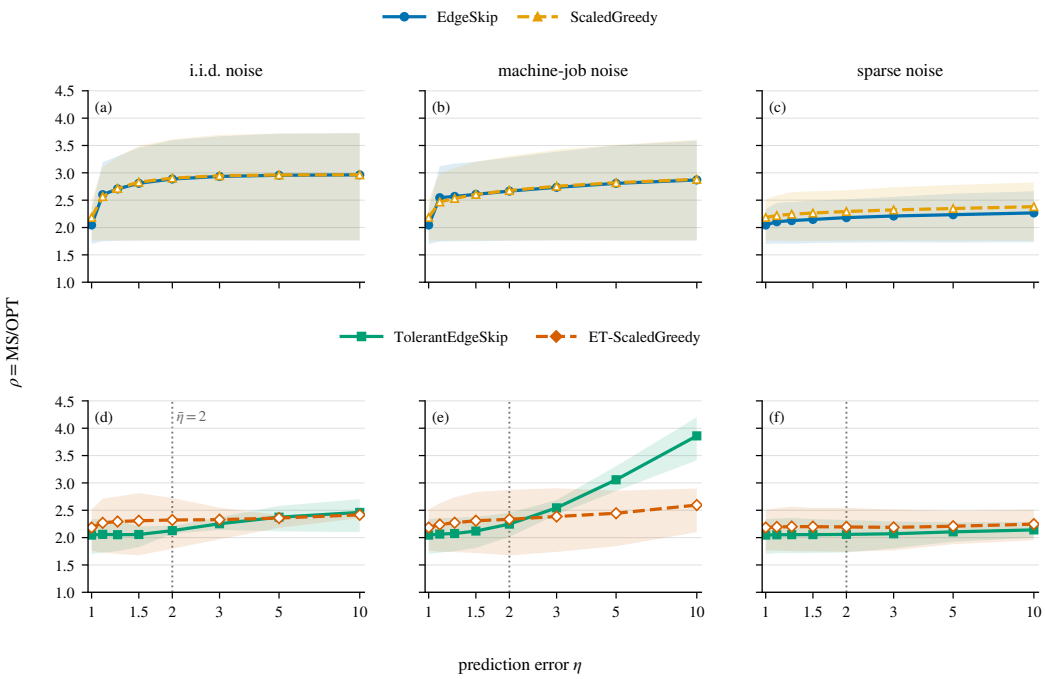


Figure 1: Direct comparisons on HCSP. Columns show i.i.d., machine–job, and sparse prediction errors. The top row compares EDGESKIP with SCALEDGREEDY; the bottom row compares TOLERANTEDGESKIP with ERRORTOLERANTSCALEDGREEDY. Lines are equal-weight means over the 12 source matrices after the within-matrix averaging described in Section 5.1; bands span the 25th–75th percentiles of those 12 matrix means. The common perfect-prediction point is repeated in all three columns, and the dotted line in the lower row marks $\bar{\eta} = 2$. Lower is better.

With perfect HCSP predictions, EDGESKIP reduces the macro-averaged ρ from 2.1837 to 2.0457 relative to SCALEDGREEDY; the same values hold for their tolerant variants. SIMPLESCALEDGREEDY performs best at $\eta = 1$, with 1.3757, followed by PREDICTIONONLY at 1.3854.

The complete HCSP comparison is reported in Appendix C.

6 CONCLUSION

We establish tighter consistency–robustness tradeoffs for strategyproof scheduling with predictions. For job-wise weighted mechanisms, our upper and lower bounds match in the absence of computational restrictions. With an optimal reference schedule, EDGESKIP is C-consistent and attains the optimal robustness $\max\{n, \frac{(n-1)C}{C-1}\}$ for every $C > 1$. With a 2-approximate reference schedule and $\gamma = 1$, it is 4-consistent and $(2n-2)$-robust, improving the polynomial-time tradeoff. The same improvement extends to the error-tolerant setting. Thus, we determine the exact weighted frontier and improve its polynomial-time guarantees, while a gap remains for unrestricted deterministic strategyproof mechanisms.

The main open problem is whether a general deterministic strategyproof mechanism can strictly improve on the optimal weighted frontier when $n \geq 3$, or whether stronger unrestricted lower bounds preclude such an improvement. Another question is to determine the optimal polynomial-time frontier, since an optimal predicted reference schedule cannot in general be computed efficiently. Further directions include obtaining the same tradeoffs using less prediction information and understanding how randomization or more refined error measures change the frontier.

Reproducibility statement

All theoretical assumptions and proofs are provided in Sections 2–4 and Appendices A and B. Section 5 specifies the benchmark, prediction-error models, parameter settings, random seeds, aggregation procedure, and optimization benchmark; Appendix C reports the complete aggregate results.

AI use statement

Generative AI played an important role in strengthening our upper and lower bounds. Starting from weaker bounds established by the authors, GPT helped obtain tighter bounds and suggested proof approaches, while Codex drafted the corresponding proofs. The authors carefully read and checked every proof, verified its correctness, and take responsibility for all theoretical claims. GPT also assisted with experimental design and code development. AI tools additionally assisted with data processing, interpretation of experimental results, literature search, and language and LaTeX editing. The authors take full responsibility for the final content of this work.

References

Priyank Agrawal, Eric Balkanski, Vasilis Gkatzelis, Tingting Ou, and Xizhi Tan. Learning-augmented mechanism design: Leveraging predictions for facility location. In *Proceedings of the 23rd ACM Conference on Economics and Computation*, pp. 497–528, 2022.

Itai Ashlagi, Shahar Dobzinski, and Ron Lavi. Optimal lower bounds for anonymous scheduling mechanisms. *Mathematics of Operations Research*, 37(2):244–258, 2012.

Eric Balkanski, Vasilis Gkatzelis, and Xizhi Tan. Strategyproof scheduling with predictions. In *14th Innovations in Theoretical Computer Science Conference (ITCS)*, pp. 11:1–11:22, 2023.

Evripidis Bampis, Konstantinos Dogeas, Alexander V Kononov, Giorgio Lucarelli, and Fanny Pascual. Scheduling with untrusted predictions. In *IJCAI*, pp. 4581–4587, 2022.

