
Approximation Algorithms for GPU Pricing under Finite Capacity

Anonymous Author(s)

Affiliation

Address

email

Abstract

1 We study revenue-optimal posted-price design for GPU markets under a finite
2 capacity B . We focus on two settings: *sales*, where each purchase permanently
3 consumes a vendor’s GPU inventory, and *rentals*, where each job occupies GPUs
4 for a finite duration and then releases them. In both settings, the vendor must
5 commit to an anonymous price menu before observing customers’ valuation, GPU
6 demand, and job duration. First, in the sales setting, we show that computing the
7 optimal anonymous static menu is NP-hard, and that the common practice of linear
8 pricing (per-GPU price) can be highly suboptimal. We develop three methods with
9 varying approximation guarantees, all of which outperform linear pricing. Our
10 main method is a convex program (CP) obtained via an ex-ante relaxation of the
11 problem, which obtains a $\frac{1-\epsilon}{2}$ approximation to the optimum when the maximum
12 GPU demand $\bar{N}$ satisfies $\bar{N} \leq \epsilon B$. Building on these results, we analyze the
13 *rental* model. We again show that linear pricing (per-GPU-hour price) is highly
14 suboptimal, and design duration-aware posted-price policies based on slot-coupled
15 ex-ante CP. Our *Throttled CP-Menu Posted Pricing* (TCMPP) policy posts the CP-
16 induced menu randomly and achieves a $\frac{1-\epsilon}{4}$ approximation. We further introduce
17 a slackened version of the same convex program, which reserves capacity in each
18 time slot and achieves stronger guarantees in the small job regime. We corroborate
19 our theoretical results with simulations.

20 1 Introduction

21 GPU usage in the cloud has grown rapidly in recent years, driven by the computational demands of
22 machine learning, video processing, and scientific computing. Major cloud providers (e.g., AWS,
23 GCP) now operate clusters with tens of thousands of accelerators, and a new class of GPU-focused
24 cloud providers (e.g., CoreWeave, Lambda Labs) has emerged to meet excess demand. Yet, GPUs
25 remain scarce with providers routinely facing capacity shortages.

26 *Limits of linear pricing.* GPU pricing has remained largely unchanged since the advent of the cloud,
27 with most vendors adopting *linear pricing*, i.e., a per-GPU-hour rate p , so that a job requesting n
28 GPUs for λ hours pays $p \cdot n \cdot \lambda$. While p may vary by GPU model, commitment type (on-demand, spot,
29 reserved), and current demand, at its core, the pricing structure within each tier is linear in (n, λ) , and
30 this has been the standard approach in both commercial clouds and the theoretical literature [23, 19].
31 However, customers’ valuations need not scale linearly with resource consumption, and in practice,
32 jobs arrive with heterogeneous GPU requirements and durations. One of our key theoretical insights
33 motivating this work is that linear pricing, while simple, can be far from optimal (Theorems 3, 6).

This work. We study the design of revenue-optimal *anonymous* pricing for GPUs (and similar fungible resources) in a market with several, but a finite number of, customers. A vendor chooses a pricing curve as a function of a job’s GPU requirement and/or duration, without knowledge of this job or the customer’s valuation. This setting inherits the traditional challenges of optimal pricing: underpricing loses revenue from high-value customers, while overpricing can suppress demand and leave GPU capacity idle. In addition, our setting introduces two key challenges over traditional settings: (i) *Multi-dimensional menu*: customers request different GPU bundle sizes and job durations, so the vendor must choose a menu (function) over bundle-duration pairs rather than a single scalar price. (ii) *Temporal and capacity coupling*: an accepted job occupies multiple GPUs from the vendor’s finite capacity for its entire duration before releasing them, so the opportunity cost of serving a job depends jointly on its size and duration; moreover, job arrivals are random and unknown ahead of time; as a result, revenue cannot be decomposed across customer types or time periods.

Our contributions. We structure our contributions around two settings. First, we study optimal pricing for GPU *sales*, where each sale permanently consumes a vendor’s GPU inventory and the price is a function of the number of GPUs purchased. In addition to being interesting in its own right, this serves as the foundation for our second setting, where we study optimal pricing for GPU *rentals*, where, each job occupies GPUs for a specified duration and then releases them, capturing the reusable nature of real-world clouds. Here, the price is a function of both the number of GPUs used and duration of the job. In both settings, customer types (value, GPU requirement, duration) are drawn from a known distribution, and the pricing curve is committed to before any customer arrives. While we use GPU markets as the running case study, the model and techniques apply more broadly to finite capacity and reusable resource markets, including cloud accelerators, reservation based compute clusters, hotel rooms and shared vehicles.

1.1 Summary of our results and methods

GPU sales (§2). In our model for GPU sales, a vendor with an inventory of $B \in \mathbb{N}$ GPUs serves m customers arriving sequentially. Each customer needs some number of GPUs N_i and has a private value V_i for that bundle, with these types (N_i, V_i) drawn i.i.d. from a known distribution. The vendor commits to a pricing curve $p(\cdot)$, mapping each bundle size to a price, before observing customer types. A customer submits a purchase if her value exceeds the price for her bundle size, i.e., $V_i \geq p(N_i)$, and the purchase goes through only if enough GPUs remain. The vendor’s expected revenue is the sum of prices collected from completed purchases. The goal is to choose p to maximize this quantity.

A key challenge here is that the price for each bundle size determines not only the instantaneous expected revenue (revenue per sale times probability of purchase), but also how quickly the vendor’s capacity B is depleted, and hence if future customers are served. Therefore, purchase decisions are coupled across customers, and revenue cannot be decomposed into independent per-customer terms.

Hardness results. We first show that this problem is NP-hard via a reduction from SUBSET SUM (Thm. 1). This motivates “simple” pricing schemes. We first study if the natural and most common practice of *linear pricing*, (i.e., $p(n) = kn$ for n GPUs for suitably chosen k) can approximate the optimal revenue OPT: we answer this in the negative, showing that in the worst-case, it could be a factor $2/\log B$ of the optimum (Thm. 3). This lower bound formalizes the failure mode of linear pricing: a single slope cannot simultaneously extract revenue from small bundles and maintain demand from large bundles when the type-specific revenue curves have different shapes.

Methods. We then study three methods for this problem. (i) First, we show that via a suitably designed discretization (cover) of the price curves, we can achieve an additive $O(\epsilon)$ -approximation to OPT, under Lipschitz value distributions. However, this has exponential complexity in the number of feasible bundle sizes which motivates our next two methods. (ii) Second, we design a tractable posted price menu based on the *ex ante* benchmark, where we relax the ex-post capacity constraint to an ex-ante one and solve the resulting convex program under Myerson’s regularity [26]. Posting the induced menu for m customers achieves a $(1 - \epsilon)/2$ approximation to OPT when the maximum request size satisfies $\bar{N} \leq \epsilon B$ (Thm. 4). Moreover, this pricing menu can be adapted to obtain a constant $1/6$ approximation beyond this small job regime. (iii) Our third method reduces revenue maximization to a virtual knapsack prophet problem. Applying known techniques [25, 12] yields a posted menu with a $1/3$ approximation when $\bar{N} \leq B/2$ and a $1/5$ approximation in general (Thm. 5). Table 1 in Appendix A summarizes the key results of all methods.

88 A primary motivation for studying the sales setting is that it lays the foundation for GPU rentals in §3.
 89 While the knapsack prophet approach is “natural” for sales, our methods for rentals build on the *ex*
 90 *ante* relaxation, similar to our second method above.

91 **GPU rentals (§3).** Here, a vendor operates a reusable GPU cluster with capacity B over a finite
 92 horizon T . Customers arrive sequentially over time. Each customer needs a specified number of
 93 GPUs for a specified duration and has a private value for completing that job, with these types
 94 drawn from a known distribution. The vendor commits to an anonymous, time-indexed pricing curve
 95 $p_t(\cdot, \cdot)$, mapping each GPU requirement and duration pair to a price, before observing the realized
 96 customer type at time t . A customer submits her job if her value exceeds the posted price for her
 97 requested bundle-duration pair, and the job is served only if enough GPUs are available at its arrival
 98 time. If served, the job pays the posted price, occupies the requested GPUs for its duration, and
 99 then releases them back to the cluster. The vendor’s expected revenue is the sum of prices collected
 100 from served jobs, and the goal is to choose the pricing curves to maximize this quantity. The rental
 101 model generalizes sales by adding temporal coupling: accepting a job reduces capacity not only now,
 102 but throughout its duration. Thus pricing must account for the future blocking caused by large or
 103 long-running jobs.

104 **Hardness results.** In this setting, the exact dynamic program (DP) must track the residual duration
 105 profile of active jobs. Its state space grows exponentially with the budget B and the maximum
 106 duration, making the exact DP computationally intractable in general (Appendix D.6). As in the sales
 107 setting, we first evaluate whether a simple linear pricing rule can approximate the optimal revenue.
 108 The natural baseline for rentals is GPU-time pricing, $p(n, \lambda) = kn\lambda$, where a single rate k is charged
 109 per GPU per time unit. We show that this baseline can be even more fragile than linear pricing for
 110 sales with worst case revenue ratio is at most $\min \left\{ \frac{2}{\lfloor \epsilon B \rfloor}, \frac{2}{\log B} \right\}$ (Thm. 6).

111 **Methods.** To obtain tractable duration-aware policies, we extend the *ex ante* benchmark from the
 112 sales setting to a time-expanded convex program. The decision variables specify ex-ante service
 113 probabilities for each time, bundle size, and duration, while the constraints enforce expected capacity
 114 separately in each time slot. This slot-coupled relaxation upper bounds the optimal dynamic posted-
 115 pricing policy. We then convert it into two posted-pricing policies. (i) First, we design *Throttled*
 116 *CP-Menu Posted Pricing* (TCMPP), which posts the convex-program menu only in a random subset
 117 of periods, creating slack to control blocking. When $\bar{N} \leq \epsilon B$, TCMPP achieves a $\frac{1-\epsilon}{4}$ approximation
 118 (Thm. 7). (ii) Second, we solve a slackened convex program that reserves an η fraction of capacity
 119 in every slot, and then post the resulting menu without random throttling. This method achieves
 120 a $(1 - \delta_\eta)(1 - \eta)$ approximation when $\bar{N} \leq \eta B$ and can improve on TCMPP in the very small-
 121 job regime ($\frac{\bar{N}}{B} \lesssim 0.12$) by controlling blocking without random throttling (Thm. 8). Table 2 in
 122 Appendix A summarizes the key results of all methods.

123 **Simulations.** In Appendix C, we corroborate our findings with simulations, showing that our ex-ante
 124 CP based price menus outperform linear pricing in both the sales and rental settings.

125 **Related work.** Our work connects to three lines of research: finite capacity revenue management,
 126 prophet inequality, and allocation and pricing for reusable resources. Classical revenue management
 127 models study dynamic pricing under stochastic demand and resource constraints [14, 24, 3, 30, 18, 10,
 128 2], while mechanism-design approaches use virtual values to design revenue-optimal or approximately
 129 optimal posted prices [26, 6, 7]. Prophet inequality and threshold pricing techniques provide constant-
 130 factor guarantees for online allocation and knapsack prophet problems [22, 1, 8, 25, 12, 31]. Work
 131 on reusable resources often assumes unit demand, common and constant durations, large capacity
 132 asymptotics, or linear pricing rules [28, 11, 32, 13, 15, 9, 16, 23, 4, 27, 19], whereas we study
 133 finite-capacity GPU rentals with heterogeneous multi-unit demands and heterogeneous job durations,
 134 and develop duration-aware posted-price policies from slot-coupled ex-ante relaxations. We provide
 135 a detailed discussion of related work in Appendix B.

136 2 GPU Sales

137 We begin with a setting where the vendor “sells” GPUs. In addition to being interesting in its own
 138 right, this simpler setting will allow us to build fundamental intuitions about optimal pricing for GPU
 139 rentals. In our setting, the vendor has a finite number of GPUs. Single-minded buyers, whose types
 140 (valuations and resource demands) are unknown to the vendor, arrive sequentially. The seller must *a*

141 *a priori* choose a revenue-optimal price, with knowledge only of the type distribution of the buyers.
 142 Customers are willing to make a purchase only if they achieve non-negative utility, and the purchase
 143 goes through only if there are enough GPUs remaining in the vendor’s inventory. Unlike when there
 144 is an infinite supply of GPUs, the problem is not separable across different sale amounts N , as the
 145 chosen price $p(N)$ affects not only the probability of sale for a customer requiring N GPUs, but also
 146 the availability of GPUs for future sales. Thus, even in the sales setting, the allocation process is
 147 dynamically coupled through the common capacity. We now formally introduce the model.

148 **Model.** There are m *single-minded* customers who arrive sequentially. Customer i ’s type is a tuple
 149 (V_i, N_i) representing the number of GPUs $N_i \in \mathbb{N}$ she needs and her value $V_i \in \mathbb{R}_+$. We will assume
 150 that each type (V_i, N_i) is drawn from a *known* distribution D . The vendor has a budget of B GPUs
 151 and posts an anonymous pricing curve $p : [B] \rightarrow \mathbb{R}_+$ without observing any of the realized types.
 152 Let $\bar{N}$ denote the maximum possible value for N_i , i.e., $\text{supp}(N_i) \subset [\bar{N}]$ for all $i \in [m]$. Customer i
 153 is willing to buy if $V_i \geq p(N_i)$. Let $a_i = \mathbb{1}\{V_i \geq p(N_i)\}$ indicate if the customer is willing to make
 154 a purchase, based on her valuation and price. We will let U_{i-1} be the total resource usage before
 155 customer i arrives and b_i to indicate if an actual purchase was made, based on resource availability;
 156 they are defined recursively as follows:

$$U_{i-1} = \sum_{j < i} b_j N_j, \quad b_i = a_i \mathbb{1}\{U_{i-1} + N_i \leq B\}.$$

157 Finally, the expected revenue $\text{Rev}(p)$ of a pricing curve p and the optimal revenue are defined as:

$$\text{Rev}(p) = \mathbb{E}_{(v_i, n_i) \sim D} \left[\sum_{i=1}^m b_i p(N_i) \right], \quad \text{OPT} = \max_p \text{Rev}(p). \quad (1)$$

158 **Hardness result.** Our first result shows that, even in this simpler setting, computing the optimal
 159 benchmark exactly is already computationally intractable.

160 **Theorem 1** (NP-hardness of optimal pricing). *The optimal pricing problem in (1) is NP-hard.*

161 Motivated by this hardness result, we develop algorithms to approximate OPT. First, we develop
 162 discretized price curves, which allow us to approximate OPT to an arbitrary precision ϵ , but are still
 163 computationally expensive to be prohibitive in practice. We then develop simpler algorithms under
 164 additional mild assumptions, with constant factor guarantees.

165 2.1 Discretized price curves

166 A direct way to approximate the optimal menu is to discretize each coordinate $p(n)$ to a finite grid.
 167 This converts the problem into a finite (although potentially very large) search problem. We prove
 168 that under a Lipschitz-like condition, this can achieve near optimal revenue. To state this formally,
 169 for each $n \in [\bar{N}]$, let $F_n(x) = \Pr[V \leq x \mid N = n]$ be the conditional cumulative density function
 170 (CDF) of the valuations. For $\epsilon \in (0, 1)$, let $G_\epsilon = \{0, \epsilon, 2\epsilon, \dots, \lfloor 1/\epsilon \rfloor \epsilon\}$.

171 **Theorem 2** (Approximation via discretized price curves). *Suppose that $\text{supp}(V_i) \in [0, 1]$, and that
 172 the conditional CDFs $\{F_n\}_{n \in [\bar{N}]}$ are L -Lipschitz. Then, for all $\epsilon > 0$, there exists a set of prices Q_ϵ
 173 such that, (i) $|Q_\epsilon| \in O(\epsilon^{-\bar{N}})$, (ii) $p'(n) \in G_\epsilon$ for all $p' \in Q_\epsilon$ and $n \in [\bar{N}]$, and, (iii) for all pricing
 174 curves p , there exists $p' \in Q_\epsilon$ such that $\text{Rev}(p') \geq \text{Rev}(p) - (B + mL)\epsilon$.*

175 Theorem 2 suggests the following straightforward procedure to obtain an $O((B + mL)\epsilon)$ approx-
 176 imation to OPT: compute $\text{Rev}(p')$ for each pricing curve p' in Q_ϵ , and choose the one with the
 177 highest. On the positive side, the optimal price can be approximated to arbitrary ϵ_0 . However, the
 178 size of the discretization and hence the runtime is exponential in $\bar{N}$.

179 2.2 Limits of linear pricing

180 Since achieving an arbitrary approximation is computationally prohibitive for practical use cases, it is
 181 natural to ask if computationally simpler pricing curves can approximate OPT. A common approach,
 182 used in practice for GPU sales is linear pricing, i.e., pricing each GPU at some $k > 0$ so that the
 183 pricing curve takes the form $p(n) = kn$. However, the next theorem shows that optimizing over
 184 linear price curves does not admit a constant factor approximation guarantee in the worst case.

185 **Theorem 3** (Worst-case lower bound for linear pricing). *Suppose the platform is restricted to the*
 186 *family $\{p_k^{\text{lin}}\}_{k \in \mathbb{R}_+}$, where $p_k^{\text{lin}}(n) = kn$. Then there exists a distribution D over customer types such*
 187 *that $\bar{N} \leq B/2$ and*

$$\sup_{k \geq 0} \frac{\text{Rev}(p_k^{\text{lin}})}{\text{OPT}} \leq \frac{2}{\log B}.$$

188 Note that both OPT and Rev, depend on the distribution D , which we have suppressed for simplicity.
 189 The key limitation of linear pricing is that a single scalar slope parameter must govern all bundle sizes
 190 simultaneously. If the size dependent willingness-to-pay profile $n \mapsto v(n)$ is sublinear or is even
 191 constant in the worst case, then a large unit price k extracts reasonable payment from small bundles
 192 but quickly prices out large bundles, because kn eventually outgrows $v(n)$. By contrast, a small unit
 193 price keeps larger bundles feasible but charges too little whenever trade occurs. Thus, linear pricing
 194 cannot adapt to curvature in the size-specific revenue opportunities: choosing k to improve extraction
 195 on small bundles worsens acceptance on large bundles, while choosing k to preserve acceptance on
 196 large bundles weakens revenue everywhere else.

197 The above result motivates our next approach which remains computationally tractable without
 198 collapsing the price curve to a single parameter.

199 2.3 A scalable ex-ante convex-programming benchmark

200 We now outline our main method for this section, and on which our method for the rental model in §3
 201 will be based on. We will first formally state our approach and then explain our intuitions. For a type
 202 distribution D (recall that an agent’s type $(V, N) \sim D$) let us define:

$$w_n = \Pr[N = n], \quad S_n(x) = \Pr[V \geq x \mid N = n], \quad R_n(x) = xS_n(x).$$

203 Here, w_n is the probability that the GPU demand is n , $S_n(x)$ is the probability that the valuation is at
 204 least x when the GPU demand is $N = n$, and $R_n(x)$ is the expected revenue from a *single* customer
 205 whose resource demand is n , if we set the price to be $p(n) = x$. Therefore, if we had no capacity
 206 constraint, the expected revenue of a price curve p would be $m \cdot \sum_{n \in \mathcal{N}} w_n R_n(p(n))$. Moreover, the
 207 expected resource usage if we had no capacity constraint will be $m \sum_{n \in [\bar{N}]} w_n n S_n(p(n))$. We will
 208 consider the following *ex ante* relaxation to (1), where we maximize the expected revenue subject to
 209 the constraint that the expected resource usage is at most B :

$$\max_p m \cdot \sum_{n \in [\bar{N}]} w_n R_n(p(n)) \quad \text{s.t.} \quad m \sum_{n \in [\bar{N}]} w_n n S_n(p(n)) \leq B. \quad (2)$$

210 Let p^{EA} and EA^* respectively denote the solution and maximum value to (2). The *ex-ante benchmark*
 211 EA^* can be viewed as an upper bound on OPT since every *ex post* feasible policy satisfies the *ex*
 212 *ante* constraint in expectation. The key advantage to solving (2) is that under Myerson’s regularity¹,
 213 it is tractable as it can be solved via a convex program (CP). Our next theorem, which is the main
 214 result of §2, shows that while p^{EA} is the optimal solution to (2), it also gives a constant-factor
 215 approximation to OPT in (1). The proof is provided in Appendix D.4.

216 **Theorem 4** (Guarantee of ex-ante benchmark). *Assume that the conditional distribution $V \mid N = n$*
 217 *satisfies Myerson’s regularity for each n . Then, if $\bar{N} \leq \epsilon B$, then using the posted price p^{EA} satisfies*

$$\text{Rev}(p^{\text{EA}}) \geq \frac{1 - \epsilon}{2} \text{OPT}.$$

218 *Moreover, let $p_{\max} = \max_n \frac{p^{\text{EA}}(n)}{n}$. Then we have the additive approximation on the optimal revenue*

$$\text{Rev}(p^{\text{EA}}) \geq \text{OPT} - p_{\max} \sqrt{\bar{N}B} - p_{\max} \bar{N}.$$

219 In the first result of Theorem 4, when the job sizes are small relative to the vendor’s capacity (which
 220 is a very natural assumption in many markets), we obtain a stronger guarantee on the approximation.
 221 Note that even when $\bar{N}$ is small, there could be a large demand if the number of customers m is large.

¹Precisely, let F_n and f_n denote the CDF and PDF of the distribution of the valuations V_i given that $N_i = n$. Let
 $\phi_n(v) = v - \frac{1 - F_n(v)}{f_n(v)}$ be the virtual value. Myerson’s regularity [26] stipulates that $\phi_n(\cdot)$ be non-decreasing for every n .

222 *Proof outline and intuition.* The ex-ante menu p^{EA} is chosen to maximize revenue while keeping total
 223 expected demand within the capacity B . Before any early arrival, the expected used capacity is still
 224 well below B , and Markov’s inequality implies that the customer is feasible with constant probability.
 225 Since $\bar{N} \leq \epsilon B$, whenever at most $(1 - \epsilon)B$ capacity has already been used, every buying customer
 226 can still be served. Thus the ex-ante menu loses only a constant factor from realized blocking, while
 227 the ex-ante benchmark itself upper bounds OPT, yielding the stated $\frac{1-\epsilon}{2}$ approximation.

228 The additive bound provides a complementary refinement. After normalizing valuations so that
 229 $\text{supp}(V_i) \subseteq [0, 1]$, the natural revenue scale is B , and the relative revenue loss is $O(\sqrt{\epsilon})$. Thus, the
 230 ex-ante menu p^{EA} becomes increasingly close to optimal as the maximum job size $\bar{N}$ becomes small
 231 relative to the capacity B . Finally, in Appendix D.4.2 we show that there exists an alternative pricing
 232 strategy p^{split} , which builds on p^{EA} , that achieves a $\frac{1}{6}$ approximation for any value for $\bar{N}$. Due to
 233 space constraints, we outline the algorithm in Appendix D.4.2.

234 2.4 Approximation via knapsack prophet inequalities

235 Our final approach for this problem leverages knapsack prophet inequalities to obtain an alternative
 236 posted price menu. Due to space constraints, we defer the details of this approach to Appendix D.5.
 237 This gives us the following theorem.

238 **Theorem 5** (Virtual knapsack prophet). *Assume each conditional distribution $V \mid N = n$ is regular.*
 239 *Then there exists a posted price menu p^{VKP} such that, when $\bar{N} \leq B/2$, we have*

$$\mathbb{E}[\text{Rev}(p^{\text{VKP}})] \geq \frac{1}{3} \text{OPT}.$$

240 *Moreover, for general $\bar{N}$, this menu satisfies $\mathbb{E}[\text{Rev}(p^{\text{VKP}})] \geq \frac{1}{5} \text{OPT}$.*

241 This knapsack prophet approach is arguably natural for the sales setting. However, compared with
 242 this reduction based approach, the ex-ante menu p^{EA} has a stronger guarantee when the demand is
 243 relatively small and has the additional advantage of an additive approximation guarantee. Moreover,
 244 a key motivation for studying sales is that it lays the foundation for GPU rentals, and our *ex ante*
 245 relaxation method extends more readily to the rental setting.