Tracy D. Braun, Howard Jay Siegel, Noah Beck, Ladislau L. Bölöni, Muthucumaru Maheswaran, Albert I. Reuther, James P. Robertson, Mitchell D. Theys, Bin Yao, Debra A. Hensgen, and Richard F. Freund. A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. *Journal of Parallel and Distributed Computing*, 61(6):810–837, 2001. doi: 10.1006/jpdc.2000.1714.

George Christodoulou, Elias Koutsoupias, and Angelina Vidali. A lower bound for scheduling mechanisms. *Algorithmica*, 55(4):729–740, 2009.

George Christodoulou, Elias Koutsoupias, and Annamária Kovács. On the Nisan-Ronen conjecture. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 839–850. IEEE, 2021.

George Christodoulou, Elias Koutsoupias, and Annamária Kovács. A proof of the Nisan-Ronen conjecture. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pp. 672–685, 2023.

George Christodoulou, Elias Koutsoupias, and Annamária Kovács. Truthful allocation in graphs and hypergraphs. *ACM Transactions on Economics and Computation*, 13(2):1–28, 2025.

Richard Cole, Anupam Gupta, and Pranav Jangir. Parsimonious predictions for strategyproof scheduling. In *Advances in Neural Information Processing Systems*, volume 38, pp. 110564–110584, 2025.

Shahar Dobzinski and Ariel Shaulker. Improved lower bounds for truthful scheduling. *arXiv preprint arXiv:2007.04362*, 2020.

Yiannis Giannakopoulos, Alexander Hammerl, and Diogo Poças. A new lower bound for deterministic truthful scheduling. *Algorithmica*, 83(9):2895–2913, 2021.

Elias Koutsoupias and Angelina Vidali. A lower bound of 1+ φ for truthful scheduling mechanisms. *Algorithmica*, 66(1):211–223, 2013.

Silvio Lattanzi, Thomas Lavastida, Benjamin Moseley, and Sergei Vassilvitskii. Online scheduling via learned weights. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1859–1877. SIAM, 2020.

Jan Karel Lenstra, David B Shmoys, and Éva Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Mathematical programming*, 46(1):259–271, 1990.

Shi Li and Jiayi Xian. Online unrelated machine load balancing with predictions revisited. In *International Conference on Machine Learning*, pp. 6523–6532. PMLR, 2021.

Thodoris Lykouris and Sergei Vassilvitskii. Competitive caching with machine learned advice. *Journal of the ACM (JACM)*, 68(4):1–25, 2021.

Michael Mitzenmacher. Scheduling with predictions and the price of misprediction. In *11th Innovations in Theoretical Computer Science Conference (ITCS)*, pp. 14:1–14:18, 2020.

Michael Mitzenmacher and Sergei Vassilvitskii. Algorithms with predictions. *Communications of the ACM*, 65(7):33–35, 2022.

Noam Nisan and Amir Ronen. Algorithmic mechanism design. In *Proceedings of the thirty-first annual ACM symposium on Theory of computing*, pp. 129–140, 1999.

Manish Purohit, Zoya Svitkina, and Ravi Kumar. Improving online algorithms via ml predictions. *Advances in Neural Information Processing Systems*, 31, 2018.

Michael Saks and Lan Yu. Weak monotonicity suffices for truthfulness on convex domains. In *Proceedings of the 6th ACM conference on Electronic commerce*, pp. 286–293, 2005.

Chenyang Xu and Pinyan Lu. Mechanism design with predictions. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*, pp. 571–577, 2022.

A Missing Proofs

A.1 Weak Monotonicity

We will often use *weak monotonicity* in the arguments. Fix the reports of all machines other than i, and consider two reports $\mathbf{p}_i$ and $\mathbf{p}'_i$ by machine i. If $\mathbf{x}_i$ and $\mathbf{x}'_i$ are the corresponding allocation vectors of that machine, then weak monotonicity requires

$$\sum_{j \in M} (p_{ij} - p'_{ij})(x_{ij} - x'_{ij}) \leq 0.$$

On our convex type domain with finitely many allocations, a deterministic allocation rule admits DSIC payments if and only if it satisfies this weak monotonicity condition (Saks & Yu, 2005).

A.2 Strategyproofness and Consistency of EdgeSkip

Lemma 2. EdgeSkip *is strategyproof and* $(1+\gamma)\kappa$*-consistent.*

Proof. The weights $\mathbf{r}^0$ depend only on the prediction, the reference schedule, and γ, not on the machines' reports. Fix the reports of all machines other than i, and consider two reports $\mathbf{p}_i$ and $\mathbf{p}'_i$ by machine i, with corresponding allocation vectors $\mathbf{x}_i$ and $\mathbf{x}'_i$. For every job j, if the two reports differ on p_{ij}, the lower reported processing time can only cause machine i to gain the job, while the higher one can only cause it to lose the job. Hence $(p_{ij} - p'_{ij})(x_{ij} - x'_{ij}) \leq 0$. Summing over all jobs gives weak monotonicity, and therefore the allocation rule admits strategyproof payments.

Suppose now that $\mathbf{p} = \widehat{\mathbf{p}}$. The target machine $y(j)$ has weight $r^0_{y(j),j} = 1$, so its weighted score is $\widehat{p}_{y(j),j}$. For every machine i, we have $r^0_{ij}\widehat{p}_{ij} = \max\{\widehat{p}_{ij}, \widehat{p}_{y(j),j}\} \geq \widehat{p}_{y(j),j}$. The target-priority tie-breaking rule therefore assigns every job j to $y(j)$.