246 3 GPU rentals

247 While the sales model captures a one shot sales problem, many GPU markets, are fundamentally
 248 *rental* markets. A job occupies GPUs for a finite amount of time and releases them, so the vendor
 249 must choose the price as a function of the bundle size and job length. Here, the vendor has a finite
 250 number of GPUs, and customers arrive sequentially. On each round, the vendor chooses a price menu,
 251 without knowledge of the customer’s type (value, resource demand, and job duration). Customers are
 252 willing to make a purchase only if they have non-negative utility and the purchase goes through only
 253 if there are enough resources left. If a purchase is made, the customer occupies the GPUs for the
 254 duration of their job before releasing it for future customers. We now formally introduce the model.

255 **Model and objective.** We consider a sequential arrival problem with time horizon T . On time t ,
 256 an customer with type $(V_t, N_t, \Lambda_t) \sim D_t$ arrives, where $N_t \in \{1, \dots, \bar{N}\}$ is the number of GPUs
 257 requested, $\Lambda_t \in \{1, \dots, L_{\max}\}$ is the job duration, and $V_t \in \mathbb{R}_+$ is the customer’s value for being
 258 served for that bundle-duration pair. We will assume that the joint distributions $\{D_t\}_{t=1}^T$ are known
 259 ahead of time to the vendor, but the type (V_t, N_t, Λ_t) on round t is realized only after the platform
 260 posts an anonymous menu $p_t : [B] \times [L_{\max}] \rightarrow \mathbb{R}_+$. The customer will submit their job if and only
 261 if $V_t \geq p_t(N_t, \Lambda_t)$; let $a_t = \mathbb{I}\{V_t - p_t(N_t, \Lambda_t) \geq 0\}$ denote the submission indicator. If the request
 262 is accepted, it occupies N_t GPUs in slots $t, t+1, \dots, t+\Lambda_t-1$. We will let U_t be the total resource
 263 when the customer at time t arrives and b_t to indicate if an actual purchase was made, based on
 264 resource availability; they are defined recursively as follows:

$$U_t = \sum_{\tau=1}^{t-1} b_\tau N_\tau \mathbf{1}\{t - \tau + 1 \leq \Lambda_\tau\}, \quad b_t = a_t \cdot \mathbf{1}\{U_t + N_t \leq B\}$$

Hence, the effect of a purchase propagates to future periods through the duration Λ_t . The total expected revenue Rev of a sequence of posted menus $\{p_t\}_{t=1}^T$, and the optimal revenue OPT_D are:

$$\text{Rev}(\{p_t\}_{t=1}^T) = \mathbb{E} \left[\sum_{t=1}^T b_t p_t(N_t, \Lambda_t) \right], \quad \text{OPT} = \max_{\{p_t\}_{t=1}^T} \text{Rev}(\{p_t\}_{t=1}^T). \quad (3)$$

The rental model inherits the challenges of the sales model but is substantially more difficult. The price menu is higher-dimensional, since each customer is characterized by both multi-unit GPU demand and job duration, and feasibility constraints continue to couple decisions across arrivals. The key additional difficulty is temporal coupling. In the sales model, the system state is summarized by a single nonincreasing scalar: the number of GPUs remaining. In the rental model, an accepted job occupies capacity over several future slots before releasing it, so the current remaining capacity no longer fully characterizes the system state. Therefore the vendor must instead reason about the future occupancy profile, and duration-aware prices must account for GPU bundle size, duration, and expected congestion over the job's occupancy interval.

The rental model as a dynamic program (DP). The rental model admits an exact DP formulation in which the state at each time step encodes the remaining duration of all active jobs. While this DP is useful for understanding the opportunity cost of accepting a job, its state space grows combinatorially in both B and $L_{\max}$, making it prohibitive to solve in all but the most trivial cases. Due to space constraints, and as this DP is not our main focus, we defer the full formulation to Appendix D.6.

3.1 Limits of linear pricing

The sales case already showed that collapsing an unrestricted menu to a single linear slope can incur substantial worst case loss. In the rental model, the same concern only becomes sharper, since the pricing rule must compress heterogeneity in both bundle size and duration into one scalar parameter. For instance, using a single rate per unit of GPU-time price menu of the form $p_t^{\text{lin}}(n, \lambda) = k_t n \lambda$, while common in real-world clouds, can be far from optimal even when every job is small relative to capacity. In particular, the following theorem shows that the best linear GPU-time price can capture only a $\min\{2/\lfloor \epsilon B \rfloor, 2/\log B\}$ fraction of the optimal rental revenue in the worst case.

Theorem 6 (Worst-case lower bound for linear duration pricing). *Suppose the platform is restricted to linear GPU-time pricing, where $p_t^{\text{lin}}(n, \lambda) = k_t n \lambda$ for some $k_t \geq 0$. Then there exists an instance with stationary distribution D over customer types such that $\bar{N} \leq \epsilon B$ and*

$$\sup_{\{k_t \geq 0\}_{t=1}^T} \frac{\text{Rev}(\{p_t^{\text{lin}}\}_{t=1}^T)}{\text{OPT}} \leq \min \left\{ \frac{2}{\lfloor \epsilon B \rfloor}, \frac{2}{\log B} \right\}.$$

3.2 A slot-coupled ex-ante benchmark and TCMPP

Analogous to the sales case, the idea of price menu based on ex-ante benchmark also applies here. However, durations fundamentally change the geometry of the corresponding relaxation. Since an accepted job consumes capacity in every slot², it spans, it must track expected occupancy slot by slot, so we have T expected capacity constraints rather than a single constraint. This leads to a slot-coupled ex-ante benchmark that preserves type- and time-dependent pricing flexibility without reverting to the full dynamic program. For each time t , write

$$w_{t,n,\lambda} = \Pr[N_t = n, \Lambda_t = \lambda], \quad F_{t,n,\lambda}(p) = \Pr[V_t \leq p \mid N_t = n, \Lambda_t = \lambda].$$

Let $x_{t,n,\lambda} \in [0, 1]$ denote an ex-ante service probability for type (n, λ) at time t . Under regular continuous distributions, the acceptance probability x can be implemented by the deterministic price $p_{t,n,\lambda}(x) = F_{t,n,\lambda}^{-1}(1 - x)$, and the corresponding revenue curve is $R_{t,n,\lambda}(x) = x F_{t,n,\lambda}^{-1}(1 - x)$. Regularity implies that $R_{t,n,\lambda}$ is concave in x , so optimizing over service probabilities yields a convex program. We therefore consider the slot coupled ex-ante benchmark:

$$\begin{aligned} \text{UB}^* := & \max_{0 \leq x_{t,n,\lambda} \leq 1} \sum_{t=1}^T \sum_{n,\lambda} w_{t,n,\lambda} R_{t,n,\lambda}(x_{t,n,\lambda}) \\ \text{s.t.} & \sum_{\tau,\lambda: \tau \leq t \leq \tau + \lambda - 1} \sum_n w_{\tau,n,\lambda} n x_{\tau,n,\lambda} \leq B, \quad t \in [T]. \end{aligned} \quad (4)$$

²We use the term slot to refer to one period of the horizon with B units of GPU capacity.

The constraints in (4) is the key extension beyond the sales case: the variable $x_{t,n,\lambda}$ appears in every slot constraint covered by a job of duration λ . The objective in (4) maximizes the total expected benchmark revenue obtained by assigning each time-type pair (t, n, λ) an ex-ante service probability $x_{t,n,\lambda}$ and evaluating that probability through the concave revenue curve $R_{t,n,\lambda}$. For each slot t , (4) adds up the expected GPU occupancy contributed by every request that would still be active in slot t . This creates T constraints for the duration case rather than through a single aggregate expected demand constraint.

Similar to the method in §2, this relaxation plays two roles. Algorithmically, it is polynomial-time solvable under regularity. Conceptually, it is an upper bound on any dynamic posted-price policy. We now turn UB^* into a posted-price policy. Let x^* be an optimal solution to (4), and define the corresponding CP menu

$$p_t^*(n, \lambda) := F_{t,n,\lambda}^{-1}(1 - x_{t,n,\lambda}^*).$$

If we posted this menu deterministically at every time, congestion fluctuations may still create substantial blocking. Our first algorithm therefore keeps the same benchmark induced menu structure but adds random throttling to control temporal congestion.

Throttled CP-Menu Posted Pricing (TCMPP). Fix $\gamma \in (0, 1)$. At each time t , independently draw $O_t \sim \text{Bernoulli}(\gamma)$. If $O_t = 0$, post the null menu $p_t(\cdot, \cdot) \equiv +\infty$. If $O_t = 1$, post the deterministic CP menu $p_t^*(\cdot, \cdot)$ and accept a buying customer if and only if the request is feasible.

Theorem 7 (Guarantee of TCMPP). *Assume that for every (t, n, λ) the conditional value distribution $V_t \mid (N_t = n, \Lambda_t = \lambda)$ is regular, and $\bar{N} \leq \epsilon B$. Then TCMPP satisfies*

$$\mathbb{E}[\text{Rev}_{\text{TCMPP}}] \geq \gamma \left(1 - \frac{\gamma}{1 - \epsilon}\right) UB^* \geq \gamma \left(1 - \frac{\gamma}{1 - \epsilon}\right) \text{OPT}.$$

Optimizing over $\gamma \in (0, 1 - \epsilon)$ gives $\gamma^* = (1 - \epsilon)/2$ and therefore $\mathbb{E}[\text{Rev}_{\text{TCMPP}}] \geq \frac{1 - \epsilon}{4} \text{OPT}$.

Proof outline and intuition. The proof has two main steps. First, we show that the slot-coupled ex-ante CP is a valid upper bound on the OPT. Any adaptive policy induces ex-ante acceptance probabilities $x_{t,n,\lambda}^\pi$ by averaging over its history dependent decisions. Pathwise feasibility implies that these variables satisfy the slot-wise capacity constraints in expectation, while concavity of the quantile revenue curves allows Jensen’s inequality to upper bound the policy’s expected revenue by the CP objective. It remains to show that TCMPP recovers a constant fraction of this ex-ante benchmark. To do so, we couple TCMPP with a tentative process that ignores blocking: at each time t , the CP menu is posted with probability γ , and every submitted job is tentatively accepted. This tentative process earns exactly γUB^* in expectation, and the CP constraints imply that the expected tentative occupied resource in every slot is at most γB . Since $N \leq \epsilon B$, a current tentative job is feasible whenever the prior tentative occupied resource is at most $(1 - \epsilon)B$. Markov’s inequality therefore bounds the blocking probability by $\gamma/(1 - \epsilon)$. Using the independence between the current tentative sale and the past tentative load, the realized revenue is at least $\gamma \left(1 - \frac{\gamma}{1 - \epsilon}\right) UB^*$. The full proof is provided in Appendix D.8

TCMPP is attractive because it is computationally tractable and, unlike linear pricing, retains the flexibility of a rich menu, allowing it to adapt to heterogeneity in both bundle size and duration. The randomized throttling step creates slack against temporal coupling across slots and yields a clean constant-factor guarantee. However throttling is deliberately conservative: by skipping a γ fraction of periods, it sacrifices revenue even when realized congestion is mild. This motivates the next construction, which moves the slack directly into the constraints of the ex-ante CP benchmark to reserve capacity in every slot without random skipping.

3.3 Slackened ex-ante CP and posted pricing

Our second policy removes random throttling and instead solves a *slackened* ex-ante program that reserves a fixed fraction of capacity in every slot. For a parameter $\eta \in (0, 1)$, we change the constraints in (4) as

$$\sum_{\tau, \lambda: \tau \leq t \leq \tau + \lambda - 1} \sum_n w_{\tau, n, \lambda} n x_{\tau, n, \lambda} \leq (1 - \eta)B \quad t \in [T]. \quad (5)$$

Let $x^{\eta,*}$ be an optimal solution and post the corresponding deterministic prices

$$p_t^\eta(n, \lambda) := F_{t,n,\lambda}^{-1}(1 - x_{t,n,\lambda}^{\eta,*}).$$

The policy π^η posts this menu whenever a type is feasible and otherwise posts $+\infty$ (not available) for that type. The key idea is that the slackened CP gives up an η fraction of the ex-ante capacity so that the realized occupancy process has room to fluctuate without random throttling. The next theorem quantifies that tradeoff.

Theorem 8 (Slackened CP guarantee). *Assume the same regularity conditions as in Theorem 7, if $\bar{N} \leq \eta B$, then the expected revenue of π^η satisfies*

$$\text{Rev}(\pi^\eta) \geq (1 - \delta_\eta)U^\eta \geq (1 - \delta_\eta)(1 - \eta) \text{OPT},$$

where $\delta_\eta = \exp\left(-\frac{(\eta B - \bar{N})^2}{2\bar{N}\left(B + \frac{\eta B - \bar{N}}{3}\right)}\right)$.

Proof intuition. The guarantee follows from two ingredients. First, since the revenue curves are concave, shrinking the slot-wise CP constraints by a factor $(1 - \eta)$ reduces the benchmark value by at most the same multiplicative factor. Second, the reserved slack ensures that tentative occupancy remains below capacity with high probability. A Bernstein bound controls the blocking probability by δ_η , so only a δ_η fraction of the tentative revenue is lost due to the violation of the constraints. By placing slack directly in the benchmark constraints rather than in the online rule, it preserves the benchmark induced menu in every period and yields a stronger guarantee in the small job regime. The full proof is provided in Appendix D.10

Comparing the two guarantees yields a simple regime split. The detailed comparison shows that the slackened-CP guarantee is stronger in the small job regime, approximately when $\bar{N}/B \lesssim 0.12$, whereas TCMPP provides the stronger guarantee once jobs are no longer small relative to capacity. We defer the finer comparison, including how to tune η , to the Appendix D.11. A practically important special case is the stationary environment where $D_t = D$ and the platform wants one menu for the whole horizon. In this case, the ex-ante relaxation has only one decision variable per bundle-duration pair, so it has $O(\bar{N}L_{\max})$ variables and is computationally simpler than the full slot-coupled program. The static TCMPP menu obtains a $(1 - \epsilon)/4$ approximation against the best static menu, and the slackened static menu obtains a $(1 - \delta_\eta)(1 - \eta)$ approximation. Details are in Appendix E.

4 Simulations

We summarize the main empirical findings and defer the full experimental setup to Appendix C. The simulations are designed to test two empirical implications of the theory. First, when willingness to pay is nonlinear in GPU demand, a one-dimensional linear price loses revenue relative to a type-dependent menu. Consistent with this prediction, the CP menu outperforms the best tuned linear price across all sales instances.

Second, in the rental model, duration-aware CP policies remain effective even when exact dynamic programming is computationally infeasible. On a small instance where exact DP can be computed, the slackened static CP policy achieves revenue close to the DP benchmark, while static TCMPP also performs well. In large horizon rental experiments, slackened static CP and static TCMPP substantially outperform the best linear GPU-time price.

5 Conclusion

We study finite budget posted pricing for GPU markets with heterogeneous multi-unit demand and, in the rental regime, heterogeneous job durations. Our results show that exact pricing is computationally hard and that simple linear pricing can be highly suboptimal. To overcome these barriers, we develop tractable nonlinear posted pricing policies based on ex-ante convex programming benchmarks, virtual value reductions, and slot-coupled ex-ante relaxations. These policies achieve provable approximation guarantees for both GPU sales and GPU rentals. Future work includes learning demand distributions from data and developing posted-price policies that are robust to distributional misspecification.

References

- [1] Saeed Alaei. Bayesian combinatorial auctions: Expanding single buyer mechanisms to many buyers. *SIAM Journal on Computing*, 43(2):930–972, 2014.
- [2] Moshe Babaioff, Shaddin Dughmi, Robert Kleinberg, and Aleksandrs Slivkins. Dynamic pricing with limited supply, 2015.
- [3] Michael Benisch, James Andrews, and Norman Sadeh. Pricing for customers with probabilistic valuations as a continuous knapsack problem. In *Proceedings of the 8th international conference on Electronic commerce: The new e-commerce: innovations for conquering current barriers, obstacles and limitations to conducting successful business on the internet*, pages 38–46, 2006.
- [4] Omar Besbes, Adam N Elmachtoub, and Yunjie Sun. Static pricing: Universal guarantees for reusable resources. In *Proceedings of the 2019 ACM Conference on Economics and Computation*, pages 393–394, 2019.
- [5] Shuchi Chawla and Balasubramanian Sivan. Bayesian algorithmic mechanism design. *ACM SIGecom Exchanges*, 13(1):5–49, 2014.
- [6] Shuchi Chawla, Jason D Hartline, and Robert Kleinberg. Algorithmic pricing via virtual valuations. In *Proceedings of the 8th ACM Conference on Electronic Commerce*, pages 243–251, 2007.
- [7] Shuchi Chawla, Jason D Hartline, David L Malec, and Balasubramanian Sivan. Multi-parameter mechanism design and sequential posted pricing. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 311–320, 2010.
- [8] Shuchi Chawla, Nikhil Devanur, and Thodoris Lykouris. Static pricing for multi-unit prophet inequalities. *Operations Research*, 72(4):1388–1399, 2024.
- [9] Steven Delong, Alireza Farhadi, Rad Niazadeh, and Balasubramanian Sivan. Online bipartite matching with reusable resources. In *Proceedings of the 23rd ACM Conference on Economics and Computation*, pages 962–963, 2022.
- [10] Arnoud V den Boer and Bert Zwart. Dynamic pricing and learning with finite inventories. *Operations research*, 63(4):965–978, 2015.
- [11] John P Dickerson, Karthik A Sankararaman, Aravind Srinivasan, and Pan Xu. Allocation problems in ride-sharing platforms: Online matching with offline reusable resources. *ACM Transactions on Economics and Computation (TEAC)*, 9(3):1–17, 2021.
- [12] Paul Dutting, Michal Feldman, Thomas Kesselheim, and Brendan Lucier. Prophet inequalities made easy: Stochastic optimization by pricing nonstochastic inputs. *SIAM Journal on Computing*, 49(3):540–582, 2020.
- [13] Yiding Feng, Rad Niazadeh, and Amin Saberi. Near-optimal bayesian online assortment of reusable resources. In *Proceedings of the 23rd ACM Conference on Economics and Computation*, pages 964–965, 2022.
- [14] Guillermo Gallego and Garrett Van Ryzin. Optimal dynamic pricing of inventories with stochastic demand over finite horizons. *Management science*, 40(8):999–1020, 1994.
- [15] Xiao-Yue Gong, Vineet Goyal, Garud N Iyengar, David Simchi-Levi, Rajan Udhwani, and Shuangyu Wang. Online assortment optimization with reusable resources. *Management Science*, 68(7):4772–4785, 2022.
- [16] Vineet Goyal, Garud Iyengar, and Rajan Udhwani. Asymptotically optimal competitive ratio for online allocation of reusable resources. *Operations Research*, 2025.
- [17] Mohammad Taghi Hajiaghayi, Robert Kleinberg, and Tuomas Sandholm. Automated online mechanism design and prophet inequalities. In *AAAI*, volume 7, pages 58–65, 2007.

- 438 [18] Stefanus Jasin and Sunil Kumar. A re-solving heuristic with bounded revenue loss for network
439 revenue management with customer choice. *Mathematics of Operations Research*, 37(2):
440 313–345, 2012.
- 441 [19] Huiwen Jia, Cong Shi, and Siqian Shen. Online learning and pricing for service systems with
442 reusable resources. *Operations Research*, 72(3):1203–1241, 2024.
- 443 [20] Jiashuo Jiang, Will Ma, and Jiawei Zhang. Tight guarantees for multi-unit prophet inequalities
444 and online stochastic knapsack. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on*
445 *Discrete Algorithms (SODA)*, pages 1221–1246. SIAM, 2022.
- 446 [21] Richard M Karp, Umesh V Vazirani, and Vijay V Vazirani. An optimal algorithm for on-line
447 bipartite matching. In *Proceedings of the twenty-second annual ACM symposium on Theory of*
448 *computing*, pages 352–358, 1990.
- 449 [22] Ulrich Krenzel and Louis Sucheston. Semiamarts and finite values. 1977.
- 450 [23] Yanzhe Lei and Stefanus Jasin. Real-time dynamic pricing for revenue management with
451 reusable resources, advance reservation, and deterministic service time requirements. *Operations*
452 *Research*, 68(3):676–685, 2020.
- 453 [24] Grace Y Lin, Yingdong Lu, and David D Yao. The stochastic knapsack revisited: Switch-over
454 policies and dynamic pricing. *Operations Research*, 56(4):945–957, 2008.
- 455 [25] Brendan Lucier. An economic view of prophet inequalities. *ACM SIGecom Exchanges*, 16(1):
456 24–47, 2017.
- 457 [26] Roger B Myerson. Optimal auction design. *Mathematics of operations research*, 6(1):58–73,
458 1981.
- 459 [27] Zachary Owen and David Simchi-Levi. Price and assortment optimization for reusable resources.
460 *Available at SSRN 3070625*, 2018.
- 461 [28] Paat Rusmevichientong, Mika Sumida, and Huseyin Topaloglu. Dynamic assortment opti-
462 mization for reusable products with random usage durations. *Management Science*, 66(7):
463 2820–2844, 2020.
- 464 [29] Alberto Vera and Siddhartha Banerjee. The bayesian prophet: A low-regret framework for
465 online decision making. *Management Science*, 67(3):1368–1391, 2021.
- 466 [30] Alberto Vera, Siddhartha Banerjee, and Itai Gurvich. Online allocation and pricing: Constant
467 regret via bellman inequalities. *Operations Research*, 69(3):821–840, 2021.
- 468 [31] Qiqi Yan. Mechanism design via correlation gap. In *Proceedings of the twenty-second annual*
469 *ACM-SIAM symposium on Discrete Algorithms*, pages 710–719. SIAM, 2011.
- 470 [32] Xilin Zhang and Wang Chi Cheung. Online resource allocation for reusable resources. *arXiv*
471 *preprint arXiv:2212.02855*, 2022.

Appendix

A Summary of our results

Method	Revenue guarantee	Computational complexity
Exact optimal menu	Computes OPT exactly, but the problem is NP-hard (Thm. 1).	Intractable in general.
ε -grid discretization	$\text{Rev}(p_\varepsilon) \geq \text{OPT} - (B + mBL)\varepsilon$ (Thm. 2).	Exhaustive search over $O\left((1/\varepsilon)^{\bar{N}}\right)$ menus.
Linear sales price $p(n) = kn$	Worst-case approximation ratio at most $\frac{2}{\log B}$ (Thm. 3).	One-dimensional search over k .
Ex-ante CP menu	Small jobs: $\text{Rev}(p^{\text{EA}}) \geq \frac{1-\epsilon}{2} \text{OPT}$. General split: at least $\frac{1}{6} \text{OPT}$. Also admits the additive bound in Thm. 4.	Polynomial-time convex programming.
Virtual knapsack prophet menu	$\mathbb{E}[\text{Rev}(p^{\text{VKP}})] \geq \frac{1}{3} \text{OPT}$ when $\bar{N} \leq B/2$, and $\mathbb{E}[\text{Rev}(p^{\text{VKP}})] \geq \frac{1}{5} \text{OPT}$ in general (Thm. 5).	Polynomial-time computable thresholds/menu.

Table 1: Summary of the main methods for the sales setting. Here $\bar{N}$ denotes the maximum GPU demand. Guarantees involving ϵ use the corresponding small-job condition $\bar{N} \leq \epsilon B$

Method	Revenue guarantee	Computational complexity
Exact dynamic programming benchmark	Computes OPT_D exactly in principle for the rental setting.	Intractable except at small scales.
Linear duration price $p(n, \lambda) = kn\lambda$	Worst-case approximation ratio at most $\min\left\{\frac{2}{\lfloor \epsilon B \rfloor}, \frac{2}{\log B}\right\}$ (Thm. 6).	One-dimensional search over k .
TCMPP	$\mathbb{E}[\text{Rev}_{\text{TCMPP}}] \geq \frac{1-\epsilon}{4} \text{OPT}_D$ (Thm. 7).	Polynomial-time convex program.
Slackened CP policy	$\text{Rev}_D(\pi^\eta) \geq (1 - \delta_\eta)(1 - \eta) \text{OPT}_D$ (Thm. 8).	Polynomial-time convex program.

Table 2: Summary of the main methods for the rental setting. Here $\bar{N}$ denotes the maximum GPU demand and $L_{\max}$ denotes the maximum duration. Guarantees involving ϵ or η use the corresponding small-job condition $\bar{N} \leq \epsilon B$ and $\bar{N} \leq \eta B$.

Remark 1. The condition $\bar{N} \leq \epsilon B$ is a worst case small job assumption. If only a small fraction of the relevant revenue comes from jobs larger than ϵB , one can instead apply the policy to jobs with $n \leq \epsilon B$ and price larger jobs out of the market. The approximation then applies to the small job revenue benchmark, losing only the small revenue contribution of the excluded large jobs.

B Additional related work

Finite capacity pricing and revenue management and posted-price mechanisms. Finite capacity pricing is a classical topic in revenue management. Early work studies finite horizon dynamic pricing with stochastic demand, typically through stochastic-control or dynamic-programming formulations [14, 24]. Closely related to our setting, Benisch et al. [3] study discriminatory pricing with probabilistic valuations and an expected-supply formulation that reduces to a continuous knapsack problem. More recently, Vera et al. [30] develop a benchmark-based perspective on online allocation and pricing under resource constraints via Bellman inequalities. Re-solving and fluid-based methods make this principle algorithmic in network revenue-management settings with customer choice [18]. Another line of work studies dynamic pricing when the demand curve is initially unknown and must be learned from data [10, 2]. From the mechanism design side, Myerson’s lemma characterizes optimal revenue in single-parameter settings as expected virtual surplus [26], and algorithmic pricing and sequential posted-price mechanisms build on this virtual-value perspective [6, 7]. Our work differs from these lines by studying anonymous posted menus for heterogeneous multi-unit GPU demand under finite capacity, and by further considering a reusable resource regime in which accepted jobs occupy capacity for heterogeneous durations.

Prophet inequalities, threshold pricing, and resource allocation. Threshold based pricing rules are widely used in resource allocation problems and are closely connected to prophet inequalities. Prophet inequalities have been extensively studied in algorithmic game theory and mechanism design [7, 17, 31, 5, 20, 29]. The classical prophet inequality was introduced by Krenkel and Sucheston [22], who proved a $1/2$ -guarantee for the one-item setting. For k -item settings, Alaei [1] establish a celebrated $1 - 1/\sqrt{k+3}$ guarantee for the k -item prophet inequality under unit demand agents and Chawla et al. [8] gave a static threshold policy for small supply regime. In contrast to standard prophet settings, our model allows agents to have heterogeneous multi-unit resource demands and heterogeneous duration demands in the rental regime. More closely related to our sales model, knapsack prophet inequalities study online selection when each arriving item has both a random value and a random size and accepted items must fit in a knapsack capacity, and threshold-based policies for this setting yield constant factor guarantees [25, 12]. Ex-ante relaxations and correlation-gap arguments provide another general route from Bayesian mechanism design to implementable mechanisms [31, 1].

Allocation and pricing for reusable resources. A growing line of work studies allocation, assortment, and pricing problems for reusable resources [28, 11, 32, 13, 15, 9]. In many allocation models, the principal observes the realized request and then decides whether to accept it. By contrast, in our posted-pricing setting, the platform must commit to a price menu before observing the customer’s realized value. Delong et al. [9] assume a common deterministic duration d for all requests and design Periodic Reranking, an extension of the Ranking algorithm introduced by Karp et al. [21]. Similar greedy and ranking-based algorithms are also used by Gong et al. [15] and Goyal et al. [16], who assume that service durations depend on the resource rather than the job and that each match consumes exactly one unit of resource. In addition, Goyal et al. [16] focus on the large capacity regime. Similar structural assumptions also appear in several works on pricing for reusable resources [28, 23, 4, 27]. For example, Rusmevichientong et al. [28] study a dynamic assortment and pricing problem with reusable products and propose a provably near-optimal policy based on approximate dynamic programming. Besbes et al. [4] focus on a single reusable resource type, unit-demand customers, and state-dependent dynamic pricing in an infinite-horizon system. Jia et al. [19] study an online pricing model in which customers arrive according to a price-dependent Poisson process, each customer requests one unit, service times are price-dependent and exponentially distributed, and unserved customers wait in a queue until a unit becomes available. Overall, prior work often relies on assumptions such as unit demand, common durations, large capacity or linear pricing function. In contrast, we study finite-capacity posted pricing for reusable GPU resources under heterogeneous multi-unit resource demands and heterogeneous duration demands.

C Simulations

We design simulations to test two empirical implications of the theory. First, when willingness to pay is nonlinear in the requested GPU bundle, a one-dimensional linear price should lose revenue relative to a type-dependent posted-price menu. Second, in the duration-aware rental problem, static benchmark-guided CP policies should remain effective in large systems where exact dynamic programming is computationally infeasible. We report sales-regime results with figures and report rental-regime results only in Table 3.

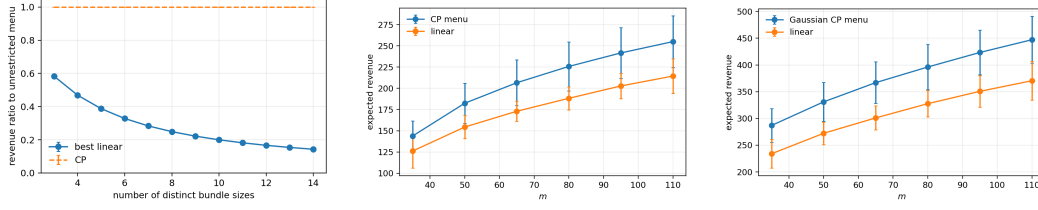
Algorithms and baselines. In the sales experiments, the CP menu from Section 2.3 is compared with the best linear price $p(n) = kn$. The slope k is selected by grid search on independent training sample paths and then evaluated on fresh sample paths. For uniform conditional values $V \mid N = n \sim \text{Unif}[0, a_n]$, the CP menu has a closed form. If x_n is the ex-ante purchase probability for type n , then $R_n(x_n) = a_n x_n (1 - x_n)$ and the KKT condition gives

$$x_n = \left[\frac{1 - \lambda n / a_n}{2} \right]_{[0, 1/2]}, \quad p(n) = a_n (1 - x_n),$$

where λ is found by bisection so that the expected accepted capacity matches the budget.

For Gaussian conditional values, the script solves the same Lagrangian relaxation numerically. If type i consumes resource r_i and $V_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$, then for each multiplier λ the type- i price is chosen from

$$p_i(\lambda) \in \arg \max_{p \geq 0} (p - \lambda r_i) \left(1 - \Phi \left(\frac{p - \mu_i}{\sigma_i} \right) \right).$$



(a): No-blocking lower-bound instance.

(b): Uniform capacity stress.

(c): Gaussian capacity stress.

Figure 1: sales-regime simulations. The CP menu preserves type-specific price flexibility, while the linear baseline is restricted to one scalar unit price. Error bars in the stochastic capacity-stress panels report one sample standard deviation; the no-blocking panel is deterministic, so its error bars have zero length.

544 This is a one-dimensional maximization for each type; we implement it by a fine price grid and again
 545 use bisection on λ . In all reported runs, the Lagrange multiplier bisection is run for 80 iterations. For
 546 Gaussian values, the per-type price maximization uses 500 grid points on $[0, \mu_i + 5\sigma_i]$, augmented
 547 with one no-sale price point; with $\sigma_i = 0.25\mu_i$, the grid spacing is about $0.0045\mu_i$.

548 In the rental experiments, we compare the exact dynamic-programming benchmark from Ap-
 549 pendix D.6, the static slackened CP policy, static TCMPP, and the best static linear GPU-time
 550 price $p(n, \lambda) = kn\lambda$. The implementable CP-based policies are built from the static ex-ante bench-
 551 mark in (25), rather than the full slot-coupled time-indexed benchmark in (4). Static TCMPP uses
 552 the corresponding static CP menu, but we do not report that menu as a separate standalone baseline.
 553 Exact DP is used only for the small benchmark instance because its state space grows as $\binom{B+L_{\max}}{L_{\max}}$.
 554 In the exact-DP benchmark, the static TCMPP offer probability γ is tuned by exact evaluation over a
 555 grid. In the large- T rental experiments, the static linear slope, the slack parameter η , and γ are tuned
 556 on independent training paths before evaluation.

557 **Sales simulation settings.** We use three sales instances. The first is a no-blocking stress test
 558 matching the lower-bound intuition for linear pricing. There are K bundle sizes $n_j = 2^{j-1}$,
 559 deterministic values $v_j = 1$, equal type probabilities, and K ranges from 3 to 14. A type-dependent
 560 menu posts price 1 for every bundle size, whereas a linear price must use a single slope and can
 561 match only a subset of the bundle sizes.

562 The second sales instance introduces stochastic demand and capacity stress. The budget is $B = 140$,
 563 bundle sizes are $n \in \{1, \dots, 18\}$, type probabilities are proportional to $n^{-0.15}$, and

$$V \mid N = n \sim \text{Unif}[0, 10n^{0.3}].$$

564 The exponent 0.3 creates a sublinear willingness-to-pay curve: large bundles have lower value per
 565 GPU than small bundles. We vary the number of arrivals over $\{35, 50, 65, 80, 95, 110\}$. Each point
 566 uses 1500 independent evaluation paths; linear prices are tuned on 500 independent training paths.
 567 Error bars report one sample standard deviation across the evaluation paths.

568 The third sales instance keeps the same budget, type probabilities, bundle sizes, and arrival counts but
 569 uses Gaussian values,

$$V \mid N = n \sim \mathcal{N}(\mu_n, \sigma_n^2), \quad \mu_n = 10n^{0.3}, \quad \sigma_n = 0.25\mu_n.$$

570 This checks that the advantage of the CP menu is not an artifact of bounded uniform value distributions.
 571 As above, each point uses 1500 independent evaluation paths, and error bars report one sample
 572 standard deviation.

573 **Rental simulation settings.** The first rental instance is a small exact-DP benchmark. We set $B = 6$,
 574 $T = 20$, $\bar{N} = 5$, and $L_{\max} = 5$. Type probabilities are uniform over the 25 size-duration pairs.
 575 Conditional values are uniform with

$$a_{n,\lambda} = 9(1 + 0.351\{\lambda = 1\})n^{0.55}\lambda^{0.35}.$$

576 The static slackened CP policy uses $\eta = 0.35$. The certified TCMPP parameter is $\gamma = (1 - \bar{N}/B)/2$,
 577 which equals $1/12$ in this instance; this value is the maximizer of the worst-case guarantee but

is overly conservative for the realized benchmark. We therefore report a tuned static TCMPP implementation, which selects $\gamma = 1$ on the exact evaluator. This benchmark is intentionally capacity stressed, so it tests whether a static menu can track the state-dependent DP value.

The second rental instance is a large-horizon uniform-value simulation with $T = 3000$ and $B = 25$. There are eight request types,

$$(n, \lambda) \in \{(20, 4), (16, 5), (10, 8), (8, 10), (5, 16), (4, 20), (2, 40), (1, 80)\}.$$

All types have identical GPU-time product $n\lambda = 80$, so a static linear GPU-time price posts the same price to every type. Conditional values are uniform on $[0, a_{n,\lambda}]$ with

$$a_{n,\lambda} \in \{4, 8, 16, 32, 64, 128, 256, 1024\},$$

and type probabilities are proportional to $a_{n,\lambda}^{-1/2}$. The slack parameter is tuned over 14 grid points in $[0, 0.65]$, static TCMPP tunes γ over 20 grid points in $[0.05, 1]$, and the static linear GPU-time slope is tuned by grid search. We use 180 training paths and 500 evaluation paths.

The third rental instance is the Gaussian analogue of the large-horizon simulation. The request types, horizon, budget, training-path count, and evaluation-path count are unchanged. Values satisfy

$$V_i \sim \mathcal{N}(\mu_i, (0.25\mu_i)^2), \quad \mu_i \in \{6, 10, 18, 34, 66, 130, 260, 1040\},$$

and type probabilities are proportional to $\mu_i^{-1/2}$. As in the uniform large- T instance, all request types have the same GPU-time product, so the static linear GPU-time price cannot distinguish high-mean and low-mean types.

Table 3: rental-regime revenue summary. Entries are revenues; the small DP column reports total expected revenue, while the two large- T columns report revenue per period.

Algorithm	Small DP	Uniform	Gaussian
exact DP	87.04 ± 0.00	—	—
slackened CP	86.28 ± 0.00	11.25 ± 1.01	29.39 ± 2.04
TCMPP	83.89 ± 0.00	10.94 ± 0.99	28.72 ± 1.97
linear GPU-time	74.92 ± 0.00	5.43 ± 0.42	14.23 ± 2.00

Results and interpretation. Figure 1 shows that the sales-regime advantage is stable across all three sales instances. In the no-blocking lower-bound instance, the linear price captures a decreasing share of menu revenue as the number of bundle sizes grows. In the uniform capacity-stress instance, the CP menu earns 255.06 ± 30.55 revenue at the largest arrival count, compared with 214.46 ± 20.50 for the best linear price. In the Gaussian capacity-stress instance, the corresponding revenues are 447.19 ± 43.66 and 370.57 ± 36.39 . These gaps arise because a single scalar unit price cannot both extract revenue from high-value small bundles and screen low-value large bundles.

Table 3 gives the rental-regime revenue comparisons. In the exact-DP benchmark, static slackened CP earns 86.28, close to the exact DP value of 87.04; tuned static TCMPP earns 83.89, and the static linear GPU-time price earns 74.92. In the two large- T experiments, the static slackened CP menu has the highest mean revenue among the implementable policies: 11.25 ± 1.01 per period in the uniform instance and 29.39 ± 2.04 per period in the Gaussian instance. Static TCMPP earns 10.94 ± 0.99 and 28.72 ± 1.97 per period, respectively. The static linear GPU-time price performs poorly in both large instances because all request types have the same GPU-time product but very different value distributions. Across all three rental experiments, tuning selects $\gamma = 1$ for static TCMPP, so the reported static TCMPP policy uses the static CP menu from (25) without additional random throttling.

610 **D Proof of Section 4**

611 **D.1 Proof of Theorem 1**

612 We reduce from SUBSET SUM. An instance consists of positive integers $a_1, \dots, a_\ell$ and a target T .
 613 Let

$$A = \sum_{j=1}^{\ell} a_j, \quad C = 2A, \quad x^* = \frac{T}{C}.$$

614 We assume without loss of generality that $1 \leq T \leq A$.

615 We construct a GPU sales instance with $m = 2$ customers drawn i.i.d. from a single distribution $\mathcal{D}$.
 616 Let

$$K = \ell + 2, \quad B = 2K.$$

617 Choose demands

$$n_{\perp} = K + 1, \quad n_j = K + 1 + j \quad (j = 1, \dots, \ell), \quad n_0 = 2K.$$

618 Thus every possible demand is strictly larger than $B/2 = K$, so at most one of the two customers
 619 can be served.

620 Let

$$\Delta = \frac{1}{16\ell C^2}, \quad v_j = 1 + j\Delta \quad (j = 1, \dots, \ell), \quad H = 7 - 8x^*.$$

621 Since $x^* \leq 1/2$, we have $H \geq 3$, and hence

$$0 < v_1 < \dots < v_\ell < H.$$

622 Define the common distribution $\mathcal{D}$ by

$$\Pr_{\mathcal{D}}[(V, N) = (H, n_0)] = \frac{1}{4},$$

623

$$\Pr_{\mathcal{D}}[(V, N) = (v_j, n_j)] = \frac{a_j}{C}, \quad j = 1, \dots, \ell,$$

624 and

$$\Pr_{\mathcal{D}}[(V, N) = (0, n_{\perp})] = \frac{1}{4}.$$

625 This is a valid distribution because

$$\frac{1}{4} + \sum_{j=1}^{\ell} \frac{a_j}{C} + \frac{1}{4} = \frac{1}{4} + \frac{A}{2A} + \frac{1}{4} = 1.$$

626 For any pricing curve, let Q be the probability that one random customer is willing to buy, and let R_1
 627 be the expected payment from one random customer before applying the capacity constraint. Since
 628 every demand exceeds $B/2$, if the first customer buys then the second cannot be served. Therefore
 629 the expected revenue from the two i.i.d. customers is

$$\text{Rev} = R_1 + (1 - Q)R_1 = R_1(2 - Q).$$

630 Define the threshold

$$R^* = \left(\frac{7}{4} - x^* \right)^2.$$

631 We show that $\text{OPT} \geq R^*$ if and only if the original SUBSET SUM instance is feasible.

632 First suppose there exists $S \subseteq [\ell]$ such that

$$\sum_{j \in S} a_j = T.$$

633 We construct a strictly increasing pricing curve that admits the special type and exactly the item types
 634 in S . For $r = 1, \dots, K$, set

$$p(r) = \frac{r\Delta}{6K}.$$

635 Set

$$p(n_{\perp}) = \frac{\Delta}{3},$$

636 and for each item type $j = 1, \dots, \ell$, set

$$p(n_j) = \begin{cases} v_j, & j \in S, \\ v_j + \Delta/3, & j \notin S. \end{cases}$$

637 Finally, set

$$p(n_0) = H.$$

638 Because $n_{\perp} < n_1 < \dots < n_{\ell} < n_0$ and

$$0 < p(1) < \dots < p(K) < p(n_{\perp}) < p(n_1) < \dots < p(n_{\ell}) < p(n_0).$$

639 Under this curve, the dummy type does not buy, the special type buys, and item type j buys if and
640 only if $j \in S$. Let

$$x = \sum_{j \in S} \frac{a_j}{C}, \quad z = \sum_{j \in S} \frac{ja_j}{C}.$$

641 Since $\sum_{j \in S} a_j = T$, we have $x = x^*$. Therefore

$$Q = \frac{1}{4} + x^*, \quad R_1 = \frac{1}{4}H + x^* + \Delta z.$$

642 Using $H = 7 - 8x^*$, we get

$$\frac{1}{4}H = \frac{7}{4} - 2x^*.$$

643 Hence

$$\text{Rev} = \left(\frac{7}{4} - x^* + \Delta z \right) \left(\frac{7}{4} - x^* \right) \geq \left(\frac{7}{4} - x^* \right)^2 = R^*.$$

644 Thus $\text{OPT} \geq R^*$.

645 Conversely, suppose the SUBSET SUM instance is infeasible. We show that no pricing curve can
646 achieve revenue at least R^* . The dummy type has value zero, so admitting it cannot increase revenue:
647 it contributes no payment and only increases the probability that the second customer is blocked.
648 Hence we ignore it in the following upper bounds.

649 First consider any pricing curve that does not admit the special type. Let $S \subseteq [\ell]$ be the admitted
650 item types, and define

$$x = \sum_{j \in S} \frac{a_j}{C}, \quad z = \sum_{j \in S} \frac{ja_j}{C}.$$

651 Then $x \leq 1/2$ and $z \leq \ell/2$. Since each admitted item type pays at most its value,

$$\text{Rev} \leq (x + \Delta z)(2 - x) \leq \frac{3}{4} + \Delta \ell.$$

652 By the choice of Δ ,

$$\frac{3}{4} + \Delta \ell = \frac{3}{4} + \frac{1}{16C^2} < \frac{25}{16}.$$

653 On the other hand, since $x^* \leq 1/2$,

$$R^* = \left(\frac{7}{4} - x^* \right)^2 \geq \left(\frac{5}{4} \right)^2 = \frac{25}{16}.$$

654 Therefore any pricing curve that does not admit the special type has revenue strictly below R^* .

655 It remains to consider pricing curves that admit the special type. Let $S \subseteq [\ell]$ be the admitted item
656 types, and define x, z as above. Since admitted types pay at most their values,

$$Q \geq \frac{1}{4} + x, \quad R_1 \leq \frac{1}{4}H + x + \Delta z.$$

Therefore

$$\text{Rev} \leq \left(\frac{1}{4}H + x + \Delta z \right) \left(\frac{7}{4} - x \right).$$

Substituting $\frac{1}{4}H = \frac{7}{4} - 2x^*$ gives

$$\text{Rev} \leq \left(\frac{7}{4} - 2x^* + x \right) \left(\frac{7}{4} - x \right) + \Delta z \left(\frac{7}{4} - x \right).$$

The first term equals

$$\left(\frac{7}{4} - x^* \right)^2 - (x - x^*)^2 = R^* - (x - x^*)^2.$$

Since the SUBSET SUM instance is infeasible, $x \neq x^*$. Both x and x^* are integer multiples of $1/C$, so

$$|x - x^*| \geq \frac{1}{C}.$$

Also, $z \leq \ell/2$ and $7/4 - x \leq 7/4$. Hence

$$\text{Rev} \leq R^* - \frac{1}{C^2} + \Delta \cdot \frac{\ell}{2} \cdot \frac{7}{4} = R^* - \frac{1}{C^2} + \frac{7}{128C^2} < R^*.$$

Thus no pricing curve achieves revenue at least R^* when the SUBSET SUM instance is infeasible.

Therefore,

$$\text{OPT} \geq R^* \iff \exists S \subseteq [\ell] \text{ such that } \sum_{j \in S} a_j = T.$$

The construction has polynomial size: the distribution has $\ell + 2$ support points, all probabilities and values are rational numbers with polynomial encoding length, and $B = 2(\ell + 2)$. Hence deciding whether $\text{OPT} \geq R^*$ is NP-hard under i.i.d. arrivals from a single common distribution $\mathcal{D}$. Consequently, computing OPT is NP-hard.

Deterministic arrival model We also give a proof for NP-hard in the deterministic arrival version of the sales problem. We consider a deterministic customer sequence, and optimizing the revenue under a deterministic customer sequence is easier (no harder than) optimizing under a customer distribution.

Given $a_1, \dots, a_m, T$, we denote $Q = m + 1$, let $L = Q^m$, and let the GPU budget be $B = T \cdot L + \sum_{i=1}^m Q^{i-1}$. Then we construct $2m$ customers, two per $i \in [m]$:

• Select- i customer:

$$n_{i,1} = a_i \cdot L + Q^{i-1}, \quad v_{i,1} = \frac{n_{i,1}}{M}.$$

• Not-select- i customer:

$$n_{i,0} = Q^{i-1}, \quad v_{i,0} = \frac{n_{i,0}}{M}.$$

Then we Choose a scaling constant $M = 2B$. Then all values $v_{i,0}, v_{i,1} \leq B/M = 1/2 \in [0, 1]$. We will prove:

$$\exists S \subseteq [m] \text{ with } \sum_{i \in S} a_i = T \iff \exists \text{ strictly increasing } p : \mathbb{N} \rightarrow [0, 1] \text{ with revenue } \geq \frac{1}{2}.$$

If Subset Sum has a solution, then revenue $\geq \frac{1}{2}$ is achievable. Assume there exists $S \subseteq [m]$ such that $\sum_{i \in S} a_i = T$. Define a price function p as follows. Let $\varepsilon = \frac{1}{4M} < \frac{1}{M}$. We first set a Start from a baseline strictly increasing function:

$$p_{\text{base}}(n) = \frac{n}{M} + \frac{\varepsilon}{2}.$$

Now modify it on the $2m$ special sizes:

• For each $i \in S$:

$$p(n_{i,1}) = \frac{n_{i,1}}{M} \quad ((i, 1) \text{ submits}), \quad p(n_{i,0}) = \frac{n_{i,0}}{M} + \varepsilon \quad ((i, 0) \text{ does not submit}).$$

684 • For each $i \notin S$:

$$p(n_{i,0}) = \frac{n_{i,0}}{M} \quad ((i, 0) \text{ submits}), \quad p(n_{i,1}) = \frac{n_{i,1}}{M} + \varepsilon \quad ((i, 1) \text{ does not submit}).$$

685 Because $\varepsilon < 1/M$, the resulting $p(\cdot)$ remains strictly increasing. For any integers $n' < n$,

$$p(n') \leq \frac{n'}{M} + \varepsilon < \frac{n' + 1}{M} \leq \frac{n}{M} \leq p(n).$$

686 So for $i \in S$, $(i, 1)$ submits; for $i \notin S$, $(i, 0)$ submits. And the total GPUs used by those submitted
687 customers is

$$\sum_{i \in S} n_{i,1} + \sum_{i \notin S} n_{i,0} = \sum_{i \in S} (a_i L + Q^{i-1}) + \sum_{i \notin S} Q^{i-1} = \left(\sum_{i \in S} a_i \right) L + \sum_{i=1}^m Q^{i-1} = TL + \sum_{i=1}^m Q^{i-1} = B.$$

688 So they fit exactly in capacity and will all be accepted. Each accepted customer pays $p(n) = n/M$,
689 hence total revenue is $\frac{B}{M} = \frac{1}{2}$.