Consider the load of a machine i in this target schedule. The jobs that remain from its reference schedule have total processing time at most T. Every other job assigned to i was counted in I_i,

and the capacity test maintains $I_i \le \gamma T$. The target load of i is consequently at most $(1+\gamma)T$. Since $\mathbf{p} = \widehat{\mathbf{p}}$ and $T \le \kappa\,\mathrm{OPT}(\widehat{\mathbf{p}})$, this load is at most $(1+\gamma)\kappa\,\mathrm{OPT}(\mathbf{p})$, proving the claimed consistency. □

A.3 Excess-Weight Forest Bound

Lemma 3 (Excess-weight forest bound)**.** *Fix a machine h and a partial matching Q. For each edge $e = (i,j) \in Q$, define its weight by $w_e := \frac{r^0_{ij}}{r^0_{hj}}$. Suppose $w_e > 1$ for some $e \in Q$. There exist a nonempty set $\mathcal{L} \subseteq \{e \in Q : w_e > 1\}$ and values $P_e \ge 1$, $e \in \mathcal{L}$, such that*

$$\sum_{e \in Q : w_e > 1} (w_e - 1) \le \sum_{e \in \mathcal{L}} (P_e - 1), \tag{A.1}$$

$$\sum_{e \in \mathcal{L}} P_e < \frac{n}{\gamma}, \tag{A.2}$$

$$\sum_{e \in \mathcal{L}} P_e \le \frac{n}{\gamma} - \max\left\{0, \frac{1}{\kappa\gamma} - 1\right\}, \tag{A.3}$$

$$\mathcal{L} = \{e\} \quad \Longrightarrow \quad P_e < \frac{n-1}{\gamma}. \tag{A.4}$$

Proof. Consider $e = (i,j) \in Q$ with $w_e > 1$. Since $r^0_{hj} \ge 1$, we have $r^0_{ij} > 1$, and hence

$$r^0_{ij} = \frac{\widehat{p}_{y(j),j}}{\widehat{p}_{ij}}, \qquad \widehat{p}_{ij} < \widehat{p}_{y(j),j} \le \widehat{p}_{\widehat{\sigma}(j),j}.$$

Hence, $(i,j) \in E$ and was therefore considered by EdgeSkip. If the target-schedule construction moved j from $\widehat{\sigma}(j)$ to its final target $y(j)$, then $(y(j),j)$ is the pair that performed this move. Because $\widehat{p}_{ij} < \widehat{p}_{y(j),j}$, the pair (i,j) has strictly larger predicted speedup than $(y(j),j)$ and was therefore processed first, while j was still available. If the construction never moved j, then j likewise remained available when (i,j) was processed. In either case, (i,j) was processed while j was available but did not move j to i; it must therefore have been rejected by the capacity test.

Let A_e be the set of jobs ℓ for which the pair (i,ℓ) was accepted before (i,j) was processed. Equivalently, these are precisely the jobs that had already been moved from their reference machines into i when (i,j) failed the capacity test. Thus, immediately before this test, $I_i = \sum_{\ell \in A_e} \widehat{p}_{i\ell}$, and $j \notin A_e$. Put

$$a_e = \frac{\widehat{p}_{\widehat{\sigma}(j),j}}{\widehat{p}_{ij}}.$$

The rejection and the global processing order give

$$\widehat{p}_{ij} + \sum_{\ell \in A_e} \widehat{p}_{i\ell} > \gamma T, \tag{A.5}$$

$$\frac{\widehat{p}_{\widehat{\sigma}(\ell),\ell}}{\widehat{p}_{i\ell}} \ge a_e \qquad (\ell \in A_e), \tag{A.6}$$

while

$$w_e \le \frac{\widehat{p}_{y(j),j}}{\widehat{p}_{ij}} \le a_e.$$

Make one vertex for every edge $e \in Q$ with $w_e > 1$, and draw an arc $e \to e'$ when $j \in A_{e'}$. Each vertex has at most one outgoing arc because a job moves at most once and the machine endpoints of the edges in Q are distinct. Every arc follows the processing order, so there is no directed cycle. Moreover, an undirected cycle together with the outdegree-one condition would force a directed cycle. The resulting graph is therefore a forest oriented toward its roots. Let $\mathcal{L}$ be its set of leaves, where a leaf has indegree zero, and let P_e be the product of the vertex weights on the path from e to its root.

A backward induction gives $P_e \le a_e$ for every vertex. The claim holds at a root because $w_e \le a_e$. If $e = (i, j)$ has parent e', then $j \in A_{e'}$, so j was moved to $y(j)$ before e' was rejected. The processing order gives

$$\frac{\widehat{p}_{\widehat{\sigma}(j),j}}{\widehat{p}_{y(j),j}} \ge a_{e'}.$$

Assuming $P_{e'} \le a_{e'}$, we obtain

$$\begin{aligned} P_e &= w_e P_{e'} \\ &\le \frac{\widehat{p}_{y(j),j}}{\widehat{p}_{ij}} a_{e'} \\ &\le \frac{\widehat{p}_{y(j),j}}{\widehat{p}_{ij}} \frac{\widehat{p}_{\widehat{\sigma}(j),j}}{\widehat{p}_{y(j),j}} = a_e. \end{aligned}$$

For a leaf $e = (i_e, j_e)$, define

$$C_e = \{j_e\} \cup A_e, \qquad x_e = \sum_{j \in C_e} \widehat{p}_{i_e j}, \qquad W_e = \sum_{j \in C_e} \widehat{p}_{\widehat{\sigma}(j),j}.$$

Equations (A.5)–(A.6) and $P_e \le a_e$ imply

$$x_e > \gamma T, \qquad W_e \ge P_e x_e. \tag{A.7}$$

The sets C_e for $e \in \mathcal{L}$ are pairwise disjoint. Indeed, the sets A_e are disjoint for distinct machines i_e, the jobs j_e are distinct because Q is a matching, and an inclusion $j_e \in A_{e'}$ would give the leaf e' an incoming arc. Consequently,

$$\gamma T \sum_{e \in \mathcal{L}} P_e < \sum_{e \in \mathcal{L}} W_e \le \sum_{j \in M} \widehat{p}_{\widehat{\sigma}(j),j} \le nT,$$

which proves (A.2). If there is only one leaf $e = (i_e, j_e)$, no job in C_e has reference machine i_e, since every such job corresponds to a strictly faster pair into i_e. Hence $W_e \le (n-1)T$, which proves (A.4).