690 **If revenue $\geq \frac{1}{2}$ is achievable, then SUBSET SUM has a solution.** Assume there exists a strictly
691 increasing $p : \mathbb{N} \rightarrow [0, 1]$ such that the platform's revenue is at least $\frac{1}{2}$. For any accepted customer j ,

$$p(n_j) \leq v_j = \frac{n_j}{M}.$$

692 Therefore, total revenue satisfies

$$\sum_{j \text{ accepted}} p(n_j) \leq \sum_{j \text{ accepted}} \frac{n_j}{M} = \frac{1}{M} \sum_{j \text{ accepted}} n_j \leq \frac{B}{M} = \frac{1}{2}.$$

693 So if revenue is $\geq \frac{1}{2}$, it must be exactly $\frac{1}{2}$, and both inequalities must hold with equality. So
694 $\sum_{j \text{ accepted}} n_j = B$ and $p(n_j) = n_j/M$ for each accepted j . Now we use the base- Q structure:

- 695 • Recall $B = TL + \sum_{i=1}^m Q^{i-1}$, where $L = Q^m$.
- 696 • Each not-select size is $n_{i,0} = Q^{i-1}$, which adds single 1 in digit position $i - 1$.
- 697 • Each select size is $n_{i,1} = a_i L + Q^{i-1}$, which adds a_i in the top digit m , plus a 1 in digit
698 $i - 1$.

699 Since the sum of accepted sizes equals B , whose lower m digits are all exactly 1, it follows that for
700 each $i \in [m]$, the accepted set must include exactly one of $\{n_{i,0}, n_{i,1}\}$. Let S be the indices i for
701 which $n_{i,1}$ is accepted. Then the top digit (the coefficient of $L = Q^m$) of the accepted-sum must
702 match the top digit of B , i.e., $\sum_{i \in S} a_i = T$. Thus Subset Sum has a solution.

703 D.2 Proof for Theorem 2

704 *Proof.* Let $J(p) = \mathbb{E}[\sum_{i=1}^m b_i p(n_i)]$ denote the expected revenue under the finite-budget acceptance
705 rule defined, where $b_i = \mathbb{I}\{v_i - p(n_i) \geq 0\} \mathbb{I}\{U_{<i} + n_i \leq B\}$.

706 Fix $\varepsilon > 0$ and define the uniform grid

$$\mathcal{P}_\varepsilon = \{0, \varepsilon, 2\varepsilon, \dots, 1\}.$$

707 For any pricing function p , let $\Pi_\varepsilon(p)$ be obtained by rounding each $p(n)$ to the nearest grid point (ties
708 arbitrary). Then

$$|\Pi_\varepsilon(p)(n) - p(n)| \leq \varepsilon/2 \quad \forall n \leq B.$$

709 We compare $J(p)$ and $J(\Pi_\varepsilon(p))$. There are two sources of change: (i) payment rounding for cus-
710 tomers who are accepted under both menus, and (ii) accept/reject changes caused by the perturbation
711 in posted prices. Since each accepted job consumes at least one GPU, at most B jobs can be accepted
712 on any realization, so the total revenue change from (i) is at most $B\varepsilon$.

713 For (ii), fix a demand size n and consider an arrival with value $V \mid N = n$. By the Lipschitz
714 assumption on F_n , the probability that the submission decision $\mathbb{I}\{V \geq p(n)\}$ differs between $p(n)$

715 and $\Pi_\varepsilon(p)(n)$ is at most $L\varepsilon$. Such a flip can change the subsequent feasible acceptances, but the total
 716 remaining revenue along any trajectory is at most B (at most B further acceptances, each paying at
 717 most 1). Over m arrivals, the expected revenue impact from these flips is therefore at most $mBL\varepsilon$.
 718 Combining the two contributions yields

$$|J(\Pi_\varepsilon(p)) - J(p)| \leq B\varepsilon + mBL\varepsilon.$$

719 Let p^* be an optimal menu. The discretized menu $\tilde{p} = \Pi_\varepsilon(p^*)$ satisfies

$$J(\tilde{p}) \geq J(p^*) - (B + mBL)\varepsilon.$$

720 To obtain an additive ε_0 -approximation in expected revenue, choose the grid step $\varepsilon = \frac{\varepsilon_0}{B(1+mL)}$.
 721 Since p^* was arbitrary, the maximizer over discretized menus achieves this guarantee. $\square$

722 D.3 Proof for Theorem 3

723 Suppose the platform is restricted to the linear pricing family $p(n) = kn$. Even when all customers
 724 $(v_i, n_i)_{i=1}^m$ are i.i.d. draws from a distribution $\mathcal{D}$, and $\max_i n_i \leq B/2$, then there exists a sequence
 725 of instances such that

$$\frac{\sup_{k \in \mathbb{R}} \mathbb{E}[\text{Rev}(p(n) = kn)]}{\text{OPT}} \leq \frac{2}{\log B}.$$

726 *Proof.* Fix an integer $T \geq 2$. Let $m = T, B = T2^T$. Define the common distribution $\mathcal{D}$ as follows.
 727 For $t \in [T]$,

$$\Pr_{\mathcal{D}}((V, N) = (1, 2^{t-1})) = \frac{1}{T}.$$

728 Because $B = T2^T$, we have $2^{T-1} \leq \frac{B}{2T} \leq \frac{B}{2}$.

729 Moreover, for any realization,

$$\sum_{i=1}^m n_i \leq m \cdot 2^{T-1} = T2^{T-1} = \frac{B}{2} < B.$$

730 Therefore the budget constraint never binds and every submitted job is accepted. Hence the realized
 731 revenue equals

$$\text{Rev}(p) = \sum_{i=1}^m p(n_i) \mathbf{1}\{v_i \geq p(n_i)\}.$$

732 Consider the pricing rule, for all n ,

$$p^*(n) = 1$$

733 *Remark:* If we additionally require $p(\cdot)$ to be strictly increasing, you can instead take, e.g., $p(n) =$
 734 $1 - 2^{-n}$, which is strictly increasing and satisfies $p(n_j) < 1$ for all j ; then every job still submits and
 735 the revenue remains $\Theta(T)$. So the separation is robust to monotonicity requirements.

736 Since every customer has value $v_i = 1$, every job submits and is accepted. Thus

$$\text{OPT} = \text{Rev}(p^*) = m = T.$$

737 Consider a linear price $p_k(n) = kn$. If $k < 0$, the platform charges negative prices and revenue is
 738 non-positive, so it suffices to consider $k \geq 0$. A job of type t with size $n = 2^{t-1}$ submits iff

$$1 \geq k2^{t-1}.$$

739 Define

$$t^* = \max\{t \in \{1, \dots, T\} : k2^{t-1} \leq 1\},$$

740 with the convention $t^* = 0$ if the set is empty.

741 Then exactly the types $t = 1, \dots, t^*$ submit. Since capacity never binds, the expected revenue equals

$$\mathbb{E}[\text{Rev}(p_k)] = m \cdot \mathbb{E}_{(V, N) \sim \mathcal{D}} [p_k(N) \mathbf{1}\{V \geq p_k(N)\}].$$

742 Because the distribution is uniform over the T types,

$$\mathbb{E}[\text{Rev}(p_k)] = T \cdot \frac{1}{T} \sum_{t=1}^{t^*} k 2^{t-1} = \sum_{t=1}^{t^*} k 2^{t-1} = k(2^{t^*} - 1).$$

743 From the definition of t^* ,

$$k 2^{t^*-1} \leq 1 \quad \Rightarrow \quad k \leq 2^{1-t^*}.$$

744 Therefore

$$\mathbb{E}[\text{Rev}(p_k)] = k(2^{t^*} - 1) \leq 2^{1-t^*}(2^{t^*} - 1) < 2.$$

745 Thus

$$\sup_{k \in \mathbb{R}} \mathbb{E}[\text{Rev}(p_k)] \leq 2.$$

746 Since $B = T 2^T$, we have $\log B = \log T + T \log 2 \leq T$ for all sufficiently large T , and hence
747 $\frac{2}{T} \leq \frac{2}{\log B}$. Combining the bounds,

$$\frac{\sup_k \mathbb{E}[\text{Rev}(p_k)]}{\text{OPT}} \leq \frac{2}{T} \leq \frac{2}{\log B}.$$

748

□

749 D.4 Proofs for Theorem 4

750 Myerson regularity implies that, for each n , the quantile revenue curve

$$r_n(q) = q F_n^{-1}(1 - q)$$

751 is concave in the sale probability q . Hence, after changing variables from prices $p(n)$ to sale
752 probabilities $q_n = S_n(p(n))$, the ex-ante benchmark can be written as

$$\max_{q_n \in [0,1]} m \sum_n w_n r_n(q_n) \quad \text{s.t.} \quad m \sum_n w_n n q_n \leq B,$$

753 which is a convex program, since it maximizes a concave objective over affine and box constraints.
754 Therefore, under a standard constraint qualification such as Slater's condition, strong duality holds.

755 For every realization,

$$\sum_{i \in A} p(n_i) = \sum_{i \in A} (p(n_i) - \lambda n_i) + \lambda \sum_{i \in A} n_i \leq \sum_i b_i (p(n_i) - \lambda n_i) + \lambda B.$$

756 Taking expectations on both sides:

$$\mathbb{E} \left[\sum_{i \in A} p(N_i) \right] \leq \lambda B + \sum_{i=1}^m \mathbb{E} [b_i (p(n_i) - \lambda n_i)] = \lambda B + m \sum_n w_n S_n(p(n)) (p(n) - \lambda n).$$

757 This is because $\mathbb{E}[b_i] = \mathbb{E}[\mathbb{I}\{v_i \geq p(n_i)\}] = \sum_n \mathbb{E}[\mathbb{I}\{v \geq p(n)\} | n] \Pr[N = n] =$
758 $\sum_n w_n S_n(p(n))$.

759 For each size n and $\lambda \geq 0$,

$$S_n(p(n)) (p(n) - \lambda n) \leq \max_{x \in [0,1]} S_n(x) (x - \lambda n).$$

760 Combining them together we have

$$\text{Rev}(p) \leq \lambda B + m \sum_n w_n \max_{x \in [0,1]} S_n(x) (x - \lambda n) \doteq \text{UB}(\lambda).$$

761 Hence, with strong duality

$$\text{Rev}(p) \leq \inf_{\lambda \geq 0} \text{UB}(\lambda) = \inf_{\lambda \geq 0} \left\{ \lambda B + m \sum_n w_n \max_x S_n(x) (x - \lambda n) \right\} = \text{EA}^*.$$

762 The sufficient conditions the strong duality holds for the optimization problem 2.

763 **D.4.1** $\bar{N} = \max_i N_i \leq \varepsilon B$

764 **At least $B/2$ GPUs remain when any agent arrives.** For customer i , $\mathbb{E}[U_{<i}] \leq \frac{i-1}{m}B$. Applying
 765 Markov's inequality:

$$\Pr[U_{<i} \geq (1-\varepsilon)B] \leq \frac{\mathbb{E}[U_{<i}]}{(1-\varepsilon)B} \leq \frac{\frac{i-1}{m}B}{(1-\varepsilon)B} = \frac{i-1}{(1-\varepsilon)m}.$$

766 As $\Pr[U_{<i} \leq (1-\varepsilon)B] = 1 - \Pr[U_{<i} \geq (1-\varepsilon)B]$, let $\Pr[U_{<i} \leq (1-\varepsilon)B] \geq 0$, then for
 767 $i \leq \lceil (1-\varepsilon)m \rceil$,

$$\Pr[U_{<i} \leq (1-\varepsilon)B] \geq \frac{(1-\varepsilon)m - i + 1}{(1-\varepsilon)m}.$$

768 Since every job size $n_i \leq \varepsilon B$, the event $U_{<i} \leq (1-\varepsilon)B$ guarantees that agent i 's job fits if offered
 769 and accepted.

770 Fix n , for an $i \leq \lceil (1-\varepsilon)m \rceil$, the expected revenue contribution for this customer i is at least

$$\Pr[U_{<i} \leq (1-\varepsilon)B] \times \Pr[\text{submit} \mid N = n] \times p^{\text{EA}}(n) \geq \frac{(1-\varepsilon)m - i + 1}{(1-\varepsilon)m} S_n(p^{\text{EA}}(n)) p^{\text{EA}}(n).$$

771 Take expectation over N_i and sum over agents:

$$\mathbb{E}[\text{revenue}] \geq \sum_{i \leq \lceil (1-\varepsilon)m \rceil} \sum_n \frac{(1-\varepsilon)m - i + 1}{(1-\varepsilon)m} w_n R_n(p^{\text{EA}}(n)) \geq \frac{1}{2}(1-\varepsilon)m \sum_n w_n R_n(p^{\text{EA}}(n)) = \frac{1-\varepsilon}{2} \text{EA}^*.$$

772 Thus, the $(1-\varepsilon)/2$ -approximation guarantee holds whenever job sizes satisfy $\max_i n_i \leq \varepsilon$.

773 Separating the analysis into two cases at the threshold $B/2$ yields a better $1/6$ bound than directly
 774 applying the $(1-\varepsilon)/2$ guarantee when ε is large.

775 **D.4.2 General $\bar{N}$**

776 Here there can be “big” jobs $n > B/2$ (at most one can fit) and “small” jobs $n \leq B/2$. Let OPT^{sm}
 777 be the optimal revenue in the instance that ignores all big jobs, and let OPT^{lg} be the optimal revenue
 778 in the instance that ignores all small jobs. For any feasible policy in the original instance, the revenue
 779 earned from small jobs is at most OPT^{sm} , and the revenue earned from big jobs is at most OPT^{lg} .
 780 Thus

$$\text{OPT} \leq \text{OPT}^{\text{sm}} + \text{OPT}^{\text{lg}}.$$

781 We construct two candidate menus and choose between them using computable certified lower bounds.
 782 Specifically, after constructing the menus below, define

$$L_{\text{sm}} = \frac{1}{4}U^{\text{sm}}, \quad L_{\text{lg}} = \frac{1}{2}\mathbb{E}[Y_{\text{max}}^{\text{lg}}].$$

783 The split menu p^{split} posts the small-job menu if $L_{\text{sm}} \geq L_{\text{lg}}$, and posts the large-job threshold menu
 784 otherwise.

785 • **Small job menu:** Solve the ex-ante benchmark restricted to $n \leq B/2$, post the resulting
 786 menu for $n \leq B/2$, and post $+\infty$ for $n > B/2$. Let U^{sm} denote the value of this restricted
 787 ex-ante benchmark. By the small-job analysis with $\varepsilon = 1/2$, this yields

$$\mathbb{E}[\text{revenue}] \geq \frac{1}{4}U^{\text{sm}} \geq \frac{1}{4}\text{OPT}^{\text{sm}}.$$

788 • **Large job threshold menu:** We do not compute OPT^{lg} or an optimal large-only posted
 789 menu. Instead, we use a one-item prophet threshold in virtual-value space. Let

$$\psi_n(v) = \max\{\phi_n(v), 0\}$$

790 denote the nonnegative virtual value for demand size n . For each arrival i , define

$$Y_i^{\text{lg}} = \begin{cases} \psi_{N_i}(V_i), & N_i > B/2, \\ 0, & \text{otherwise.} \end{cases}$$

791 Let $Y_{\max}^{\text{lg}} = \max_{i \in [m]} Y_i^{\text{lg}}$. By the classical one-item prophet inequality [22], there exists a
 792 threshold τ^{lg} (for example, the Samuel-Cahn median $\Pr[Y_{\max}^{\text{lg}} \geq \tau^{\text{lg}}] = \frac{1}{2}$ threshold for
 793 $Y_{\max}^{\text{lg}}$) such that selecting the first $Y_i^{\text{lg}} \geq \tau^{\text{lg}}$ has expected virtual surplus at least

$$\frac{1}{2} \mathbb{E}[Y_{\max}^{\text{lg}}].$$

794 This selection rule is implemented by the posted menu

$$p^{\text{lg}}(n) = \begin{cases} \inf\{v \geq 0 : \psi_n(v) \geq \tau^{\text{lg}}\}, & n > B/2, \\ \text{Not available}, & \text{otherwise.} \end{cases}$$

795 Under regularity, this posted menu induces a monotone allocation rule, so Myerson's
 796 lemma identifies its expected revenue with its expected virtual surplus. Moreover, the
 797 large-only posted-menu optimum is upper bounded by the Bayesian one-item virtual-surplus
 798 benchmark:

$$\text{OPT}^{\text{lg}} \leq \mathbb{E}[Y_{\max}^{\text{lg}}].$$

799 Therefore,

$$\mathbb{E}[\text{revenue from } p^{\text{lg}}] \geq \frac{1}{2} \text{OPT}^{\text{lg}}.$$

800 By construction, this split policy chooses the candidate with the larger certified lower bound. There-
 801 fore,

$$\mathbb{E}[\text{revenue}] \geq \max\{L_{\text{sm}}, L_{\text{lg}}\} \geq \max\left\{\frac{1}{4}\text{OPT}^{\text{sm}}, \frac{1}{2}\text{OPT}^{\text{lg}}\right\} \geq \frac{1}{6}(\text{OPT}^{\text{sm}} + \text{OPT}^{\text{lg}}) \geq \frac{1}{6}\text{OPT}.$$

802 D.4.3 Expected additive form

803 Fix a posted price function p and assume job sizes satisfy $1 \leq N_i \leq \bar{N}$. For this fixed menu, set
 804 $p_{\max} = \max_{1 \leq n \leq \bar{N}} p(n)/n$. Let $(V_i, N_i)_{i=1}^m$ be i.i.d. from $\mathcal{D}$ and define the submission indicator

$$b_i = \mathbf{1}\{V_i \geq p(N_i)\}, \quad X_i = N_i b_i.$$

805 Let $S_\ell(p) = \sum_{j=1}^\ell X_j$ and

$$K(p; V_{1:m}, N_{1:m}) = \max\left\{k \in \{0, \dots, m\} : \sum_{j=1}^k X_j \leq B\right\}.$$

806 Define the stopped-prefix revenue

$$\tilde{R}_B(p) = \sum_{i=1}^{K(p; \cdot)} b_i p(N_i).$$

807 The original sequential policy may accept later feasible jobs after rejecting an infeasible submitted
 808 job, so its realized revenue $R_B(p)$ satisfies $R_B(p) \geq \tilde{R}_B(p)$.

809 Define the “infinite-capacity” revenue and the stopped-prefix loss as

$$R_\infty(p) = \sum_{i=1}^m b_i p(N_i), \quad \tilde{\Delta}R(p) = R_\infty(p) - \tilde{R}_B(p).$$

810 Then

$$\mathbb{E}[R_B(p)] \geq \mathbb{E}[R_\infty(p)] - \mathbb{E}[\tilde{\Delta}R(p)], \quad \mathbb{E}[R_\infty(p)] = m \mathbb{E}_{(V, N) \sim \mathcal{D}}[p(N) \mathbf{1}\{V \geq p(N)\}].$$

811 If $S_m(p) > B$, then

$$S_m(p) - S_{K(p; \cdot)}(p) \leq (S_m(p) - B)_+ + \bar{N}.$$

812 Since $p(N_i) \leq p_{\max} N_i$, the stopped-prefix loss satisfies

$$\mathbb{E}[\tilde{\Delta}R(p)] \leq p_{\max} \mathbb{E}[(S_m(p) - B)_+] + p_{\max} \bar{N}.$$

813 Let p^* be an optimizer of the ex-ante benchmark

$$\max_{p(\cdot)} \mathbb{E}[R_\infty(p)] \quad \text{s.t.} \quad \mathbb{E}[S_m(p)] \leq B \quad (\text{i.e. } m \mathbb{E}[N \mathbf{1}\{V \geq p(N)\}] \leq B).$$

814 Write $\mu^* = \mathbb{E}[S_m(p^*)]$ and $p_{\max}^* = \max_{1 \leq n \leq \bar{N}} p^*(n)/n$. Since $\mu^* \leq B$,

$$\mathbb{E}[(S_m(p^*) - B)_+] \leq \mathbb{E}[(S_m(p^*) - \mu^*)_+] \leq \sqrt{\text{Var}(S_m(p^*))}.$$

815 The variables X_i are independent and satisfy $0 \leq X_i \leq \bar{N}$, so

$$\text{Var}(S_m(p^*)) = \sum_{i=1}^m \text{Var}(X_i) \leq \sum_{i=1}^m \mathbb{E}[X_i^2] \leq \bar{N} \sum_{i=1}^m \mathbb{E}[X_i] = \bar{N} \mu^*.$$

816 Consequently,

$$\mathbb{E}[R_B(p^*)] \geq \mathbb{E}[R_\infty(p^*)] - p_{\max}^* \sqrt{\bar{N} \mu^*} - p_{\max}^* \bar{N}. \quad (6)$$

817 The ex-ante benchmark upper bounds the optimal finite-capacity expected revenue, and p^* optimizes
818 that benchmark. Therefore, for $p^{\text{EA}} = p^*$, $p_{\max} = p_{\max}^*$, and $\mu_{\text{EA}} = \mu^*$,

$$\text{Rev}(p^{\text{EA}}) \geq \text{OPT} - p_{\max} \sqrt{\bar{N} \mu_{\text{EA}}} - p_{\max} \bar{N} \geq \text{OPT} - p_{\max} \sqrt{\bar{N} B} - p_{\max} \bar{N}.$$

819 D.4.4 Regret form

820 Fix a posted price function $p : \mathbb{N} \rightarrow [0, p_{\max}]$ and assume job sizes satisfy $1 \leq N_i \leq \bar{N}$. Let
821 $(V_i, N_i)_{i=1}^m$ be i.i.d. from $\mathcal{D}$ and define the submission indicator

$$b_i = \mathbf{1}\{V_i \geq p(N_i)\}, \quad X_i = N_i b_i.$$

822 Let $S_\ell(p) = \sum_{j=1}^\ell X_j$ and

$$K(p; V_{1:m}, N_{1:m}) = \max \left\{ k \in \{0, \dots, m\} : \sum_{j=1}^k X_j \leq B \right\}.$$

823 The realized sequential revenue (first come first serve until budget B is hit) is

$$R_B(p) = \sum_{i=1}^{K(p; \cdot)} b_i p(N_i),$$

824 so $\mathbb{E}[R_B(p)]$ is the exact expected revenue in the original sequential model.