For the refined budget, remove all jobs in the sets C_e, $e \in \mathcal{L}$, from the reference schedule, and let b_i be the remaining load of machine i. Reassign every C_e to machine i_e, leaving all other jobs on their reference machines. This is feasible because the sets C_e are pairwise disjoint and the machines i_e are distinct. Its load is $b_{i_e} + x_e$ on each machine i_e and b_i on every other machine. Since $T \le \kappa \, \mathrm{OPT}(\widehat{\mathbf{p}})$, every feasible schedule has makespan at least T/κ. Thus, the reassigned schedule has some machine load of at least T/κ. Writing $z_e = x_e - \gamma T > 0$ and using $P_e \ge 1$, we get

$$\sum_i b_i + \sum_{e \in \mathcal{L}} P_e z_e \ge \max\left\{0, \left(\frac{1}{\kappa} - \gamma\right) T\right\}.$$

By the definitions of W_e and b_i, the total predicted processing time in the reference schedule decomposes as

$$\begin{aligned} nT &\ge \sum_{j \in M} \widehat{p}_{\widehat{\sigma}(j),j} \\ &= \sum_{e \in \mathcal{L}} W_e + \sum_i b_i \\ &\ge \gamma T \sum_{e \in \mathcal{L}} P_e + \sum_{e \in \mathcal{L}} P_e z_e + \sum_i b_i \\ &\ge \gamma T \sum_{e \in \mathcal{L}} P_e + \max\left\{0, \left(\frac{1}{\kappa} - \gamma\right) T\right\}, \end{aligned}$$

which proves (A.3).

It remains to prove the excess inequality. For any rooted tree with vertex weights at least one,

$$\sum_v (w_v - 1) \le \sum_{e \text{ leaf}} (P_e - 1).$$

For a one-vertex tree this is equality. Otherwise remove a root of weight w, and let P'_e be the leaf products in the remaining subtrees. After applying induction to those subtrees, the required step is

$$(w-1)+\sum_e (P'_e-1) \le \sum_e (wP'_e-1),$$

which is equivalent to $(w-1)(\sum_e P'_e - 1) \ge 0$. Summing over the tree components proves (A.1). □

A.4 Proof of Theorem 2

Proof. The weights $\widetilde{\mathbf{r}}$ depend only on public information, so TolerantEdgeSkip is strategyproof. Let $\eta = \eta(\mathbf{p}, \widehat{\mathbf{p}}) \le \bar{\eta}$. Fix a job j. For the target machine $y(j)$,

$$\widetilde{r}_{y(j),j} p_{y(j),j} \le \frac{\eta \widehat{p}_{y(j),j}}{\bar{\eta}^2} \le \frac{\widehat{p}_{y(j),j}}{\bar{\eta}}.$$

For any machine $i \ne y(j)$,

$$\widetilde{r}_{ij} p_{ij} = r^0_{ij} p_{ij} \ge \frac{r^0_{ij} \widehat{p}_{ij}}{\eta} = \frac{\max\{\widehat{p}_{y(j),j}, \widehat{p}_{ij}\}}{\eta} \ge \frac{\widehat{p}_{y(j),j}}{\bar{\eta}}.$$

Therefore, $\widetilde{r}_{y(j),j} p_{y(j),j} \le \widetilde{r}_{ij} p_{ij}$, and the tie-breaking rule selects $y(j)$. Since this holds for every job, TolerantEdgeSkip returns the schedule y. As shown in the proof of Lemma 2, the load of every machine in y, evaluated at $\widehat{\mathbf{p}}$, is at most $(1+\gamma)T$. Moreover, $p_{ij} \le \eta \widehat{p}_{ij}$ for every i, j. Hence, for every machine h, the actual load is

$$\sum_{j: y(j)=h} p_{hj} \le \eta \sum_{j: y(j)=h} \widehat{p}_{hj} \le \eta(1+\gamma)T.$$

Taking the maximum over h, the makespan of y under $\mathbf{p}$ is at most $\eta(1+\gamma)T$. If $\mathbf{x}^*(\mathbf{p})$ is optimal for the actual instance, its predicted makespan is at most $\eta \operatorname{OPT}(\mathbf{p})$, and hence $\operatorname{OPT}(\widehat{\mathbf{p}}) \le \eta \operatorname{OPT}(\mathbf{p})$. Using $T \le \kappa \operatorname{OPT}(\widehat{\mathbf{p}})$ proves the error-dependent guarantee $(1+\gamma)\kappa\eta^2$.

For the unconditional guarantee, let $\mathbf{x}$ be the allocation produced on $(\mathbf{p}, \widehat{\mathbf{p}})$. For every i, h, j, observe that

$$\frac{\widetilde{r}_{ij}}{\widetilde{r}_{hj}} \le \bar{\eta}^2 \frac{r^0_{ij}}{r^0_{hj}}.$$

Fix a machine h and a partial matching Q. If $(h, j) \in Q$, that edge contributes 1, and applying the preceding ratio bound to the other edges and $w(Q) \le R_\kappa((1+\gamma)\kappa)$ gives

$$\sum_{(i,j)\in Q} \frac{\widetilde{r}_{ij}}{\widetilde{r}_{hj}} \le 1 + \bar{\eta}^2 \big(R_\kappa((1+\gamma)\kappa) - 1\big).$$

If Q contains no such edge, then $|Q| \le n-1$. When every edge has weight at most one, $w(Q) \le n-1 \le R_\kappa((1+\gamma)\kappa) - 1$. Otherwise, retaining $|Q|$ in (A.8) and applying the same leaf-budget bounds saves one in each case, again giving $w(Q) \le R_\kappa((1+\gamma)\kappa) - 1$. Hence

$$\sum_{(i,j)\in Q} \frac{\widetilde{r}_{ij}}{\widetilde{r}_{hj}} \le \bar{\eta}^2 \big(R_\kappa((1+\gamma)\kappa) - 1\big).$$

Set $J_i = \{j : x_{hj} = 1,\ x^*_{ij} = 1\}$. For every nonempty J_i, choose j_i maximizing $\widetilde{r}_{ij}/\widetilde{r}_{hj}$; then $Q_h = \{(i, j_i) : J_i \ne \varnothing\}$ is a partial matching. Since $\widetilde{r}_{hj} p_{hj} \le \widetilde{r}_{ij} p_{ij}$ for every $j \in J_i$,

$$\begin{aligned}
L_h(\mathbf{p}, \mathbf{x}) &= \sum_{i: J_i \ne \varnothing} \sum_{j \in J_i} p_{hj} \\
&\le \sum_{i: J_i \ne \varnothing} \frac{\widetilde{r}_{ij_i}}{\widetilde{r}_{hj_i}} \sum_{j \in J_i} p_{ij} \\
&\le \sum_{(i,j_i)\in Q_h} \frac{\widetilde{r}_{ij_i}}{\widetilde{r}_{hj_i}} \operatorname{OPT}(\mathbf{p}) \\
&\le \big[1 + \bar{\eta}^2 \big(R_\kappa((1+\gamma)\kappa) - 1\big)\big] \operatorname{OPT}(\mathbf{p}).
\end{aligned}$$

Taking the maximum over h proves the unconditional robustness bound. □

A.5 Robustness Proof for Theorem 1

Next, we prove the robustness guarantee in Theorem 1 through a partial-matching bound on the load of each machine in the output allocation. Fix arbitrary reports $\mathbf{p}$, let $\mathbf{x}$ be the resulting allocation, and let $\mathbf{x}^*$ be an optimal allocation for $\mathbf{p}$. Fix an arbitrary machine $h \in N$, and set

$$J_i = \{j : x_{hj} = 1,\ x^*_{ij} = 1\}.$$

Thus, J_i consists exactly of the jobs assigned to h by the mechanism and to i by the optimal allocation. For every nonempty J_i, choose $j_i \in J_i$ maximizing r^0_{ij}/r^0_{hj}. The weighted allocation rule gives

$$\begin{aligned}\sum_{j\in J_i} p_{hj} &\le \left(\max_{j\in J_i} \frac{r^0_{ij}}{r^0_{hj}}\right)\left(\sum_{j\in J_i} p_{ij}\right)\\ &\le \frac{r^0_{ij_i}}{r^0_{hj_i}}\,\mathrm{OPT}(\mathbf{p}).\end{aligned}$$

The pairs (i, j_i) have distinct machines and distinct jobs, so they form a partial matching. Consequently, it is enough to bound the weight

$$w(Q) = \sum_{(i,j)\in Q} \frac{r^0_{ij}}{r^0_{hj}}$$

of every partial matching Q between machines and jobs, for every choice of h.

Proof of the robustness guarantee in Theorem 1. Fix a machine h and consider an arbitrary partial matching Q. If every edge of Q has weight at most one, then $w(Q) \le |Q| \le n$. Otherwise, $\mathcal{L}$ is nonempty. Equation (A.1) gives

$$\begin{aligned}w(Q) &= \sum_{e\in Q} \min\{w_e, 1\} + \sum_{e\in Q: w_e>1} (w_e - 1)\\ &\le |Q| + \sum_{e\in\mathcal{L}} (P_e - 1)\\ &\le n - |\mathcal{L}| + \sum_{e\in\mathcal{L}} P_e. \qquad \text{(A.8)}\end{aligned}$$

If $|\mathcal{L}| = 1$, equation (A.4) gives

$$w(Q) < n - 1 + \frac{n-1}{\gamma}.$$

If $|\mathcal{L}| \ge 2$, equations (A.8) and (A.3) give

$$w(Q) \le n - 2 + \frac{n}{\gamma} - \max\left\{0, \frac{1}{\kappa\gamma} - 1\right\}.$$

For $0 < \gamma \le 1/\kappa$, the bound obtained when $|\mathcal{L}| \ge 2$ is $n-1+(n-1/\kappa)/\gamma$, and it is at least the bound obtained when $|\mathcal{L}| = 1$. For $1/\kappa < \gamma \le 1$, the bound obtained when $|\mathcal{L}| \ge 2$ is $n - 2 + n/\gamma$, and it is again at least the bound obtained when $|\mathcal{L}| = 1$. For $\gamma > 1$, the bound obtained when $|\mathcal{L}| = 1$ is the larger one, because it exceeds the bound obtained when $|\mathcal{L}| \ge 2$ by $1 - 1/\gamma$. Combining these three cases with the initial bound $w(Q) \le n$ when all edge weights are at most one gives

$$w(Q) \le R_\kappa((1+\gamma)\kappa)$$

for every partial matching Q.

Let $Q_h = \{(i, j_i) : J_i \ne \varnothing\}$ be the partial matching constructed above. Recalling that $L_h(\mathbf{p}, \mathbf{x}) = \sum_{j:x_{hj}=1} p_{hj}$ is the load of machine h under allocation $\mathbf{x}$, summing the preceding bounds over i

gives

$$\begin{aligned} L_h(\mathbf{p}, \mathbf{x}) &= \sum_{i: J_i \neq \varnothing} \sum_{j \in J_i} p_{hj} \\ &\leq \sum_{i: J_i \neq \varnothing} \frac{r^0_{ij_i}}{r^0_{hj_i}} \sum_{j \in J_i} p_{ij} \\ &\leq \sum_{(i,j_i) \in Q_h} \frac{r^0_{ij_i}}{r^0_{hj_i}} \operatorname{OPT}(\mathbf{p}) \\ &= w(Q_h) \operatorname{OPT}(\mathbf{p}) \\ &\leq R_\kappa((1+\gamma)\kappa) \operatorname{OPT}(\mathbf{p}). \end{aligned}$$

Taking the maximum over h yields the robustness claim. □

A.6 Proof of Theorem 3

Proof. The bound $R \geq n$ follows from the information-theoretic lower bound of Christodoulou et al. (2023) for n unrelated machines without predictions. It remains to prove the consistency-dependent term. First suppose that $C > 1$. Choose $0 < K < \frac{1}{C-1}$, and then choose $0 < \delta < \min\{K, 1\}$ small enough that $K + 1 > C(K + \delta)$. Also choose $0 < \xi < K$. It is enough to use two machines and three jobs a, b, c; on any additional machine, all three jobs have a sufficiently large processing time H. Consider the prediction

$$\widehat{\mathbf{p}} = \begin{array}{c|ccc} & a & b & c \\ \hline 1 & K & 1 & H \\ 2 & H & K & \delta \end{array}.$$

Its optimal makespan is $K + \delta$, attained by assigning a to machine 1 and b, c to machine 2. For large enough H, consistency forces a and c to these machines. It also forces b to machine 2, since assigning it to machine 1 would give makespan $K + 1$, and $K + 1 > C(K + \delta)$ violates the consistency requirement.