825 Define the “infinite-capacity” revenue (accept all submitted jobs) and finite capacity loss as

$$R_\infty(p) = \sum_{i=1}^m b_i p(N_i), \quad \Delta R(p) = R_\infty(p) - R_B(p).$$

826 Then

$$\mathbb{E}[R_B(p)] = \mathbb{E}[R_\infty(p)] - \mathbb{E}[\Delta R(p)], \quad \mathbb{E}[R_\infty(p)] = m \mathbb{E}_{(V, N) \sim \mathcal{D}}[p(N) \mathbf{1}\{V \geq p(N)\}].$$

827 When $S_m(p) > B$, $S_m(p) - S_{K(p; \cdot)}(p) \leq (S_m(p) - B)_+ + \bar{N}$, $p(N_i) = \frac{p(N_i)}{N_i} N_i \leq p_{\max} N_i$, so
828 the finite capacity loss satisfies

$$\mathbb{E}[\Delta R(p)] \leq p_{\max} \mathbb{E}[(S_m(p) - B)_+] + p_{\max} \bar{N},$$

829 where $S_m(p) = \sum_{i=1}^m X_i$.

830 If $S_m(p) \leq B$, then $\mathbb{E}[\Delta R(p)] = 0$. So next we only consider $S_m(p) > B$. Let p^* be an optimizer
831 of the ex-ante benchmark

$$\max_{p(\cdot)} \mathbb{E}[R_\infty(p)] \quad \text{s.t.} \quad \mathbb{E}[S_m(p)] \leq B \quad (\text{i.e. } m \mathbb{E}[N \mathbf{1}\{V \geq p(N)\}] \leq B).$$

832 Then $\mathbb{E}[S_m(p^*)] \leq B$. Using Hoeffding’s inequality, with probability at least $1 - \delta$,

$$S_m(p^*) - \mathbb{E}[S_m(p^*)] \leq \sqrt{Nm \log(1/\delta)}$$

833 Since $\mathbb{E}[S_m(p^*)] \leq B$, we have

$$S_m(p^*) - B \leq \sqrt{2Nm \log(1/\delta)}$$

834 Consequently,

$$\mathbb{E}[R_B(p^*)] \geq \mathbb{E}[R_\infty(p^*)] - p_{\max} \sqrt{2Nm \log(1/\delta)} - p_{\max} \bar{N} \quad (7)$$

835 And by the strong duality, $\max_p \text{Rev} \leq \mathbb{E}[R_\infty(p^*)]$, so for $p^{\text{EA}} = p^*$, with probability $1 - \delta$,

$$\text{Rev}(p^{\text{EA}}) \geq \max_p \text{Rev}(p) - p_{\max} \sqrt{2Nm \log(1/\delta)} - p_{\max} \bar{N}.$$

836 D.4.5 Monotone ex-ante benchmark menu

837 Define the normalized virtual value

$$\psi_n(x) = \frac{\phi_n(x)}{n}.$$

838 The quantity $\psi_n(x)$ is the virtual value per unit of GPU capacity generated by a sale of size n at total
839 price x .

840 **Assumption 1** (Regularity and interiority). *For every $n \in \mathcal{N}$, the conditional distribution $V \mid N = n$
841 is strictly regular, i.e., $\phi_n(x)$ is strictly increasing on the relevant support of $V \mid N = n$. Moreover,
842 the ex-ante benchmark has an interior optimizer p^{EA} , meaning*

$$0 < S_n(p^{\text{EA}}(n)) < 1 \quad \text{for all } n \in \mathcal{N}.$$

843 Finally, whenever $n < n'$, the price $p^{\text{EA}}(n)$ lies in the domain on which $\psi_{n'}$ is defined. This is
844 automatically satisfied, for example, when the conditional value supports are nested in n .

845 The following condition is the key cross-size monotonicity condition.

846 **Assumption 2** (Normalized virtual-value ordering). *For every pair $n < n'$ in $\mathcal{N}$ and every relevant
847 price x such that $\psi_n(x) \geq 0$,*

$$\psi_{n'}(x) \leq \psi_n(x).$$

848 Equivalently,

$$\frac{\phi_{n'}(x)}{n'} \leq \frac{\phi_n(x)}{n}.$$

849 We say that the ordering is strict if, for every $n < n'$ and every relevant price x such that $\psi_n(x) \geq 0$,

$$\psi_{n'}(x) < \psi_n(x).$$

850 Assumption 2 has a simple economic interpretation. At the same total price x , a larger GPU request
851 should generate weakly lower virtual value per GPU than a smaller request. Thus, larger jobs may
852 have larger total values, but not in a way that makes their virtual surplus per unit of capacity larger at
853 the same total price. This is a single-crossing condition across demand sizes.

854 **Proposition 1** (Monotonicity of the ex-ante benchmark menu). *Suppose Assumption 1 holds.*

855 1. *If Assumption 2 holds, then the ex-ante optimal menu is nondecreasing on $\mathcal{N}$:*

$$n < n' \implies p^{\text{EA}}(n) \leq p^{\text{EA}}(n').$$

856 2. *If the strict version of Assumption 2 holds, then the ex-ante optimal menu is strictly increasing
857 on $\mathcal{N}$:*

$$n < n' \implies p^{\text{EA}}(n) < p^{\text{EA}}(n').$$

858 *Proof.* Let $\beta = B/m$, and reparameterize by acceptance probabilities $q_n = S_n(p(n))$. Writing P_n
 859 for the inverse demand curve and $r_n(q) = qP_n(q)$, the ex-ante benchmark is

$$\max_{q_n \in [0,1]} \sum_{n \in \mathcal{N}} w_n r_n(q_n) \quad \text{s.t.} \quad \sum_{n \in \mathcal{N}} w_n n q_n \leq \beta.$$

860 By Assumption 1, the optimizer is interior and the KKT conditions give, for some $\lambda \geq 0$,

$$r'_n(q_n^{\text{EA}}) = \lambda n, \quad n \in \mathcal{N}.$$

861 If $p = P_n(q)$, then $q = S_n(p)$ and $P'_n(q) = -1/f_n(p)$, so

$$r'_n(q) = P_n(q) + qP'_n(q) = p - \frac{S_n(p)}{f_n(p)} = \phi_n(p).$$

862 Hence, for every n ,

$$\phi_n(p^{\text{EA}}(n)) = \lambda n, \quad \text{equivalently} \quad \psi_n(p^{\text{EA}}(n)) = \lambda.$$

863 Fix $n < n'$ and write $p_n = p^{\text{EA}}(n)$ and $p_{n'} = p^{\text{EA}}(n')$. Since $\lambda \geq 0$, Assumption 2 implies

$$\psi_{n'}(p_n) \leq \psi_n(p_n) = \lambda = \psi_{n'}(p_{n'}).$$

864 Because $\psi_{n'}$ is strictly increasing, $p_n \leq p_{n'}$. Thus p^{EA} is nondecreasing.

865 If the strict version of Assumption 2 holds, then

$$\psi_{n'}(p_n) < \psi_n(p_n) = \lambda = \psi_{n'}(p_{n'}),$$

866 and strict monotonicity of $\psi_{n'}$ gives $p_n < p_{n'}$. Hence p^{EA} is strictly increasing. $\square$

867 **Remark 2** (Binding capacity and a slightly weaker strict condition). *If the ex-ante capacity constraint*
 868 *binds, then $\lambda > 0$. In that case, for strict monotonicity it is enough to assume strict normalized*
 869 *virtual-value ordering only on the region where $\psi_n(x) > 0$, rather than on the weakly nonnegative*
 870 *region $\psi_n(x) \geq 0$. Indeed, at the optimum,*

$$\psi_n(p^{\text{EA}}(n)) = \lambda > 0,$$

871 *so the proof only uses the strict ordering at prices with positive normalized virtual value.*

872 *A sufficient condition for the capacity constraint to bind is that the unconstrained monopoly menu*
 873 *violates the ex-ante capacity constraint. If p_n^M solves*

$$\phi_n(p_n^M) = 0,$$

874 *and*

$$\sum_{n \in \mathcal{N}} w_n n S_n(p_n^M) > \frac{B}{m},$$

875 *then the capacity constraint binds and $\lambda > 0$.*

876 **Primitive sufficient families.** We next give useful primitive conditions under which Assumption 2
 877 holds.

878 **Lemma 1** (Sublinear scale-value model). *Suppose*

$$V = g(N)X,$$

879 *where X is independent of N and has a strictly increasing virtual value*

$$\phi_X(z) = z - \frac{S_X(z)}{f_X(z)}.$$

880 *Assume that*

$$g(n) > 0, \quad g(n) \text{ is increasing in } n, \quad \frac{g(n)}{n} \text{ is nonincreasing in } n.$$

881 *Then the weak normalized virtual-value ordering in Assumption 2 holds. If g is strictly increasing,*
 882 *then the strict normalized virtual-value ordering holds on the region $\psi_n(x) \geq 0$. Consequently, under*
 883 *Assumption 1, the ex-ante benchmark menu is strictly increasing.*

884 *Proof.* Fix n . Since $V = g(n)X$ conditional on $N = n$,

$$S_n(x) = S_X\left(\frac{x}{g(n)}\right), \quad f_n(x) = \frac{1}{g(n)}f_X\left(\frac{x}{g(n)}\right).$$

885 Hence

$$\phi_n(x) = x - \frac{S_n(x)}{f_n(x)} = g(n)\phi_X\left(\frac{x}{g(n)}\right), \quad \psi_n(x) = \frac{g(n)}{n}\phi_X\left(\frac{x}{g(n)}\right).$$

886 Fix $n < n'$ and write

$$a_n = \frac{g(n)}{n}, \quad z_n = \frac{x}{g(n)}.$$

887 Then $a_{n'} \leq a_n$ and $z_{n'} \leq z_n$. If $\psi_n(x) \geq 0$, then $\phi_X(z_n) \geq 0$. Since ϕ_X is increasing, $\phi_X(z_{n'}) \leq$
888 $\phi_X(z_n)$, and therefore

$$\begin{aligned} \psi_n(x) - \psi_{n'}(x) &= a_n\phi_X(z_n) - a_{n'}\phi_X(z_{n'}) \\ &= a_{n'}(\phi_X(z_n) - \phi_X(z_{n'})) + (a_n - a_{n'})\phi_X(z_n) \geq 0. \end{aligned}$$

889 Thus $\psi_{n'}(x) \leq \psi_n(x)$, which proves the weak normalized virtual-value ordering.

890 If g is strictly increasing, then for every relevant $x > 0$ we have $z_{n'} < z_n$. Since ϕ_X is strictly
891 increasing, the first term in the last display is strictly positive, while the second is nonnegative.
892 Hence $\psi_{n'}(x) < \psi_n(x)$, proving the strict ordering on the relevant positive-price region. The strict
893 monotonicity of p^{EA} then follows from Proposition 1. $\square$

894 **Examples satisfying the assumptions.** Lemma 1 applies to scale models $V = g(N)X$ with X
895 strictly regular and independent of N . In this case

$$\phi_n(x) = g(n)\phi_X\left(\frac{x}{g(n)}\right), \quad \psi_n(x) = \frac{g(n)}{n}\phi_X\left(\frac{x}{g(n)}\right).$$

896 Thus Assumption 2 holds whenever g is increasing and $g(n)/n$ is nonincreasing. If g is strictly
897 increasing, the ordering is strict on the relevant positive-price region, and Proposition 1 implies that
898 the ex-ante benchmark menu is strictly increasing.

899 **1. Linear per-GPU values.** If $V = nX$, then $g(n) = n$ and

$$\psi_n(x) = \phi_X(x/n).$$

900 The KKT condition gives

$$\phi_X\left(\frac{p^{\text{EA}}(n)}{n}\right) = \lambda, \quad p^{\text{EA}}(n) = n\phi_X^{-1}(\lambda).$$

901 Hence the ex-ante benchmark charges a constant per-GPU price.

902 **2. Power scaling.** If $V = n^\alpha X$ with $0 < \alpha < 1$, then $g(n) = n^\alpha$ is increasing and
903 $g(n)/n = n^{\alpha-1}$ is decreasing. Hence the strict ordering holds under strict regularity of X .
904 Moreover,

$$\psi_n(x) = n^{\alpha-1}\phi_X\left(\frac{x}{n^\alpha}\right), \quad p^{\text{EA}}(n) = n^\alpha\phi_X^{-1}(\lambda n^{1-\alpha}).$$

905 The case $\alpha = 1/2$ gives the useful model $V = \sqrt{n}X$.

906 **3. General concave scaling.** More generally, if g is increasing, concave, and $g(0) = 0$, then
907 $g(n)/n$ is nonincreasing. Therefore any strictly increasing such g yields a strictly increasing
908 ex-ante menu under strict regularity. Examples include

$$g(n) = \sqrt{n}, \quad g(n) = n^\alpha \ (0 < \alpha \leq 1), \quad g(n) = \log(1 + n).$$

909 **4. Size-independent total values.** If $V = X$, then

$$\psi_n(x) = \frac{\phi_X(x)}{n}.$$

910 Thus the ordering is weak on $\{\phi_X(x) \geq 0\}$ and strict on $\{\phi_X(x) > 0\}$. If the ex-ante
911 capacity constraint binds, so that $\lambda > 0$, then

$$p^{\text{EA}}(n) = \phi_X^{-1}(\lambda n),$$

912 and the menu is strictly increasing. If the constraint is slack, then $\lambda = 0$ and all sizes receive
913 the same monopoly price, so strict monotonicity need not hold.

The baseline distribution X may be any strictly regular distribution, including exponential distributions, uniform distributions on an interval, Weibull distributions with shape parameter at least one, and, under standard smoothness conditions, distributions with increasing hazard rate.

The key point is that regularity is only a within-size condition. Monotonicity across GPU demands requires the cross-size ordering in Assumption 2: at the same total price, larger requests must have no higher normalized virtual value. Under this condition, the ex-ante benchmark equalizes virtual value per GPU and therefore assigns weakly, or strictly, higher total prices to larger requests.

D.5 Virtual value reduction and proof of Theorem 5

Knapsack prophet inequalities are designed for online selection problems where items arrive sequentially, and the decision maker must irrevocably accept or reject each item subject to a knapsack capacity. Our problem setting shares a similar structure, but the main difference is that we optimize revenue rather than welfare: the platform does not directly observe a value to select, but instead posts a price and earns revenue only when the customer buys. However, we are able to adapt this method to our setting by leveraging Myerson’s lemma.

Myerson’s lemma characterizes optimal revenue in single parameter mechanism design as expected virtual surplus [26] and so that links the welfare optimization and revenue optimization. Inspired by this, our second method for GPU selling reduces revenue maximization to welfare maximization in a *virtual* knapsack instance, and then applies a static-price prophet inequality for knapsack [25, 12].

Define the nonnegative virtual value $\psi_n(v) = \max\{\phi_n(v), 0\}$. For a realized sequence $\{(v_i, n_i)\}_{i=1}^m$, consider the induced virtual knapsack instance with item value $y_i = \psi_{n_i}(v_i)$ and normalized size $s_i = n_i/B \in (0, 1]$. Let

$$V_\psi^* = \max\left\{\sum_{i=1}^m y_i z_i : z_i \in \{0, 1\}, \sum_{i=1}^m s_i z_i \leq 1\right\}$$

denote the offline optimal virtual welfare.

The knapsack-prophet step supplies a static threshold rule in this virtual instance. When $\bar{N} \leq B/2$, we use a single per-unit virtual threshold

$$\tau(n) = \omega n, \quad \omega = \frac{2}{3B} \hat{\mu}_\psi, \quad \hat{\mu}_\psi = \mathbb{E}[V_\psi^*].$$

We then post the real-value menu

$$p^{\text{VKP}}(n) = \inf\{v \geq 0 : \psi_n(v) \geq \tau(n)\}.$$

Thus a buyer accepts iff her nonnegative virtual value is no less than the prophet threshold. In the general case $\bar{N} > B/2$, the 5 approximation guarantee is obtained by taking the better of two static menus.

Virtual knapsack benchmark. Given a realization $\{(v_i, n_i)\}_{i=1}^m$, define

$$y_i = \psi_{n_i}(v_i), \quad s_i = \frac{n_i}{B}.$$

The corresponding virtual knapsack optimum is

$$V_\psi^*(v_{1:m}, n_{1:m}) = \max\left\{\sum_{i=1}^m y_i z_i : z_i \in \{0, 1\}, \sum_{i=1}^m s_i z_i \leq 1\right\}.$$

We write simply V_ψ^* for this random variable. For the general regime $\bar{N} > B/2$, we also use the small/large decomposition

$$V_{\psi, \text{sm}}^* = \max\left\{\sum_{i: n_i \leq B/2} y_i z_i : z_i \in \{0, 1\}, \sum_{i: n_i \leq B/2} s_i z_i \leq 1\right\},$$

946

$$V_{\psi, \text{lg}}^* = \max\{y_i : n_i > B/2\},$$

where $V_{\psi, \text{lg}}^* = 0$ if no large item is present. Since at most one large item can be accepted, the large-item part is a single-choice prophet problem.

We use the following exact-threshold consequence of the static-price prophet inequalities for knapsack: static anonymous thresholds achieve expected welfare at least $1/3$ of the offline optimum when all sizes satisfy $n_i \leq 1/2$, and at least $1/5$ of the offline optimum in general. In the small-item regime, the threshold can be taken to be a single per-unit price proportional to $\mathbb{E}[V^*]$; in the general regime, the $1/5$ guarantee is obtained by combining that small-item price with a separate large-item reserve and taking the better of the two menus [25, 12].

Lemma 2 (Static thresholds for stochastic integral knapsack). *Consider a sequence of independent stochastic items $(Y_i, S_i)_{i=1}^m$, where $Y_i \geq 0$ is the item value and $S_i \in (0, 1]$ is the normalized size. The value and size of the same item may be arbitrarily correlated. For a realized profile, let*

$$V^* = \max \left\{ \sum_i Y_i z_i : z_i \in \{0, 1\}, \sum_i S_i z_i \leq 1 \right\}.$$

If $S_i \leq 1/2$ almost surely for all i , then the static per-unit threshold policy with

$$p = \frac{2}{3} \mathbb{E}[V^*] = \mathbb{E} \left[\frac{2}{3} V^* \right]$$

that accepts item i iff $Y_i \geq pS_i$ and the item fits obtains expected welfare at least $\frac{1}{3} \mathbb{E}[V^*]$.

In the general case, define

$$V_{\text{sm}}^* = \max \left\{ \sum_{i: S_i \leq 1/2} Y_i z_i : z_i \in \{0, 1\}, \sum_{i: S_i \leq 1/2} S_i z_i \leq 1 \right\},$$

and

$$V_{\text{lg}}^* = \max \{ Y_i : S_i > 1/2 \},$$

with $V_{\text{lg}}^* = 0$ if no large item is present. Let

$$\mu_{\text{sm}} = \mathbb{E}[V_{\text{sm}}^*], \quad \mu_{\text{lg}} = \mathbb{E}[V_{\text{lg}}^*].$$

Then the small-item per-unit threshold

$$p_{\text{sm}} = \frac{2}{3} \mu_{\text{sm}} = \mathbb{E} \left[\frac{2}{3} V_{\text{sm}}^* \right]$$

obtains expected welfare at least $\mu_{\text{sm}}/3$ from small items, and the large-item take-it-or-leave-it threshold

$$\theta_{\text{lg}} = \frac{1}{2} \mu_{\text{lg}} = \mathbb{E} \left[\frac{1}{2} V_{\text{lg}}^* \right]$$

obtains expected welfare at least $\mu_{\text{lg}}/2$ from large items. Hence the better of these two static threshold policies obtains expected welfare at least

$$\frac{1}{5} \mathbb{E}[V^*].$$

Proof. First suppose all items are small, so $S_i \leq 1/2$ almost surely. Let $\mu = \mathbb{E}[V^*]$ and set $p = 2\mu/3$. Let x_i be the acceptance indicator of the threshold policy, and let

$$C = \sum_i S_i x_i$$

be the total capacity sold. Decompose the welfare of the policy as

$$\sum_i Y_i x_i = pC + \sum_i x_i (Y_i - pS_i).$$

Let $q = \Pr[C \leq 1/2]$. The revenue part satisfies

$$\mathbb{E}[pC] \geq \frac{p}{2} \Pr[C > 1/2] = \frac{\mu}{3} (1 - q).$$

972 It remains to lower-bound the surplus term. Fix an offline optimal solution $O(T)$ for each type profile
 973 $T = ((Y_1, S_1), \dots, (Y_m, S_m))$, with arbitrary tie-breaking. For each item i , define

$$z_i(T_i) = \Pr_{T_{-i}}\{i \in O(T_i, T_{-i})\}.$$

974 Then

$$\sum_i \mathbb{E}[z_i(T_i)Y_i] = \mu, \quad \sum_i \mathbb{E}[z_i(T_i)S_i] \leq 1.$$

975 Let $C_{<i}$ be the capacity sold before item i arrives. Since $C \leq 1/2$ implies $C_{<i} \leq 1/2$, we have

$$\Pr[C_{<i} \leq 1/2] \geq q.$$

976 Moreover, when $C_{<i} \leq 1/2$, every small item still fits. Therefore item i obtains surplus at least

$$\mathbf{1}\{C_{<i} \leq 1/2\} z_i(T_i)(Y_i - pS_i)^+.$$

977 Using independence of T_i from the previous history,

$$\mathbb{E}[x_i(Y_i - pS_i)] \geq q \mathbb{E}[z_i(T_i)(Y_i - pS_i)^+].$$

978 Summing over i and using $(a)^+ \geq a$ gives

$$\mathbb{E}\left[\sum_i x_i(Y_i - pS_i)\right] \geq q \sum_i \mathbb{E}[z_i(T_i)(Y_i - pS_i)] \geq q(\mu - p) = \frac{\mu}{3}q.$$

979 Adding the revenue and surplus lower bounds yields

$$\mathbb{E}\left[\sum_i Y_i x_i\right] \geq \frac{\mu}{3}(1 - q) + \frac{\mu}{3}q = \frac{\mu}{3}.$$

980 This proves the small-item guarantee. The same argument applies to V_{sm}^* in the general case by
 981 ignoring all items with $S_i > 1/2$.

982 For large items, set

$$X_i = Y_i \mathbf{1}\{S_i > 1/2\}, \quad M = \max_i X_i = V_{\text{lg}}^*, \quad \theta = \frac{1}{2}\mathbb{E}[M].$$

983 The threshold policy accepts the first item with $X_i \geq \theta$. Let W denote its value, and let

$$r_i = \Pr[X_j < \theta \text{ for all } j < i].$$

984 Then

$$\mathbb{E}[W] = \theta \Pr[M \geq \theta] + \sum_i r_i \mathbb{E}[(X_i - \theta)^+].$$

985 Since $r_i \geq \Pr[M < \theta]$ for every i ,

$$\mathbb{E}[W] \geq \theta \Pr[M \geq \theta] + \Pr[M < \theta] \sum_i \mathbb{E}[(X_i - \theta)^+].$$

986 Pointwise,

$$M \leq \theta + \sum_i (X_i - \theta)^+,$$

987 and hence

$$\sum_i \mathbb{E}[(X_i - \theta)^+] \geq \mathbb{E}[M] - \theta = \theta.$$

988 Therefore,

$$\mathbb{E}[W] \geq \theta \Pr[M \geq \theta] + \theta \Pr[M < \theta] = \theta = \frac{1}{2}\mathbb{E}[V_{\text{lg}}^*].$$

989 Finally, pointwise,

$$V^* \leq V_{\text{sm}}^* + V_{\text{lg}}^*,$$

990 because the small part of any feasible solution is feasible for V_{sm}^* , while the large part contains at
 991 most one item. Thus

$$\mathbb{E}[V^*] \leq \mu_{\text{sm}} + \mu_{\text{lg}}.$$

992 The better of the small and large threshold policies obtains at least

$$\max\left\{\frac{\mu_{\text{sm}}}{3}, \frac{\mu_{\text{lg}}}{2}\right\} \geq \frac{\mu_{\text{sm}} + \mu_{\text{lg}}}{5} \geq \frac{1}{5}\mathbb{E}[V^*],$$

993 which proves the general guarantee. $\square$

994 **D.5.1 Threshold for different regimes**

995 **Small-item regime** ($\bar{N} \leq B/2$). Let $\mu_\psi = \mathbb{E}[V_\psi^*]$ and set

$$\tau(n) = \omega n, \quad \omega = \frac{2}{3B} \mu_\psi.$$

996 **General regime** ($\bar{N} > B/2$). Construct two candidate threshold menus:

$$\tau_{\text{sm}}(n) = \begin{cases} \frac{2}{3B} \mu_{\psi, \text{sm}} n, & n \leq B/2, \\ +\infty, & n > B/2, \end{cases}$$

997 where $\mu_{\psi, \text{sm}} = \mathbb{E}[V_{\psi, \text{sm}}^*]$, and

$$\tau_{\text{lg}}(n) = \begin{cases} \frac{1}{2} \mu_{\psi, \text{lg}}, & n > B/2, \\ +\infty, & n \leq B/2, \end{cases}$$

998 where $\mu_{\psi, \text{lg}} = \mathbb{E}[V_{\psi, \text{lg}}^*]$. Let p^{VKP} denote the better of the two induced menus.

999 **Posted prices in value space.** Given any virtual threshold $\tau(\cdot)$, define the posted menu

$$p_\tau(n) = \inf\{v \geq 0 : \psi_n(v) \geq \tau(n)\}.$$

1000 This is well-defined because ψ_n is nondecreasing after ironing. The key equivalence is immediate:

$$v \geq p_\tau(n) \iff \psi_n(v) \geq \tau(n). \quad (8)$$

1001 Therefore, if the platform posts p_τ and accepts iff the request fits, then the resulting acceptance rule
1002 coincides pathwise with the threshold-based prophet rule in the virtual instance.

1003 **Lemma 3** (Expected revenue equals expected virtual surplus). *Let $p(\cdot)$ be any posted menu and let*
1004 *$x_i \in \{0, 1\}$ be its acceptance indicator. Then*

$$\mathbb{E}[\text{Rev}(p)] = \mathbb{E}\left[\sum_{i=1}^m x_i \phi_{n_i}(v_i)\right].$$

1005 *If, in addition, $x_i = 1$ implies $\psi_{n_i}(v_i) \geq 0$, then*

$$\mathbb{E}[\text{Rev}(p)] = \mathbb{E}\left[\sum_{i=1}^m x_i \psi_{n_i}(v_i)\right].$$

1006 *Proof.* Fix an agent i and condition on the realized history up to time $i - 1$ and on $n_i = n$. Under
1007 any posted menu, the agent is accepted iff the feasibility event determined by the history occurs
1008 and $v_i \geq p(n)$. Hence the allocation rule is monotone in v_i , and the payment upon acceptance is
1009 the critical value $p(n)$. The standard Myerson payment identity for single-parameter environments
1010 therefore implies that expected payment equals expected (ironed) virtual value times the allocation
1011 probability. Summing over agents gives the first identity.

1012 For the second identity, note that $\psi_n(v) = \phi_n(v)$ whenever $\phi_n(v) \geq 0$, and the hypothesis ensures
1013 that allocation occurs only in that region. Hence replacing ϕ_n by ψ_n does not change the value of the
1014 expectation. $\square$

1015 **Lemma 4** (Virtual benchmark upper-bounds posted-menu revenue). *Let Rev^* denote the Bayesian-*
1016 *optimal DSIC revenue over all mechanisms satisfying the same knapsack feasibility constraint.*
1017 *Then*

$$\text{OPT} \leq \text{Rev}^* = \mathbb{E}[V_\psi^*].$$

1018 *Proof.* The inequality $\text{OPT} \leq \text{Rev}^*$ is immediate because anonymous posted menus are a special
1019 case of DSIC mechanisms. For the equality, public sizes n_i reduce the private information of each
1020 buyer to a single scalar value v_i , so the problem is a single-parameter Bayesian mechanism design
1021 problem with a downward-closed feasibility constraint (knapsack). By Myerson's characterization, the
1022 DSIC revenue-optimal mechanism maximizes ironed virtual surplus pointwise subject to feasibility.
1023 Because we truncate negative virtual values at zero, this pointwise maximization is exactly the offline
1024 knapsack problem defining V_ψ^* . Taking expectations yields the claim. $\square$

1025 *Proof of Theorem 5.* We first treat the small-item regime $\bar{N} \leq B/2$. Let $\tau(n) = \omega n$ with $\omega =$
 1026 $(2/3B)\mu_\psi$, and let $p^{\text{VKP}} = p_\tau$. By (8), a type (v_i, n_i) accepts the posted price iff its nonnegative
 1027 virtual value satisfies $\psi_{n_i}(v_i) \geq \tau(n_i)$. Since both the posted-price rule and the prophet rule
 1028 additionally impose the same feasibility test “fits in the remaining capacity,” they accept exactly the
 1029 same set of buyers on every sample path.

1030 Let x_i denote this common acceptance indicator. By Lemma 3,

$$\mathbb{E}[\text{Rev}(p^{\text{VKP}})] = \mathbb{E}\left[\sum_{i=1}^m x_i \psi_{n_i}(v_i)\right].$$

1031 Applying Lemma 2,

$$\mathbb{E}\left[\sum_{i=1}^m x_i \psi_{n_i}(v_i)\right] \geq \frac{1}{3} \mathbb{E}[V_\psi^*].$$

1032 By Lemma 4,

$$\mathbb{E}[\text{Rev}(p^{\text{VKP}})] \geq \frac{1}{3} \mathbb{E}[V_\psi^*] \geq \frac{1}{3} \text{OPT}.$$

1033 This proves the first part.

1034 For the general regime $\bar{N} > B/2$, construct the two candidate menus induced by τ_{sm} and τ_{lg} , and let
 1035 p^{VKP} be the better of the two. By the same acceptance-equivalence argument and Lemma 3, each
 1036 candidate menu earns expected revenue exactly equal to the expected virtual welfare obtained by the
 1037 corresponding threshold rule in the virtual instance. By Lemma 2,

$$\mathbb{E}[\text{Rev}(p^{\text{VKP}})] \geq \frac{1}{5} \mathbb{E}[V_\psi^*].$$

1038 Applying Lemma 4 again yields

$$\mathbb{E}[\text{Rev}(p^{\text{VKP}})] \geq \frac{1}{5} \text{OPT}.$$

1039 This completes the proof. □

1040 **Algorithmic remark.** The theorem is stated in the idealized exact-threshold regime, where the
 1041 benchmark expectations μ_ψ , $\mu_{\psi, \text{sm}}$, and $\mu_{\psi, \text{lg}}$ are available exactly. If these quantities are instead
 1042 computed via the FPTAS-based threshold routine of [12], then for every fixed $\delta > 0$ one obtains
 1043 polynomial-time computable menus with guarantees $(3 + \delta)^{-1}$ in the small item regime and $(5 + \delta)^{-1}$
 1044 in the general regime.

1045 **Monotonicity of the VKP menu** We next record that the same cross-size virtual-value ordering
 1046 used for the ex-ante benchmark also implies monotonicity of the virtual-knapsack-prophet menu.
 1047 Since the argument is analogous to Proposition 1, we keep the discussion brief.

1048 To avoid overloading notation, write

$$\phi_n^+(v) = \max\{\phi_n(v), 0\}$$

1049 for the nonnegative virtual value used by the VKP construction, and write

$$\Psi_n(v) = \frac{\phi_n(v)}{n}$$

1050 for the normalized virtual value appearing in Assumption 2. The VKP menu is

$$p^{\text{VKP}}(n) = \inf\{v \geq 0 : \phi_n^+(v) \geq \omega n\}.$$

1051 When $\omega > 0$, this is equivalently

$$p^{\text{VKP}}(n) = \inf\{v \geq 0 : \phi_n(v) \geq \omega n\} = \inf\{v \geq 0 : \Psi_n(v) \geq \omega\}.$$

1052 We impose the following mild threshold-interiority condition.

1053 **Assumption 3** (VKP threshold interiority). *The VKP threshold satisfies $\omega > 0$. For every $n \in \mathcal{N}$,*
 1054 *the virtual value ϕ_n is continuous and strictly increasing on the relevant support of $V \mid N = n$, and*
 1055 *the equation*

$$\phi_n(v) = \omega n$$

1056 *has a solution in the interior of that support. Moreover, whenever $n < n'$, the price $p^{\text{VKP}}(n)$ lies in*
 1057 *the domain on which $\Psi_{n'}$ is defined.*

1058 **Proposition 2** (Monotonicity of the VKP menu). *Suppose Assumption 3 holds.*

1059 1. *If Assumption 2 holds, then the VKP menu is nondecreasing on $\mathcal{N}$:*

$$n < n' \implies p^{\text{VKP}}(n) \leq p^{\text{VKP}}(n').$$

1060 2. *If the strict version of Assumption 2 holds, then the VKP menu is strictly increasing on $\mathcal{N}$:*

$$n < n' \implies p^{\text{VKP}}(n) < p^{\text{VKP}}(n').$$

1061 *Proof.* By Assumption 3, for each $n \in \mathcal{N}$ the equation

$$\phi_n(p) = \omega n$$

1062 has a unique interior solution. Since $\omega > 0$, the truncation in $\phi_n^+ = \max\{\phi_n, 0\}$ is inactive at this
 1063 threshold, so this solution is $p^{\text{VKP}}(n)$. Hence

$$\Psi_n(p^{\text{VKP}}(n)) = \omega, \quad n \in \mathcal{N}.$$

1064 Fix $n < n'$ and write

$$p_n = p^{\text{VKP}}(n), \quad p_{n'} = p^{\text{VKP}}(n').$$

1065 Since $\Psi_n(p_n) = \omega > 0$, Assumption 2 gives

$$\Psi_{n'}(p_n) \leq \Psi_n(p_n) = \omega = \Psi_{n'}(p_{n'}).$$

1066 Because $\Psi_{n'}$ is strictly increasing, $p_n \leq p_{n'}$. Thus the VKP menu is nondecreasing.

1067 If the strict version of Assumption 2 holds, then

$$\Psi_{n'}(p_n) < \Psi_n(p_n) = \omega = \Psi_{n'}(p_{n'}),$$

1068 and strict monotonicity of $\Psi_{n'}$ yields $p_n < p_{n'}$. Hence the VKP menu is strictly increasing. $\square$

1069 D.6 Dynamic pricing benchmark and state complexity

1070 We briefly record the exact dynamic programming benchmark for the duration setting. The point of
 1071 this benchmark is mainly conceptual: it makes precise the opportunity cost of accepting a job and
 1072 clarifies why the duration case is much harder than the single-period setting.

1073 The natural state variable must therefore track the residual-duration profile of currently active jobs.
 1074 Let

$$S_t = (s_t(1), \dots, s_t(L_{\max})) \in \{0, \dots, B\}^{L_{\max}}, \quad \sum_{\ell=1}^{L_{\max}} s_t(\ell) \leq B,$$

1075 where $s_t(\ell)$ denotes the number of GPUs currently allocated to jobs with ℓ remaining periods. If no
 1076 new job is admitted, the state evolves by a deterministic shift:

$$s_{t+1}(\ell-1) \leftarrow s_t(\ell), \quad \ell = 2, \dots, L_{\max}, \quad s_{t+1}(L_{\max}) \leftarrow 0.$$

1077 If a job requiring n GPUs for λ periods is accepted, then after this shift we add n GPUs to the
 1078 λ -bucket. Writing $\Phi(S)$ for the shift operator and e_λ for the λ -th unit vector, the next states are

$$S^{\text{rej}} = \Phi(S), \quad S^{\text{acc}} = \Phi(S) + ne_\lambda.$$

1079 A type (n, λ) is feasible in state S if

$$\sum_{\ell=1}^{L_{\max}} s(\ell) + n \leq B.$$

1080 Let $V_t(S)$ denote the optimal expected revenue from periods $t, t+1, \dots, T$ starting from state S at
 1081 time t . Fix a period t , a state S , and a realized feasible type (n, λ) . Let

$$q_{n,\lambda}(p) = \mathbf{Pr}[v_t \geq p \mid n_t = n, \lambda_t = \lambda]$$

1082 be the conditional sale probability at price p . Posting price p yields expected continuation value

$$V_{t+1}(S^{\text{rej}}) + q_{n,\lambda}(p)(p + V_{t+1}(S^{\text{acc}}) - V_{t+1}(S^{\text{rej}})).$$

1083 Define the continuation gap

$$\Delta_t(S; n, \lambda) = V_{t+1}(S^{\text{rej}}) - V_{t+1}(S^{\text{acc}}) \geq 0,$$

1084 which is the opportunity cost of allocating n GPUs for λ periods. Conditional on type (n, λ) , the
 1085 pricing problem becomes

$$p_t^*(S; n, \lambda) \in \arg \max_{p \geq 0} q_{n,\lambda}(p)(p - \Delta_t(S; n, \lambda)).$$

1086 Hence the optimal online menu is

$$p_t(n, \lambda) = \begin{cases} +\infty, & \text{if } (n, \lambda) \text{ is infeasible in } S_t, \\ p_t^*(S_t; n, \lambda), & \text{otherwise.} \end{cases}$$

1087 The full Bellman recursion then averages this conditional problem over the arrival process for (n_t, λ_t)
 1088 and the valuation distribution.

1089 This benchmark already shows why the duration case is difficult. In the single-period model, the state
 1090 can often be summarized by a scalar, namely the currently available capacity. With heterogeneous
 1091 durations, that scalar is no longer sufficient: two systems with the same number of occupied GPUs
 1092 can have very different continuation values depending on when those GPUs will be released. The
 1093 DP must therefore keep track of the entire residual-duration profile S_t , which turns the state into an
 1094 $L_{\max}$ -dimensional object.

1095 The number of feasible states is

$$|\mathcal{S}| = \left| \left\{ s \in \mathbb{Z}_+^{L_{\max}} : \sum_{\ell=1}^{L_{\max}} s(\ell) \leq B \right\} \right| = \sum_{k=0}^B \binom{k + L_{\max} - 1}{L_{\max} - 1} = \binom{B + L_{\max}}{L_{\max}}.$$

1096 Thus the state space grows combinatorially in both B and $L_{\max}$. Even for moderate parameters this
 1097 is already prohibitive. For instance, when $B = 64$ and $L_{\max} = 8$,

$$|\mathcal{S}| = \binom{72}{8} \approx 1.2 \times 10^{10}.$$

1098 Backward induction must evaluate these states at every time step and, within each state, solve a
 1099 pricing problem for each feasible type (n, λ) . If GPU requirements range over $\{1, \dots, B\}$ and
 1100 durations over $\{1, \dots, L_{\max}\}$, then the per-period type space is already of order $BL_{\max}$, before one
 1101 even accounts for the need to optimize over prices or evaluate valuation tails $q_{n,\lambda}(\cdot)$. Consequently,
 1102 the exact DP is subject to a severe curse of dimensionality and is computationally realistic only at
 1103 very small scales.

1104 D.7 Proof of Theorem 6

First construction We construct an instance with m types. Since this is a worst-case lower bound,
 it suffices to consider the single-period case $T = 1$. We can fill $T \geq 2$ with vacant jobs. We first
 define the m types of customers. For each $j \in [m]$, define

$$(n_j, \lambda_j) = (j, m + 1 - j).$$

Then

$$n_j = j \leq m = \lfloor \epsilon B \rfloor \leq \epsilon B,$$

so the small-job condition is satisfied for every type. Define

$$P_j = n_j \lambda_j = j(m + 1 - j),$$

and set the value of type j to be

$$v_j = P_j 2^j.$$

We place probability mass

$$\pi_j = \frac{1}{Z P_j 2^j}, \quad Z = \sum_{r=1}^m \frac{1}{P_r 2^r}.$$

Since

$$\sum_{j=1}^m \pi_j = \frac{1}{Z} \sum_{j=1}^m \frac{1}{P_j 2^j} = 1,$$

this indeed defines a distribution $\mathcal{D}$ over the m types. We claim that the points (n_j, λ_j) are pairwise incomparable under the coordinate-wise partial order. Indeed, if $i < j$, then

$$n_i = i < j = n_j, \quad \lambda_i = m + 1 - i > m + 1 - j = \lambda_j.$$

So neither $(n_i, \lambda_i) \leq (n_j, \lambda_j)$ nor $(n_j, \lambda_j) \leq (n_i, \lambda_i)$ holds coordinate-wise. Define a pricing rule p^* on the support by

$$p^*(n_j, \lambda_j) = v_j \quad \text{for every } j \in [m].$$

Since the support points are pairwise incomparable, this assignment is consistent with monotonicity. One explicit nondecreasing extension to all $(n, \lambda) \in \mathbb{N}^2$ is

$$p^*(n, \lambda) = \max\{v_j : n \geq n_j, \lambda \geq \lambda_j\},$$

with the convention that the maximum over the empty set is 0. By construction, for each support point (n_j, λ_j) ,

$$v_j \geq p^*(n_j, \lambda_j) = v_j,$$

so type j buys with probability 1. Since $T = 1$, every submitted job is accepted, and therefore the expected revenue of p^* is

$$\text{Rev}_{\mathcal{D}}(p^*) = \sum_{j=1}^m \pi_j p^*(n_j, \lambda_j) = \sum_{j=1}^m \pi_j v_j.$$

Substituting the definitions of π_j and v_j ,

$$\sum_{j=1}^m \pi_j v_j = \sum_{j=1}^m \frac{1}{Z P_j 2^j} \cdot P_j 2^j = \sum_{j=1}^m \frac{1}{Z} = \frac{m}{Z}.$$

Hence

$$\text{OPT}_{\mathcal{D}} \geq \text{Rev}_{\mathcal{D}}(p^*) = \frac{m}{Z}.$$

Now consider an arbitrary linear price

$$p_k(n, \lambda) = k n \lambda, \quad k \geq 0.$$

For type j , the quoted price is

$$p_k(n_j, \lambda_j) = k P_j.$$

Since type j has value $v_j = P_j 2^j$, it buys if and only if

$$v_j \geq p_k(n_j, \lambda_j) \iff P_j 2^j \geq k P_j \iff 2^j \geq k.$$

Thus, for a given k , the set of buying types is always a suffix of $\{1, \dots, m\}$. If $k > 2^m$, then no type buys and the revenue is 0. Otherwise, let $\ell \in [m]$ be such that

$$2^{\ell-1} < k \leq 2^\ell.$$

Then exactly the types $j \geq \ell$ buy. Hence the expected revenue is

$$\text{Rev}_{\mathcal{D}}(p_k) = \sum_{j=\ell}^m \pi_j k P_j.$$

Using $k \leq 2^\ell$, we obtain

$$\text{Rev}_{\mathcal{D}}(p_k) \leq \sum_{j=\ell}^m \pi_j 2^\ell P_j.$$

Substituting $\pi_j = 1/(Z P_j 2^j)$,

$$\text{Rev}_{\mathcal{D}}(p_k) \leq \sum_{j=\ell}^m \frac{1}{Z P_j 2^j} \cdot 2^\ell P_j = \frac{2^\ell}{Z} \sum_{j=\ell}^m 2^{-j}.$$

Since

$$\sum_{j=\ell}^m 2^{-j} \leq \sum_{j=\ell}^{\infty} 2^{-j} = 2^{-\ell+1},$$

it follows that

$$\text{Rev}_{\mathcal{D}}(p_k) \leq \frac{2^\ell}{Z} \cdot 2^{-\ell+1} = \frac{2}{Z}.$$

Because this bound holds for every $k \geq 0$,

$$\max_{k \geq 0} \text{Rev}_{\mathcal{D}}(p_k) \leq \frac{2}{Z}.$$

Combining the lower bound for the optimal nondecreasing rule and the upper bound for the best linear rule gives

$$\frac{\max_{k \geq 0} \text{Rev}_{\mathcal{D}}(p_k)}{\text{OPT}_{\mathcal{D}}} \leq \frac{2/Z}{m/Z} = \frac{2}{m}.$$

Recalling that $m = \lfloor \epsilon B \rfloor$, we conclude

$$\frac{\max_{k \geq 0} \text{Rev}_{\mathcal{D}}(p_k)}{\text{OPT}_{\mathcal{D}}} \leq \frac{2}{\lfloor \epsilon B \rfloor}.$$

Second construction. Let

$$m_2 = \lceil \log B \rceil, \quad n_0 = \lfloor \epsilon B \rfloor.$$

Using similar proof idea for the Proof of Theorem 3, we define D_2 by

$$\Pr_{D_2}((V, N, \Lambda) = (1, n_0, 2^t)) = \frac{1}{m_2}, \quad t \in [m_2].$$

Thus $N = n_0 \leq \epsilon B$ and $\Lambda \geq 2$ almost surely. The nondecreasing price

$$p^*(n, \lambda) = 1$$

extracts revenue 1 from every type, and no price can extract more than the value $V = 1$. Hence

$$\text{OPT}_{D_2} = 1.$$

For a linear price $p_k(n, \lambda) = kn\lambda$, type t buys iff

$$1 \geq kn_0 2^t.$$

Let

$$t^* = \max\{t \in [m_2] : kn_0 2^t \leq 1\},$$

with $t^* = 0$ if the set is empty. Then

$$\text{Rev}_{D_2}(p_k) = \frac{1}{m_2} \sum_{t=1}^{t^*} kn_0 2^t.$$

If $t^* = 0$, this is zero. Otherwise,

$$kn_0 2^{t^*} \leq 1,$$

so

$$kn_0 \leq 2^{-t^*}.$$

Therefore

$$\text{Rev}_{D_2}(p_k) = \frac{kn_0}{m_2} (2^{t^*+1} - 2) < \frac{2}{m_2}.$$

Thus

$$\text{LinOPT}_{D_2} \leq \frac{2}{m_2}.$$

Since $m_2 = \lceil \log B \rceil \geq \log B$,

$$\frac{\text{LinOPT}_{D_2}}{\text{OPT}_{D_2}} \leq \frac{2}{\log B}.$$

Finally, choose $D = D_1$ if

$$\frac{2}{\lfloor \epsilon B \rfloor} \leq \frac{2}{\log B},$$

and choose $D = D_2$ otherwise. Then

$$\frac{\text{LinOPT}_D}{\text{OPT}_D} \leq \min \left\{ \frac{2}{\lfloor \epsilon B \rfloor}, \frac{2}{\log B} \right\}.$$

1105

□

1106 There's an alternative proof for the bound $O(\frac{1}{\log B})$. This version allows the platform to choose a
1107 history-dependent coefficient k_t each period before seeing the current request.

Proof. Proof. Fix an integer $m \geq 2$. Let

$$H = B = 2^m.$$

Define the common i.i.d. distribution D by

$$\Pr_D((V, N, \Lambda) = (1, 1, 2^s)) = \frac{1}{m}, \quad s \in [m].$$

Thus $V = 1$, $N = 1$, and $\Lambda \geq 2$ almost surely. For every fixed $\epsilon \in (0, 1)$ and all sufficiently large B ,

$$N = 1 \leq \epsilon B.$$

The capacity constraint never binds. Indeed, at any time there are at most H active jobs, and each active job uses one GPU. Therefore the total active demand is at most

$$H = B.$$

1108 Hence every submitted job is feasible.

The nondecreasing price

$$p^*(n, \lambda) = 1$$

makes every customer buy, so

$$\text{Rev}_D(p^*) = H.$$

Since each customer has value 1, no pricing rule can extract more than one unit of revenue per period. Therefore

$$\text{OPT}_D = H.$$

Now allow dynamic linear prices

$$p_t(n, \lambda) = k_t n \lambda,$$

1109 where k_t may depend on the history before period t , but not on the current request. Negative prices
1110 cannot improve revenue, so we restrict to $k_t \geq 0$.