Keep this prediction fixed and first change only the report of machine 1 from $(K, 1, H)$ to $(\delta, 1 + \xi, H)$. Let x'_{1a} and x'_{1b} be its new allocation indicators. Weak monotonicity gives

$$(K - \delta)(1 - x'_{1a}) + \xi x'_{1b} \leq 0.$$

Both terms are nonnegative, so machine 1 continues to receive a and not b. This is the first locking step: the processing time of the job already assigned to machine 1 is decreased, while that of the unassigned job is increased. If b or c is now assigned to a machine on which its processing time is H, the approximation ratio is already arbitrarily large as H grows. We may therefore continue with the only remaining case, in which machine 2 receives both jobs.

Keeping the modified report $(\delta, 1 + \xi, H)$ of machine 1 fixed, change only the report of machine 2 from (H, K, δ) to $(H, K - \xi, 1)$. The job c remains on machine 2: otherwise the mechanism already has an arbitrarily large approximation ratio as H grows. Writing x''_{2b} and x''_{2c} for the new allocation indicators, weak monotonicity for this second change gives

$$\xi(1 - x''_{2b}) - (1 - \delta)(1 - x''_{2c}) \leq 0.$$

We have just argued that $x''_{2c} = 1$, and hence this inequality forces $x''_{2b} = 1$. Thus, on the final instance, machine 2 receives both b and c, and the mechanism has makespan at least $K - \xi + 1$. The allocation that assigns a, b to machine 1 and c to machine 2 has makespan $1 + \xi + \delta$. Consequently,

$$R \geq \frac{K + 1 - \xi}{1 + \xi + \delta}.$$

Letting ξ and δ tend to zero and K tend to $\frac{1}{C-1}$ proves $R \geq 1 + \frac{1}{C-1} = \frac{C}{C-1}$. When $C = 1$, the same construction allows K to be arbitrarily large, and hence no finite robustness is possible. □

A.7 Proof of Lemma 1

Proof. For every machine i, let $g_i = \left(\prod_{j\in J} r_{ij}\right)^{1/(n-1)}$. The repeated-ratio assumption gives $g_s/g_h \geq K$. Fix a machine i. For each bijection $\pi : N \setminus \{i\} \to J$, form the partial matching

$$Q_\pi = \{(i,a)\} \cup \{(\ell, \pi(\ell)) : \ell \neq i\}.$$

Averaging its weight relative to machine i over all $(n-1)!$ bijections, and then applying the arithmetic–geometric mean (AM–GM) inequality for each $\ell \neq i$, gives

$$\begin{aligned}
\mathbb{E}_\pi\left[\sum_{(\ell,j)\in Q_\pi} \frac{r_{\ell j}}{r_{ij}}\right] &= 1 + \sum_{\ell\neq i} \frac{1}{n-1} \sum_{j\in J} \frac{r_{\ell j}}{r_{ij}} \\
&\geq 1 + \sum_{\ell\neq i} \left(\prod_{j\in J} \frac{r_{\ell j}}{r_{ij}}\right)^{1/(n-1)} \\
&= 1 + \sum_{\ell\neq i} \frac{g_\ell}{g_i} = \frac{\sum_{\ell\in N} g_\ell}{g_i}.
\end{aligned}$$

Choose i minimizing g_i. Every ratio g_ℓ/g_i is at least one, and $g_s/g_i \geq g_s/g_h \geq K$. The last sum is therefore at least $n-1+K$. Hence some bijection π satisfies $\sum_{(\ell,j)\in Q_\pi} r_{\ell j}/r_{ij} \geq n-1+K$. Fix such a bijection π.

It remains to realize this sum as an approximation ratio. Choose $\varepsilon > 0$ smaller than every ratio $r_{\ell j}/r_{ij}$ appearing in Q_π. For each $(\ell, j) \in Q_\pi$ with $\ell \neq i$, set

$$p_{\ell j} = 1, \qquad p_{ij} = \frac{r_{\ell j}}{r_{ij}} - \varepsilon,$$

and make the processing time of j on every other machine sufficiently large. Then $r_{ij}p_{ij} < r_{\ell j}p_{\ell j}$, so $M_{\mathbf{r}}$ assigns j to i. For the edge (i,a), set $p_{ia} = 1$ and make the processing time of a on every other machine sufficiently large. Give every remaining job processing time $\delta > 0$ on every machine. The weighted mechanism puts load at least $\sum_{(\ell,j)\in Q_\pi} r_{\ell j}/r_{ij} - n\varepsilon$ on i, whereas assigning each job in Q_π to the machine paired with it gives a feasible schedule of makespan at most $1 + m\delta$, and hence $\mathrm{OPT}(\mathbf{p}) \leq 1 + m\delta$. Letting δ and then ε tend to zero proves the claim. □

A.8 Proof of Theorem 4

Proof. When $C \geq n$, the second term in the claimed bound is at most n, so the bound reduces to $R \geq n$. This follows as in Theorem 3, by fixing the prediction and applying the lower bound for deterministic strategyproof scheduling. When $C = 1$, the unboundedness follows from the same theorem. It therefore suffices to consider $1 < C < n$.

Choose a value $c \in \left(\frac{C-1}{n-1}, 1\right)$. Relabel the n machines as one root machine 0 and leaf machines $1, \ldots, n-1$. There is an anchor job a and, for each leaf i, there are k regular jobs $j_{i,1}, \ldots, j_{i,k}$. For a finite value $H > C$, define the prediction by

$$\begin{aligned}
&\widehat{p}_{0a} = 1, \qquad \widehat{p}_{ia} = H && (i \neq 0), \\
&\widehat{p}_{0j_{i,t}} = \frac{c}{k}, \qquad \widehat{p}_{ij_{i,t}} = \frac{1}{k}, \qquad \widehat{p}_{\ell j_{i,t}} = H && (\ell \notin \{0, i\}).
\end{aligned}$$

Assigning the anchor to the root and each regular job to its own leaf has makespan one. Hence, $\mathrm{OPT}(\widehat{\mathbf{p}}) = 1$.

When the prediction is correct, C-consistency rules out every H-entry. Thus the anchor goes to the root, while a regular job goes either to the root or to its own leaf. If the root receives q regular jobs, its load is $1 + qc/k$, and hence $q \leq \frac{(C-1)k}{c}$. At least $k(n-1-(C-1)/c)$ regular jobs therefore remain on their leaves. Choose a sufficiently large but finite k so that

$$k\left(n - 1 - \frac{C-1}{c}\right) > (n-1)(n-2). \tag{A.9}$$

By the pigeonhole principle, some leaf h receives at least $n-1$ of its own regular jobs.

Fix the weight matrix $\mathbf{r}$ selected for this prediction. For every such job j assigned to h, the weighted-score comparison with the root gives $r_{hj}\frac{1}{k} \le r_{0j}\frac{c}{k}$, and $\frac{r_{0j}}{r_{hj}} \ge \frac{1}{c}$. Choose $n-1$ of these distinct jobs and use the anchor as the additional job in Lemma 1, with $s=0$ and $K=1/c$. Lemma 1 therefore yields $R \ge n-1+\frac{1}{c}$. Letting c decrease to $(C-1)/(n-1)$, and choosing a finite k satisfying (A.9) each time, proves $R \ge n-1+\frac{n-1}{C-1}$. □

B THE TWO-MACHINE CASE

This appendix isolates the special case of two unrelated machines. Its purpose is to record a simple threshold mechanism whose proof avoids the forest argument used for EDGESKIP and whose guarantee can be stronger when the reference schedule is only approximately optimal.

Suppose $N=\{1,2\}$, and fix parameters $C>1$ and $\kappa \ge 1$. Let $\widehat{\sigma}$ be a κ-approximate reference schedule computed from the prediction $\widehat{\mathbf{p}}$, and set $\lambda_C = \max\{1, \frac{1}{C-1}\}$. For each job, the mechanism favors its reference machine by a factor of λ_C: the reference machine receives the job unless the other machine is more than λ_C times faster.

Mechanism 2: Two-machine threshold mechanism

Input: Prediction $\widehat{\mathbf{p}}$, reports $\mathbf{p}$, parameters $C>1$ and $\kappa \ge 1$
Compute a fixed κ-approximate reference schedule $\widehat{\sigma}$ for $\widehat{\mathbf{p}}$, and set $\lambda_C \leftarrow \max\{1, \frac{1}{C-1}\}$;
for each job $j \in M$ **do**
  Assign j to $\widehat{\sigma}(j)$ temporarily;
  for each machine $k \in N$ **do**
    if $p_{\widehat{\sigma}(j),j} > \lambda_C p_{kj}$ **then**
      Reassign j to k;

The rule is a job-wise weighted mechanism: job j has weight 1 on $\widehat{\sigma}(j)$ and weight λ_C on every $k \in N \setminus \{\widehat{\sigma}(j)\}$. The reference machine wins a tie.

Theorem 5. *For every $C>1$ and $\kappa \ge 1$, the two-machine threshold mechanism is deterministic, strategyproof, κC-consistent, and $\max\{2, \frac{C}{C-1}\}$-robust.*

Proof. The reference schedule and λ_C depend only on the public prediction and on the public parameters C and κ. Together with the fixed tie-breaking rule, these choices make the allocation deterministic. Since the weights are fixed independently of the reports and each job is allocated independently, each per-job allocation is monotone in the reported processing time. Summing the resulting per-job inequalities gives weak monotonicity, so the allocation rule admits strategyproof payments.

We next prove consistency. Suppose $\mathbf{p}=\widehat{\mathbf{p}}$, write $\mathbf{x}$ for the resulting allocation, and let T be the makespan of $\widehat{\sigma}$ on $\mathbf{p}$. Since $\widehat{\sigma}$ is κ-approximate, $T \le \kappa\,\mathrm{OPT}(\mathbf{p})$. Fix a machine h, and let k be the other machine. The jobs that both the reference schedule and the mechanism assign to h have total processing time at most T. Every remaining job assigned to h has reference machine k. Such a job moves to h only if $p_{kj} > \lambda_C p_{hj}$, and hence $p_{hj} < \frac{p_{kj}}{\lambda_C}$. These jobs form a subset of the reference load of k, whose total is at most T. Consequently,

$$L_h(\mathbf{p},\mathbf{x}) \le T + \frac{T}{\lambda_C} \le CT \le \kappa C\,\mathrm{OPT}(\mathbf{p}).$$

The inequality $T + T/\lambda_C \le CT$ is an equality when $1 < C \le 2$; when $C>2$, it follows from $\lambda_C = 1$ and $2 < C$. The bound holds for both machines, proving κC-consistency.

For robustness, fix an arbitrary pair $(\mathbf{p},\widehat{\mathbf{p}})$, let $\mathbf{x}$ be the resulting allocation, and fix a machine h. Again let k be the other machine. The jobs assigned to h by both $\mathbf{x}$ and $\mathbf{x}^*(\mathbf{p})$ contribute at most $\mathrm{OPT}(\mathbf{p})$. Consider instead a job j assigned to h by $\mathbf{x}$ and to k by $\mathbf{x}^*(\mathbf{p})$. If h is its reference machine, the threshold rule gives $p_{hj} \le \lambda_C p_{kj}$. If k is its reference machine, the job moves to h, so $p_{hj} < \frac{p_{kj}}{\lambda_C} \le \lambda_C p_{kj}$, where the last inequality uses $\lambda_C \ge 1$. Thus every such job satisfies the same

upper bound, and their total processing time on k under $\mathbf{x}^*(\mathbf{p})$ is at most $\mathrm{OPT}(\mathbf{p})$. It follows that

$$L_h(\mathbf{p}, \mathbf{x}) \le (1 + \lambda_C)\,\mathrm{OPT}(\mathbf{p}).$$

Finally, $1+\lambda_C = \max\{2, \frac{C}{C-1}\}$. Taking the maximum over the two machines proves the robustness claim. □

Taking $\kappa = 1$ in Theorem 5 matches the general lower bound in Theorem 3. Thus, without computational restrictions, the exact frontier for all deterministic DSIC mechanisms on two machines is $\max\{2, \frac{C}{C-1}\}$ for every $C > 1$, not only for the job-wise weighted class.