Let $\mathcal{F}_{t-1}$ be the history before period t . Conditional on $\mathcal{F}_{t-1}$, the current type is still uniform over $s \in [m]$. For type s , the posted price is

$$p_t(1, 2^s) = k_t 2^s,$$

and the customer buys iff

$$1 \geq k_t 2^s.$$

Thus

$$\mathbb{E}[\text{Rev}_t \mid \mathcal{F}_{t-1}] = \frac{1}{m} \sum_{s=1}^m k_t 2^s \mathbf{1}\{k_t 2^s \leq 1\}.$$

If no type buys, this conditional expectation is zero. Otherwise define

$$s^* = \max\{s \in [m] : k_t 2^s \leq 1\}.$$

Then

$$\mathbb{E}[\text{Rev}_t \mid \mathcal{F}_{t-1}] = \frac{k_t}{m} \sum_{s=1}^{s^*} 2^s = \frac{k_t}{m} (2^{s^*+1} - 2).$$

Since

$$k_t 2^{s^*} \leq 1,$$

we have

$$k_t \leq 2^{-s^*}.$$

Therefore

$$\mathbb{E}[\text{Rev}_t \mid \mathcal{F}_{t-1}] < \frac{2}{m}.$$

This bound holds for every history and every admissible choice of k_t . Summing over periods gives

$$\text{DynLinOPT}_D \leq \sum_{t=1}^H \frac{2}{m} = \frac{2H}{m}.$$

Since $\text{OPT}_D = H$,

$$\frac{\text{DynLinOPT}_D}{\text{OPT}_D} \leq \frac{2}{m}.$$

Finally, because $B = 2^m$,

$$m = \log_2 B.$$

Hence

$$\frac{\text{DynLinOPT}_D}{\text{OPT}_D} \leq \frac{2}{\log_2 B} = O\left(\frac{1}{\log B}\right).$$

1111

□

1112 D.8 Proof of Theorem 7

1113 **Model** Time is indexed by $t = 1, \dots, T$. The platform has capacity B GPUs per time step. At each
1114 time t , an independent random triple $(V_t, N_t, \Lambda_t) \sim \mathcal{D}_t$ arrives. Define

$$w_{t,n,\lambda} = \Pr[N_t = n, \Lambda_t = \lambda], \quad F_{t,n,\lambda}(v) = \Pr[V_t \leq v \mid N_t = n, \Lambda_t = \lambda], \quad S_{t,n,\lambda}(p) = 1 - F_{t,n,\lambda}(p).$$

1115 **Revenue curve in the regular continuous case.** For each (t, n, λ) , define

$$p_{t,n,\lambda}(q) = F_{t,n,\lambda}^{-1}(1 - q), \quad q \in [0, 1],$$

1116 and

$$R_{t,n,\lambda}(q) = q p_{t,n,\lambda}(q) = q F_{t,n,\lambda}^{-1}(1 - q), \quad q \in [0, 1].$$

1117 Set $R_{t,n,\lambda}(0) = 0$.

1118 **Lemma 5.** For every (t, n, λ) , the function $R_{t,n,\lambda}$ is concave on $[0, 1]$.

1119 *Proof.* Fix (t, n, λ) and suppress subscripts for readability. Let

$$v(q) = F^{-1}(1 - q), \quad R(q) = qv(q).$$

1120 Since $1 - F(v(q)) = q$, differentiating both sides with respect to q gives $-f(v(q))v'(q) = 1$.

1121 Therefore

$$R'(q) = v(q) + qv'(q) = v(q) - \frac{q}{f(v(q))} = v(q) - \frac{1 - F(v(q))}{f(v(q))} = \phi(v(q)).$$

1122 Now $v(q) = F^{-1}(1 - q)$ is decreasing in q , and regularity means $\phi(\cdot)$ is increasing in its argument.

1123 Hence the composition $\phi(v(q))$ is decreasing in q . Therefore $R'(q)$ is decreasing, which implies that

1124 R is concave on $[0, 1]$. □

1125 **Time-expanded upper bound.** Consider the benchmark

$$\begin{aligned} \text{UB}^* = & \max_{0 \leq x_{t,n,\lambda} \leq 1} \sum_{t=1}^T \sum_{n,\lambda} w_{t,n,\lambda} R_{t,n,\lambda}(x_{t,n,\lambda}) \\ \text{s.t.} \quad & \sum_{\tau: \tau \leq t \leq \tau + \lambda - 1} \sum_{n,\lambda} w_{\tau,n,\lambda} n x_{\tau,n,\lambda} \leq B, \quad t = 1, \dots, T. \end{aligned} \quad (\text{UB})$$

1126 By Lemma 5, this is a convex optimization problem.

1127 **Algorithm Throttled LP-Menu Posted Pricing (TCMPP).** Let x^* be an optimal solution to
 1128 (UB). Since we are in the regular continuous case, each coordinate $x_{t,n,\lambda}^*$ is implemented by the
 1129 deterministic posted price

$$p_t^*(n, \lambda) = F_{t,n,\lambda}^{-1}(1 - x_{t,n,\lambda}^*).$$

1130 Thus

$$\Pr[V_t \geq p_t^*(n, \lambda) \mid N_t = n, \Lambda_t = \lambda] = x_{t,n,\lambda}^*,$$

1131 and

$$p_t^*(n, \lambda) \cdot \Pr[V_t \geq p_t^*(n, \lambda) \mid N_t = n, \Lambda_t = \lambda] = R_{t,n,\lambda}(x_{t,n,\lambda}^*).$$

1132 Fix $\gamma \in (0, 1 - \varepsilon)$. At each time t :

- 1133 1. Draw $O_t \sim \text{Bernoulli}(\gamma)$, independently across t .
- 1134 2. If $O_t = 0$, post $p_t(\cdot, \cdot) \equiv +\infty$.
- 1135 3. If $O_t = 1$, post the deterministic menu $p_t(\cdot, \cdot) = p_t^*(\cdot, \cdot)$.
- 1136 4. If customer t buys, accept if and only if capacity is feasible.

1137 *Proof.* Let $A_t \in \{0, 1\}$ denote the actual acceptance indicator under TCMPP. We divide the proof
 1138 into two parts. We first prove the Lemma that $\text{OPT}_{\mathcal{D}} \leq \text{UB}^*$.

1139 **Lemma 6.** $\text{OPT}_{\mathcal{D}} \leq \text{UB}^*$.

1140 *Proof.* Let π be any feasible dynamic posted-pricing policy. At time t , before seeing customer t , the
 1141 policy observes a history H_t and posts a menu. For each type (n, λ) , denote the posted menu entry by

$$P_t^\pi(H_t; n, \lambda).$$

1142 Also let

$$C_t^\pi(H_t; n, \lambda) \in \{0, 1\}$$

1143 be the indicator that, given history H_t , the policy would actually admit a type (n, λ) buyer if that
 1144 buyer is willing to pay. This captures both out of service and capacity feasibility.

1145 For fixed H_t and fixed type (n, λ) , define the conditional acceptance probability

$$Q_t^\pi(H_t; n, \lambda) = C_t^\pi(H_t; n, \lambda) S_{t,n,\lambda}(P_t^\pi(H_t; n, \lambda)).$$

1146 Now set

$$x_{t,n,\lambda}^\pi = \mathbb{E}[Q_t^\pi(H_t; n, \lambda)].$$

1147 Since the acceptance event at time t is exactly the event that the customer buys and the platform
 1148 admits the job, we also have

$$x_{t,n,\lambda}^\pi = \Pr_\pi(A_t = 1 \mid N_t = n, \Lambda_t = \lambda).$$

1149 **Lemma 7.** For every t, n, λ ,

$$\mathbb{E}_\pi[\text{revenue at time } t \mid N_t = n, \Lambda_t = \lambda] \leq R_{t,n,\lambda}(x_{t,n,\lambda}^\pi).$$

1150 *Proof.* Fix t, n, λ . Conditional on H_t and on the event $(N_t, \Lambda_t) = (n, \lambda)$, the realized revenue at
 1151 time t equals

$$C_t^\pi(H_t; n, \lambda) P_t^\pi(H_t; n, \lambda) S_{t,n,\lambda}(P_t^\pi(H_t; n, \lambda)).$$

1152 There are two cases. If $C_t^\pi(H_t; n, \lambda) = 0$, then the realized revenue is 0, and also $Q_t^\pi(H_t; n, \lambda) = 0$.
 1153 Since $R_{t,n,\lambda}(0) = 0$, we have

$$\text{revenue at time } t = R_{t,n,\lambda}(Q_t^\pi(H_t; n, \lambda)).$$

1154 If $C_t^\pi(H_t; n, \lambda) = 1$, then

$$Q_t^\pi(H_t; n, \lambda) = S_{t,n,\lambda}(P_t^\pi(H_t; n, \lambda)).$$

1155 Because $F_{t,n,\lambda}$ is continuous and strictly increasing,

$$P_t^\pi(H_t; n, \lambda) = F_{t,n,\lambda}^{-1}(1 - Q_t^\pi(H_t; n, \lambda)).$$

1156 Hence

$$P_t^\pi(H_t; n, \lambda) S_{t,n,\lambda}(P_t^\pi(H_t; n, \lambda)) = R_{t,n,\lambda}(Q_t^\pi(H_t; n, \lambda)).$$

1157 Therefore, in all cases,

$$\text{revenue at time } t = R_{t,n,\lambda}(Q_t^\pi(H_t; n, \lambda)) \quad \text{conditional on } (H_t, N_t = n, \Lambda_t = \lambda).$$

1158 Taking expectation over H_t and using concavity of $R_{t,n,\lambda}$ from Lemma 5 and Jensen's inequality,

$$\mathbb{E}_\pi[\text{revenue at time } t \mid N_t = n, \Lambda_t = \lambda] = \mathbb{E}[R_{t,n,\lambda}(Q_t^\pi(H_t; n, \lambda))] \leq R_{t,n,\lambda}(\mathbb{E}[Q_t^\pi(H_t; n, \lambda)]).$$

1159 By definition, $\mathbb{E}[Q_t^\pi(H_t; n, \lambda)] = x_{t,n,\lambda}^\pi$, so the claim follows. $\square$

1160 **Lemma 8.** $x^\pi = (x_{t,n,\lambda}^\pi)$ is feasible for (UB).

1161 *Proof.* Because policy π is feasible pathwise, for every slot s ,

$$\sum_{t=1}^T A_t N_t \mathbf{1}\{t \leq s \leq t + \Lambda_t - 1\} \leq B.$$

1162 Taking expectation,

$$\mathbb{E}\left[\sum_{t=1}^T A_t N_t \mathbf{1}\{t \leq s \leq t + \Lambda_t - 1\}\right] \leq B.$$

1163 For each fixed t ,

$$\mathbb{E}[A_t N_t \mathbf{1}\{t \leq s \leq t + \Lambda_t - 1\}] = \sum_{n,\lambda} w_{t,n,\lambda} n \mathbf{1}\{t \leq s \leq t + \lambda - 1\} \mathbf{Pr}_\pi(A_t = 1 \mid N_t = n, \Lambda_t = \lambda).$$

1164 Using the definition of $x_{t,n,\lambda}^\pi$, this becomes

$$\mathbb{E}[A_t N_t \mathbf{1}\{t \leq s \leq t + \Lambda_t - 1\}] = \sum_{n,\lambda} w_{t,n,\lambda} n \mathbf{1}\{t \leq s \leq t + \lambda - 1\} x_{t,n,\lambda}^\pi.$$

1165 Summing over t yields

$$\sum_{t: t \leq s \leq t + \lambda - 1} \sum_{n,\lambda} w_{t,n,\lambda} n x_{t,n,\lambda}^\pi \leq B.$$

1166 Thus x^π is feasible for (UB). $\square$

1167 Applying Lemma 7 to an optimal policy π^* and then Lemma 8, we obtain

$$\text{OPT}_{\mathcal{D}} = \mathbb{E}[\text{Rev}_{\pi^*}] \leq \sum_{t,n,\lambda} w_{t,n,\lambda} R_{t,n,\lambda}(x_{t,n,\lambda}^{\pi^*}) \leq \text{UB}^*.$$

1168 This proves

$$\text{OPT}_{\mathcal{D}} \leq \text{UB}^*.$$

1169 $\square$

1170 Define the tentative acceptance indicator

$$A_t^0 = O_t \mathbf{1}\{V_t \geq p_t^*(N_t, \Lambda_t)\}.$$

1171 This is the acceptance indicator of the coupled process that ignores capacity constraints. The actual
1172 TCMPP acceptance indicator is

$$A_t = A_t^0 \mathbf{1}\{\ell_t + N_t \leq B\},$$

1173 where ℓ_t is the actual current load.

1174 Define tentative and actual revenues by

$$Y_t^0 = p_t^*(N_t, \Lambda_t) A_t^0, \quad Y_t = p_t^*(N_t, \Lambda_t) A_t.$$

1175 Clearly,

$$A_t \leq A_t^0 \quad \text{for all } t.$$

Lemma 9.

$$\mathbb{E}\left[\sum_{t=1}^T Y_t^0\right] = \gamma \text{UB}^*.$$

1176 *Proof.* Fix t, n, λ . Conditional on $(N_t, \Lambda_t) = (n, \lambda)$, the event $A_t^0 = 1$ occurs exactly when $O_t = 1$
1177 and $V_t \geq p_t^*(n, \lambda)$. Therefore

$$\Pr(A_t^0 = 1 \mid N_t = n, \Lambda_t = \lambda) = \gamma x_{t,n,\lambda}^*.$$

1178 Moreover,

$$\mathbb{E}[Y_t^0 \mid N_t = n, \Lambda_t = \lambda] = \gamma p_t^*(n, \lambda) x_{t,n,\lambda}^* = \gamma R_{t,n,\lambda}(x_{t,n,\lambda}^*).$$

1179 Multiply by $w_{t,n,\lambda}$ and sum over all t, n, λ :

$$\mathbb{E}\left[\sum_{t=1}^T Y_t^0\right] = \sum_{t,n,\lambda} w_{t,n,\lambda} \mathbb{E}[Y_t^0 \mid N_t = n, \Lambda_t = \lambda] = \gamma \sum_{t,n,\lambda} w_{t,n,\lambda} R_{t,n,\lambda}(x_{t,n,\lambda}^*) = \gamma \text{UB}^*.$$

1180 □

1181 Define the tentative current load at time t by

$$\ell_t^0 = \sum_{\tau=1}^{t-1} A_\tau^0 N_\tau \mathbf{1}\{\tau + \Lambda_\tau - 1 \geq t\}.$$

1182 **Lemma 10.** For every t ,

$$\mathbb{E}[\ell_t^0] \leq \gamma B.$$

1183 *Proof.* By definition,

$$\ell_t^0 = \sum_{\tau=1}^{t-1} A_\tau^0 N_\tau \mathbf{1}\{\tau \leq t \leq \tau + \Lambda_\tau - 1\}.$$

1184 Hence

$$\mathbb{E}[\ell_t^0] = \sum_{\tau=1}^{t-1} \mathbb{E}[A_\tau^0 N_\tau \mathbf{1}\{\tau \leq t \leq \tau + \Lambda_\tau - 1\}].$$

1185 For each fixed τ , conditioning on $(N_\tau, \Lambda_\tau) = (n, \lambda)$ gives

$$\mathbb{E}[A_\tau^0 N_\tau \mathbf{1}\{\tau \leq t \leq \tau + \Lambda_\tau - 1\}] = \sum_{n,\lambda} w_{\tau,n,\lambda} n \mathbf{1}\{\tau \leq t \leq \tau + \lambda - 1\} \gamma x_{\tau,n,\lambda}^*.$$

1186 Therefore

$$\mathbb{E}[\ell_t^0] = \gamma \sum_{\tau=1}^{t-1} \sum_{n,\lambda} w_{\tau,n,\lambda} n x_{\tau,n,\lambda}^* \mathbf{1}\{\tau \leq t \leq \tau + \lambda - 1\}.$$

1187 The right-hand side is at most

$$\gamma \sum_{\tau=1}^T \sum_{n,\lambda} w_{\tau,n,\lambda} n x_{\tau,n,\lambda}^* \mathbf{1}\{\tau \leq t \leq \tau + \lambda - 1\},$$

1188 which is at most γB by feasibility of x^* in the slot- t constraint of (UB). □

1189 **Lemma 11.** For every t ,

$$\Pr(\ell_t^0 > B - N_t) \leq \frac{\gamma}{1 - \varepsilon}.$$

1190 Consequently,

$$\Pr(\ell_t^0 \leq B - N_t) \geq 1 - \frac{\gamma}{1 - \varepsilon}.$$

1191 *Proof.* Since $N_t \leq \varepsilon B$, so

$$B - N_t \geq (1 - \varepsilon)B.$$

1192 Thus

$$\Pr(\ell_t^0 > B - N_t) \leq \Pr(\ell_t^0 > (1 - \varepsilon)B).$$

1193 Applying Markov's inequality and then Lemma 10,

$$\Pr(\ell_t^0 > (1 - \varepsilon)B) \leq \frac{\mathbb{E}[\ell_t^0]}{(1 - \varepsilon)B} \leq \frac{\gamma B}{(1 - \varepsilon)B} = \frac{\gamma}{1 - \varepsilon}.$$

1194 The second inequality follows by complement. $\square$

1195 **Lemma 12.** For every t ,

$$\mathbb{E}[Y_t] \geq \left(1 - \frac{\gamma}{1 - \varepsilon}\right) \mathbb{E}[Y_t^0].$$

1196 Hence

$$\mathbb{E}\left[\sum_{t=1}^T Y_t\right] \geq \left(1 - \frac{\gamma}{1 - \varepsilon}\right) \mathbb{E}\left[\sum_{t=1}^T Y_t^0\right].$$

1197 *Proof.* For every $\tau < t$ we have $A_\tau \leq A_\tau^0$, so

$$\ell_t = \sum_{\tau=1}^{t-1} A_\tau N_\tau \mathbf{1}\{\tau + \Lambda_\tau - 1 \geq t\} \leq \sum_{\tau=1}^{t-1} A_\tau^0 N_\tau \mathbf{1}\{\tau + \Lambda_\tau - 1 \geq t\} = \ell_t^0.$$

1198 Therefore, if $A_t^0 = 1$ and $\ell_t^0 \leq B - N_t$, then also $\ell_t \leq B - N_t$, so the actual algorithm accepts the
1199 job. Hence

$$A_t \geq A_t^0 \mathbf{1}\{\ell_t^0 \leq B - N_t\}.$$

1200 Multiplying by the nonnegative posted price,

$$Y_t \geq Y_t^0 \mathbf{1}\{\ell_t^0 \leq B - N_t\}. \quad (9)$$

1201 Now condition on (N_t, Λ_t) . The random variable Y_t^0 depends only on time- t randomness, namely
1202 O_t and V_t , together with (N_t, Λ_t) . The random variable ℓ_t^0 depends only on randomness from times
1203 $1, \dots, t-1$. By independence across time,

$$Y_t^0 \perp \ell_t^0 \quad \text{conditional on } (N_t, \Lambda_t).$$

1204 Using (9),

$$\mathbb{E}[Y_t \mid N_t, \Lambda_t] \geq \mathbb{E}[Y_t^0 \mathbf{1}\{\ell_t^0 \leq B - N_t\} \mid N_t, \Lambda_t].$$

1205 By conditional independence,

$$\mathbb{E}[Y_t^0 \mathbf{1}\{\ell_t^0 \leq B - N_t\} \mid N_t, \Lambda_t] = \mathbb{E}[Y_t^0 \mid N_t, \Lambda_t] \Pr(\ell_t^0 \leq B - N_t \mid N_t, \Lambda_t).$$

1206 Lemma 11 yields

$$\Pr(\ell_t^0 \leq B - N_t \mid N_t, \Lambda_t) \geq 1 - \frac{\gamma}{1 - \varepsilon}.$$

1207 Therefore

$$\mathbb{E}[Y_t \mid N_t, \Lambda_t] \geq \left(1 - \frac{\gamma}{1 - \varepsilon}\right) \mathbb{E}[Y_t^0 \mid N_t, \Lambda_t].$$

1208 Unconditioning proves the first inequality, and summing over t proves the second. $\square$

1209 Combining Lemma 12 with Lemma 9, we obtain

$$\mathbb{E}[\text{Rev}_{\text{TCMPP}}] = \mathbb{E}\left[\sum_{t=1}^T Y_t\right] \geq \left(1 - \frac{\gamma}{1-\varepsilon}\right) \mathbb{E}\left[\sum_{t=1}^T Y_t^0\right] = \gamma \left(1 - \frac{\gamma}{1-\varepsilon}\right) \text{UB}^*.$$

1210 Together with Lemma 6 this implies

$$\mathbb{E}[\text{Rev}_{\text{TCMPP}}] \geq \gamma \left(1 - \frac{\gamma}{1-\varepsilon}\right) \text{OPT}_{\mathcal{D}}.$$

1211 Finally define

$$f(\gamma) = \gamma \left(1 - \frac{\gamma}{1-\varepsilon}\right), \quad 0 \leq \gamma \leq 1 - \varepsilon.$$

1212 The maximizer is $\gamma^* = \frac{1-\varepsilon}{2}$, and $f(\gamma^*) = \frac{1-\varepsilon}{2} \cdot \frac{1}{2} = \frac{1-\varepsilon}{4}$. Therefore

$$\mathbb{E}[\text{Rev}_{\text{TCMPP}}] \geq \frac{1-\varepsilon}{4} \text{OPT}_{\mathcal{D}}.$$

1213 This completes the proof. □

1214 D.9 Proof of Theorem 8

1215 Let the finite type space be

$$\mathcal{K} = \{k = (n_k, \lambda_k)\}.$$

1216 At each time t , the arriving customer has type $K_t \in \mathcal{K}$ with

$$\Pr(K_t = k) = w_{t,k},$$

1217 independently across t . Conditional on $K_t = k$, the value v_t is drawn independently from an atomless
1218 distribution $F_{t,k}$. Define

$$\bar{F}_{t,k}(p) = 1 - F_{t,k}(p).$$

1219 Assume the revenue curve

$$R_{t,k}(x) = x F_{t,k}^{-1}(1 - x), \quad x \in [0, 1],$$

1220 is concave for every (t, k) (the usual regularity assumption).

1221 For $\eta \in (0, 1)$, consider the slackened fluid LP

$$\begin{aligned} U^\eta &= \max_{0 \leq x_{t,k} \leq 1} \sum_{t=1}^T \sum_{k \in \mathcal{K}} w_{t,k} R_{t,k}(x_{t,k}) \\ \text{s.t. } &\sum_{\tau=1}^u \sum_{k \in \mathcal{K}} n_k \mathbf{1}\{u - \tau + 1 \leq \lambda_k\} w_{\tau,k} x_{\tau,k} \leq (1 - \eta)B, \quad \forall u = 1, \dots, T. \end{aligned} \tag{10}$$

1222 Let $x^* = (x_{t,k}^*)$ be an optimal solution of (10) and define the base prices

$$p_{t,k} = F_{t,k}^{-1}(1 - x_{t,k}^*).$$

1223 **Online policy π^η .** At time t , if type k is feasible given the current occupancy, offer price $p_{t,k}$;
1224 otherwise offer $+\infty$. Equivalently, the menu posted at time t is

$$\hat{p}_t(k) = \begin{cases} p_{t,k}, & \text{if the system has at least } n_k \text{ free GPUs at time } t, \\ +\infty, & \text{otherwise.} \end{cases}$$

1225 **D.10 Proof of Theorem 8**

1226 By Lemma 6, the following lemma holds.

1227 **Lemma 13.** *Let U^0 denote the value of (LP_η) with $\eta = 0$. Then*

$$\text{OPT}_\mathcal{D} \leq U^0.$$

Lemma 14 (The slackened LP loses only a factor $1 - \eta$).

$$U^\eta \geq (1 - \eta) U^0.$$

1228 *Proof.* Let x^0 be an optimal solution of LP_0 .