To compare with Theorem 1, specialize that theorem to $n = 2$ and set $\gamma = C - 1$. The consistency parameter in that theorem is then $(1+\gamma)\kappa = \kappa C$, so both mechanisms are κC-consistent. If $\kappa = 1$, Theorem 1 gives

$$R_1(C) = \max\left\{2, \frac{C}{C-1}\right\},$$

so the two robustness guarantees coincide for every $C > 1$. For arbitrary κ, they also coincide when $C \ge 2$, in which case both robustness bounds equal two.

When $\kappa > 1$ and $1 < C < 2$, the threshold mechanism is strictly stronger: its robustness is $C/(C-1)$, whereas the specialization of Theorem 1 is

$$R_\kappa(\kappa C) = \begin{cases} 1 + \dfrac{2 - 1/\kappa}{C-1}, & 1 < C \le 1 + 1/\kappa, \\ \dfrac{2}{C-1}, & 1 + 1/\kappa < C < 2. \end{cases}$$

Thus the appendix improves the two-machine guarantee in precisely this approximate-reference regime, while Theorem 1 applies to an arbitrary number of machines.

C Additional Experimental Results

Figure 2 gives the complete seven-method HCSP comparison, and Table 1 reports every corresponding macro average. These results use the same hierarchy and the same sole performance measure ρ as the main experiments.

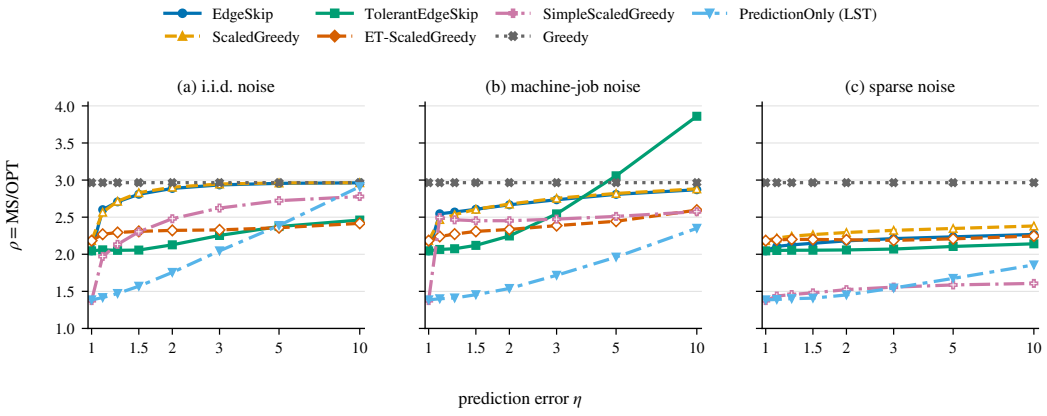


Figure 2: All seven methods on HCSP under i.i.d., machine–job, and sparse prediction errors. Each point is the equal-weight mean over the 12 source matrices after within-matrix averaging. The common perfect-prediction value is repeated at $\eta = 1$ in all three panels. Lower is better.

Table 1: Complete results on the classic HCSP matrices. Entries are source-matrix macro-averaged $\rho = \mathrm{MS} / \mathrm{OPT}$; lower is better. ES, SG, TES, ET-SG, SSG, and Pred.-LST denote EDGESKIP, SCALEDGREEDY, TOLERANTEDGESKIP, ERRORTOLERANTSCALEDGREEDY, SIMPLESCALEDGREEDY, and PREDICTIONONLY (LST), respectively. The perfect-prediction row is shared by all three noise families at $\eta = 1$.

Prediction	η	ES	SG	TES	ET-SG	SSG	Greedy	Pred.-LST
Perfect	1	2.0457	2.1837	2.0457	2.1837	**1.3757**	2.9652	1.3854
i.i.d.	1.1	2.6010	2.5667	2.0595	2.2693	1.9708	2.9652	**1.4156**
	1.25	2.7063	2.7145	2.0529	2.2934	2.1334	2.9652	**1.4710**
	1.5	2.8087	2.8301	2.0565	2.3084	2.3025	2.9652	**1.5689**
	2	2.8875	2.9037	2.1271	2.3218	2.4790	2.9652	**1.7553**
	3	2.9346	2.9471	2.2534	2.3283	2.6227	2.9652	**2.0455**
	5	2.9559	2.9614	2.3735	**2.3575**	2.7225	2.9652	2.3878
	10	2.9628	2.9642	2.4609	**2.4165**	2.7783	2.9652	2.9108
Machine-job	1.1	2.5456	2.4682	2.0650	2.2393	2.4941	2.9652	**1.3999**
	1.25	2.5688	2.5326	2.0743	2.2716	2.4645	2.9652	**1.4107**
	1.5	2.6065	2.6064	2.1196	2.3082	2.4522	2.9652	**1.4547**
	2	2.6663	2.6769	2.2465	2.3342	2.4519	2.9652	**1.5386**
	3	2.7348	2.7547	2.5442	2.3859	2.4736	2.9652	**1.7165**
	5	2.8073	2.8199	3.0576	2.4450	2.5103	2.9652	**1.9613**
	10	2.8699	2.8814	3.8601	2.5948	2.5736	2.9652	**2.3500**
Sparse	1.1	2.1062	2.2191	2.0513	2.1939	1.4367	2.9652	**1.3902**
	1.25	2.1263	2.2435	2.0540	2.2011	1.4569	2.9652	**1.4015**
	1.5	2.1481	2.2651	2.0545	2.1998	1.4799	2.9652	**1.4079**
	2	2.1810	2.2931	2.0583	2.1949	1.5231	2.9652	**1.4514**
	3	2.2109	2.3225	2.0702	2.1872	1.5569	2.9652	**1.5449**
	5	2.2347	2.3474	2.1052	2.2078	**1.5864**	2.9652	1.6744
	10	2.2670	2.3809	2.1408	2.2460	**1.6070**	2.9652	1.8562