1229 Because the constraints are linear, $(1 - \eta)x^0$ is feasible for LP_η : for every u ,

$$\sum_{\tau=1}^u \sum_k n_k w_{\tau,k} [(1-\eta)x_{\tau,k}^0] \mathbf{1}\{u-\tau+1 \leq \lambda_k\} = (1-\eta) \sum_{\tau=1}^u \sum_k n_k w_{\tau,k} x_{\tau,k}^0 \mathbf{1}\{u-\tau+1 \leq \lambda_k\} \leq (1-\eta)B.$$

1230 Therefore

$$U^\eta \geq \sum_{t,k} w_{t,k} R_{t,k}((1-\eta)x_{t,k}^0). \quad (11)$$

1231 Since each $R_{t,k}$ is concave and $R_{t,k}(0) = 0$,

$$R_{t,k}((1-\eta)x) = R_{t,k}((1-\eta)x + \eta \cdot 0) \geq (1-\eta)R_{t,k}(x) + \eta R_{t,k}(0) = (1-\eta)R_{t,k}(x).$$

1232 Applying this to (11),

$$U^\eta \geq (1-\eta) \sum_{t,k} w_{t,k} R_{t,k}(x_{t,k}^0) = (1-\eta)U^0.$$

1233

□

1234 **Lemma 15** (Local blocking probability is exponentially small). *Fix a time t and a type k . Under the*
1235 *policy π^η ,*

$$\Pr(\text{type } k \text{ is infeasible at time } t) \leq \delta_\eta.$$

1236 *Proof.* We couple the actual system with a tentative system that ignores feasibility constraints.

1237 Define the tentative acceptance indicator

$$\tilde{A}_\tau = \mathbf{1}\{v_\tau \geq p_{\tau,K_\tau}\}, \quad \tau = 1, \dots, T.$$

1238 This is what happens if we always offer the base LP price p_{τ,K_τ} , regardless of remaining capacity.

1239 Define the actual acceptance indicator

$$A_\tau = \mathbf{1}\{v_\tau \geq p_{\tau,K_\tau}\} \mathbf{1}\{Y_\tau + n_{K_\tau} \leq B\},$$

1240 where Y_τ is the actual occupancy right before customer τ is processed.

1241 Clearly, pathwise,

$$A_\tau \leq \tilde{A}_\tau \quad \text{for every } \tau. \quad (12)$$

1242 Define the actual occupancy at time t :

$$Y_t = \sum_{\tau=1}^{t-1} n_{K_\tau} A_\tau \mathbf{1}\{t-\tau+1 \leq \lambda_{K_\tau}\}.$$

1243 Define the tentative occupancy at time t :

$$\tilde{Y}_t = \sum_{\tau=1}^{t-1} n_{K_\tau} \tilde{A}_\tau \mathbf{1}\{t-\tau+1 \leq \lambda_{K_\tau}\}.$$

1244 By (12),

$$Y_t \leq \tilde{Y}_t \quad \text{pathwise.} \quad (13)$$

1245 A type- k job is infeasible at time t exactly when

$$Y_t > B - n_k.$$

1246 Hence, by (13),

$$\Pr(\text{type } k \text{ infeasible at time } t) \leq \Pr(\tilde{Y}_t > B - n_k). \quad (14)$$

1247 So it remains to bound $\Pr(\tilde{Y}_t > B - n_k)$.

1248 For each $\tau < t$, define

$$W_\tau^{(t)} = n_{K_\tau} \tilde{A}_\tau \mathbf{1}\{t - \tau + 1 \leq \lambda_{K_\tau}\}. \quad (15)$$

1249 Then

$$\tilde{Y}_t = \sum_{\tau=1}^{t-1} W_\tau^{(t)}.$$

1250 Because arrivals and values are independent across time, the random variables $W_1^{(t)}, \dots, W_{t-1}^{(t)}$ are
1251 independent. Also,

$$0 \leq W_\tau^{(t)} \leq \bar{N}.$$

1252 Let

$$\mu_t = \mathbb{E}[\tilde{Y}_t].$$

1253 Using (15) and the definition of the prices,

$$\mathbb{E}[W_\tau^{(t)}] = \sum_{j \in \mathcal{K}} n_j w_{\tau,j} x_{\tau,j}^* \mathbf{1}\{t - \tau + 1 \leq \lambda_j\}.$$

1254 Therefore

$$\mu_t = \sum_{\tau=1}^{t-1} \sum_{j \in \mathcal{K}} n_j w_{\tau,j} x_{\tau,j}^* \mathbf{1}\{t - \tau + 1 \leq \lambda_j\}.$$

1255 Now compare with the LP constraint for calendar time $u = t$:

$$\sum_{\tau=1}^t \sum_{j \in \mathcal{K}} n_j w_{\tau,j} x_{\tau,j}^* \mathbf{1}\{t - \tau + 1 \leq \lambda_j\} \leq (1 - \eta)B.$$

1256 Dropping the nonnegative $\tau = t$ term yields

$$\mu_t \leq (1 - \eta)B. \quad (16)$$

1257 We next bound the variance. Let

$$\sigma_t^2 = \text{Var}(\tilde{Y}_t) = \sum_{\tau=1}^{t-1} \text{Var}(W_\tau^{(t)}). \quad (17)$$

1258 Since $W_\tau^{(t)} \geq 0$,

$$\text{Var}(W_\tau^{(t)}) \leq \mathbb{E}[(W_\tau^{(t)})^2].$$

1259 Using $0 \leq W_\tau^{(t)} \leq \bar{N}$,

$$(W_\tau^{(t)})^2 \leq \bar{N} W_\tau^{(t)}.$$

1260 Hence

$$\sigma_t^2 \leq \sum_{\tau=1}^{t-1} \mathbb{E}[(W_\tau^{(t)})^2] \leq \bar{N} \sum_{\tau=1}^{t-1} \mathbb{E}[W_\tau^{(t)}] = \bar{N} \mu_t \leq,$$

1261 where the last inequality uses (16).

1262 Define

$$a_{t,k} = B - n_k - \mu_t.$$

1263 By (16),

$$a_{t,k} \geq B - n_k - (1 - \eta)B = \eta B - n_k \geq \eta B - \bar{N}. \quad (18)$$

1264 Our assumption $\bar{N} < \eta B$ implies $a_{t,k} > 0$.

1265 Now

$$\Pr(\tilde{Y}_t > B - n_k) = \Pr(\tilde{Y}_t - \mu_t > a_{t,k}).$$

1266 Since $\tilde{Y}_t$ is a sum of independent random variables bounded by $\bar{N}$, Bernstein gives

$$\Pr(\tilde{Y}_t - \mu_t > a_{t,k}) \leq \exp\left(-\frac{a_{t,k}^2}{2(\sigma_t^2 + \frac{a_{t,k}}{3})}\right).$$

1267 Using (17),

$$\Pr(\tilde{Y}_t - \mu_t > a_{t,k}) \leq \exp\left(-\frac{a_{t,k}^2}{2\bar{N}(B + \frac{a_{t,k}}{3})}\right). \quad (19)$$

1268 Now define

$$a_0 = \eta B - \bar{N} > 0.$$

1269 From (18), $a_{t,k} \geq a_0$. Consider the function

$$g(a) = \frac{a^2}{B + a/3}, \quad a > 0.$$

1270 Its derivative is

$$g'(a) = \frac{2a(B + a/3) - a^2/3}{(B + a/3)^2} = \frac{2aB + a^2/3}{(B + a/3)^2} > 0.$$

1271 So g is increasing on $(0, \infty)$. Therefore,

$$\frac{a_{t,k}^2}{B + a_{t,k}/3} \geq \frac{a_0^2}{B + a_0/3}. \quad (20)$$

1272 Plugging (20) into (19),

$$\Pr(\tilde{Y}_t > B - n_k) \leq \exp\left(-\frac{a_0^2}{2\bar{N}(B + a_0/3)}\right) = \delta_\eta. \quad (21)$$

1273 Combining (14) and (21),

$$\Pr(\text{type } k \text{ infeasible at time } t) \leq \delta_\eta.$$

1274

□

Lemma 16.

$$\text{Rev}_{\mathcal{D}}(\pi^\eta) \geq (1 - \delta_\eta)U^\eta.$$

1275 *Proof.* At time t , the actual revenue is

$$\text{rev}_t = \sum_{k \in \mathcal{K}} p_{t,k} \mathbf{1}\{K_t = k\} \mathbf{1}\{v_t \geq p_{t,k}\} \mathbf{1}\{Y_t \leq B - n_k\}.$$

1276 Indeed, if type k is infeasible, the policy posts $+\infty$, which is equivalent to multiplying by $\mathbf{1}\{Y_t \leq$
1277 $B - n_k\}$.

1278 Taking expectation,

$$\mathbb{E}[\text{rev}_t] = \sum_{k \in \mathcal{K}} p_{t,k} \Pr(K_t = k, v_t \geq p_{t,k}, Y_t \leq B - n_k).$$

1279 Now Y_t depends only on times $1, \dots, t-1$, while (K_t, v_t) is independent of the past. Hence Y_t is
1280 independent of (K_t, v_t) . Therefore

$$\Pr(K_t = k, v_t \geq p_{t,k}, Y_t \leq B - n_k) = \Pr(K_t = k) \Pr(v_t \geq p_{t,k} \mid K_t = k) \Pr(Y_t \leq B - n_k).$$

1281 So

$$\mathbb{E}[\text{rev}_t] = \sum_{k \in \mathcal{K}} p_{t,k} w_{t,k} x_{t,k}^* \Pr(Y_t \leq B - n_k).$$

1282 By Lemma 3,

$$\Pr(Y_t \leq B - n_k) \geq 1 - \delta_\eta.$$

1283 Hence

$$\mathbb{E}[\text{rev}_t] \geq (1 - \delta_\eta) \sum_{k \in \mathcal{K}} p_{t,k} w_{t,k} x_{t,k}^*.$$

1284 Summing over t ,

$$\text{Rev}_{\mathcal{D}}(\pi^\eta) = \sum_{t=1}^T \mathbb{E}[\text{rev}_t] \geq (1 - \delta_\eta) \sum_{t=1}^T \sum_{k \in \mathcal{K}} p_{t,k} w_{t,k} x_{t,k}^*. \quad (22)$$

1285 But by construction

$$p_{t,k} = F_{t,k}^{-1}(1 - x_{t,k}^*),$$

1286 so

$$x_{t,k}^* p_{t,k} = R_{t,k}(x_{t,k}^*).$$

1287 Therefore

$$\sum_{t,k} p_{t,k} w_{t,k} x_{t,k}^* = \sum_{t,k} w_{t,k} R_{t,k}(x_{t,k}^*) = U^\eta. \quad (23)$$

1288 Combining (22) and (23),

$$\text{Rev}_{\mathcal{D}}(\pi^\eta) \geq (1 - \delta_\eta) U^\eta.$$

1289

□

1290 By Lemma 13, Lemma 14 and Lemma 16, we have

$$\text{Rev}_{\mathcal{D}}(\pi^\eta) \geq (1 - \delta_\eta) U^\eta \geq (1 - \delta_\eta)(1 - \eta) U^0 \geq (1 - \delta_\eta)(1 - \eta) \text{OPT}_{\mathcal{D}}.$$

1291 **D.11 How to tune the slack parameter η .**

1292 Theorem 8 yields the certified bound

$$g(\eta, \bar{N}) := (1 - \delta_\eta)(1 - \eta).$$

1293 A useful simplification is to normalize the largest job size by capacity,

$$\rho := \frac{\bar{N}}{B}.$$

1294 Substituting $\bar{N} = \rho B$ into δ_η shows that the guarantee depends on the primitive parameters only
1295 through ρ ,

$$g_\rho(\eta) = (1 - \eta) \left(1 - \exp \left(-\frac{(\eta - \rho)^2}{2\rho(1 + \frac{\eta - \rho}{3})} \right) \right), \quad \eta > \rho. \quad (24)$$

1296 This normalized form makes the design tradeoff explicit. Choosing η too close to ρ leaves too little
1297 safety margin, so the blocking term $1 - \delta_\eta$ is poor. Choosing η too large makes blocking negligible
1298 but sacrifices too much capacity through the multiplicative factor $(1 - \eta)$. The optimum is therefore
1299 interior.

1300 Figure 2 visualizes this tradeoff. The figure compares the slackened-CP guarantee $g_\rho(\eta)$ with the
1301 TCMPP guarantee $(1 - \rho)/4$, which is the appropriate comparison when one identifies the small-job
1302 parameter in Theorem 7 with $\epsilon = \rho$. Three conclusions are useful in practice.

1303 First, for each fixed ρ , the function $g_\rho(\eta)$ is numerically unimodal, so η can be tuned by a simple
1304 one-dimensional line search. Second, after optimizing over η , the slackened-CP lower bound exceeds
1305 the TCMPP bound for approximately

$$\rho \lesssim 0.12,$$

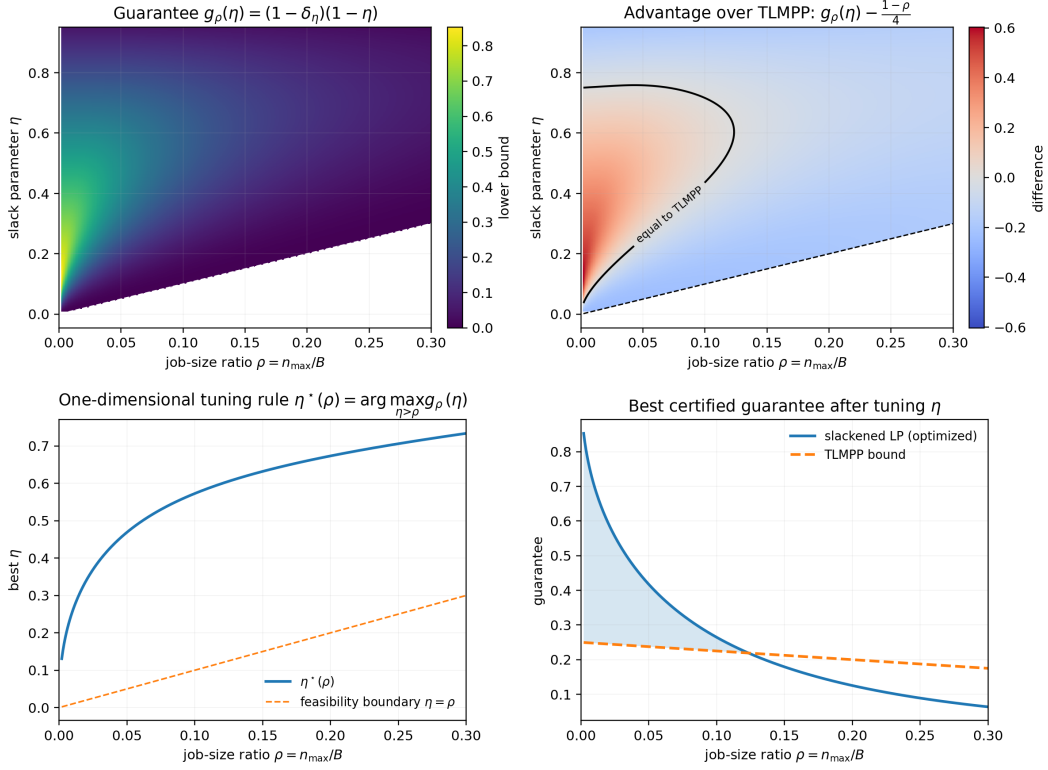


Figure 2: Phase diagram for the slackened-CP guarantee in Theorem 8. Top left: the bound $g_\rho(\eta) = (1 - \delta_\eta)(1 - \eta)$. Top right: the signed difference between the slackened-CP bound and the TCMPP guarantee $(1 - \rho)/4$; the black contour marks equality. Bottom left: the optimizer $\eta^*(\rho) = \arg \max_{\eta > \rho} g_\rho(\eta)$. Bottom right: the best certified slackened-CP guarantee after tuning η , compared with TCMPP. The plots show that the slackened-CP policy is especially attractive in the very small-job regime and that the choice of η should increase with the largest-job fraction $\rho = \bar{N}/B$.

that is, when the largest single job is at most about 12% of total capacity. In this regime the advantage can be substantial. Third, a conservative rule is to choose the smallest η that drives δ_η below a target blocking level β . For small ρ , the first-order approximation

$$\delta_\eta \approx \exp\left(-\frac{(\eta - \rho)^2}{2\rho}\right)$$

suggests the heuristic

$$\eta \approx \rho + \sqrt{2\rho \log(1/\beta)}.$$

When computationally feasible, however, directly maximizing (24) is the cleaner choice.

E Static rental pricing

A practically important special case of the rental model is the stationary environment in which $D_t = D$ for every $t \in [T]$ and the platform prefers to post a single menu throughout the horizon. In that case the relevant benchmark is the best *static* menu

$$\text{OPT}^{\text{stat}} := \max_p \text{Rev}_D(p).$$

1315 Under stationarity, a natural ex-ante relaxation is

$$\begin{aligned}
U_{\text{stat}}^\eta &:= \max_p \sum_{n,\lambda} w_{n,\lambda} p(n, \lambda) S_{n,\lambda}(p(n, \lambda)) \\
\text{s.t. } &\sum_{n,\lambda} w_{n,\lambda} n \lambda S_{n,\lambda}(p(n, \lambda)) \leq (1 - \eta)B,
\end{aligned} \tag{25}$$

1316 where $S_{n,\lambda}(p) = \Pr[V \geq p \mid N = n, \Lambda = \lambda]$. Compared with the time-indexed benchmark, (25)
1317 uses only one decision variable for each bundle-duration pair, so it has $O(\bar{N}L_{\max})$ variables and is
1318 computationally simpler than the full slot-coupled program.

1319 Let $p^{s,\eta}$ denote an optimal solution to (25). By the same arguments as in Theorem 7 and Theorem 8,
1320 the benchmark-guided policies extend directly to this stationary static-setting benchmark.

1321 **Corollary 1** (Static rental price menu). *Let $p^{s,0}$ be an optimal solution to (25) with $\eta = 0$. If TCMPP*
1322 *uses the static menu $p^{s,0}$ in place of the time-dependent CP menu, then*

$$\mathbb{E}[\text{Rev}_{\text{TCMPP}}] \geq \frac{1 - \epsilon}{4} \text{OPT}^{\text{stat}}.$$

1323 *More generally, for $\eta > 0$, let $\pi^{\eta,s}$ be the policy that posts $p^{s,\eta}(\cdot, \cdot)$ whenever the requested type is*
1324 *feasible and posts $+\infty$ otherwise. Then*

$$\text{Rev}_D(\pi^{\eta,s}) \geq (1 - \delta_\eta)(1 - \eta) \text{OPT}^{\text{stat}}.$$

1325 NeurIPS Paper Checklist

1326 1. Claims

1327 Question: Do the main claims made in the abstract and introduction accurately reflect the
1328 paper’s contributions and scope?

1329 Answer: [Yes].

1330 Justification: The abstract and introduction state the paper’s main contributions: hardness
1331 of optimal anonymous posted-price menus, limitations of linear pricing, ex-ante convex-
1332 programming benchmarks, virtual-value reductions, and duration-aware posted-pricing
1333 guarantees. These claims are matched by the formal theorem statements and the summary
1334 of results in the main text and Appendix A.

1335 2. Limitations

1336 Question: Does the paper discuss the limitations of the work performed by the authors?

1337 Answer: [No].

1338 Justification: The paper discusses computational intractability of exact optimal pricing, the
1339 exponential cost of discretized menus, the curse of dimensionality of the exact dynamic
1340 program, and the regimes in which the proposed guarantees are strongest. The conclusion
1341 also identifies future directions, including learning demand distributions and robustness to
1342 distributional misspecification. But this paper does not include a separate limitation section.
1343 So we answer this question as [No].

1344 3. Theory assumptions and proofs

1345 Question: For each theoretical result, does the paper provide the full set of assumptions and
1346 a complete (and correct) proof?

1347 Answer: [Yes].

1348 Justification: The main theoretical results state their assumptions, including regularity,
1349 small-job conditions, finite-capacity constraints, and duration assumptions. Full proofs
1350 for the hardness results, approximation guarantees, virtual-knapsack reduction, dynamic-
1351 programming benchmark, TCMPP guarantee, and slackened-CP guarantee are provided in
1352 Appendix C.

1353 4. Experimental result reproducibility

1354 Question: Does the paper fully disclose all the information needed to reproduce the main ex-
1355 perimental results of the paper to the extent that it affects the main claims and/or conclusions
1356 of the paper (regardless of whether the code and data are provided or not)?

1357 Answer: [Yes].

1358 Justification: The simulations use synthetic instances, and Appendix B specifies the dis-
1359 tributions, budgets, horizons, request types, training-path counts, evaluation-path counts,
1360 and tuning procedures used to reproduce the reported results. The paper also describes the
1361 baselines and how the CP, slackened-CP, TCMPP, and linear-pricing policies are computed.

1362 5. Open access to data and code

1363 Question: Does the paper provide open access to the data and code, with sufficient instruc-
1364 tions to faithfully reproduce the main experimental results, as described in supplemental
1365 material?

1366 Answer: [No].

1367 Justification: The current draft provides full synthetic data-generating processes and im-
1368 plementation details, but it does not yet provide an anonymized code package. Since the
1369 experiments are based on synthetic instances rather than proprietary datasets, an anonymized
1370 reproduction script can be added to the supplemental material if desired.

1371 6. Experimental setting/details

1372 Question: Does the paper specify all the training and test details (e.g., data splits, hyperpa-
1373 rameters, how they were chosen, type of optimizer) necessary to understand the results?

1374 Answer: [Yes].

1375 Justification: Appendix B describes the selling and lending simulation settings, including
1376 the value distributions, request-type distributions, budgets, horizons, training and evaluation
1377 path counts, and grid-search procedures for tuning the linear slope, slack parameter, and
1378 TCMPP offer probability. These details are sufficient to understand the reported empirical
1379 comparisons.

1380 7. Experiment statistical significance

1381 Question: Does the paper report error bars suitably and correctly defined or other appropriate
1382 information about the statistical significance of the experiments?

1383 Answer: [No].

1384 Justification: The draft reports averages over many independent evaluation paths, but it does
1385 not currently include error bars, confidence intervals, or standard errors. This should be
1386 added if space permits; for example, the figures and tables can report standard errors over
1387 evaluation paths.

1388 8. Experiments compute resources

1389 Question: For each experiment, does the paper provide sufficient information on the com-
1390 puter resources (type of compute workers, memory, time of execution) needed to reproduce
1391 the experiments?

1392 Answer: [No].

1393 Justification: The draft describes the simulation design and number of training and evaluation
1394 paths, but it does not yet report the CPU/GPU type, memory usage, wall-clock runtime, or
1395 total compute budget. Since the experiments appear to be lightweight synthetic simulations,
1396 this information can be added in Appendix B.

1397 9. Code of ethics

1398 Question: Does the research conducted in the paper conform, in every respect, with the
1399 NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

1400 Answer: [Yes].

1401 Justification: The paper studies theoretical and simulation-based pricing algorithms for finite-
1402 capacity resource allocation and does not involve human subjects, private data, scraped
1403 datasets, model training on sensitive data, or deployment of high-risk systems. To the best
1404 of our knowledge, the research conforms to the NeurIPS Code of Ethics.

1405 10. Broader impacts

1406 Question: Does the paper discuss both potential positive societal impacts and negative
1407 societal impacts of the work performed?

1408 Answer: [No].

1409 Justification: The current draft motivates the work through GPU-market efficiency and
1410 deployable pricing, but it does not yet include a dedicated broader-impact discussion. A
1411 short paragraph should be added noting that better pricing can improve utilization and access
1412 to scarce compute, while poorly designed pricing may disadvantage budget-constrained
1413 users or amplify inequities in access to AI infrastructure.

1414 11. Safeguards

1415 Question: Does the paper describe safeguards that have been put in place for responsible
1416 release of data or models that have a high risk for misuse (e.g., pre-trained language models,
1417 image generators, or scraped datasets)?

1418 Answer: [N/A].

1419 Justification: The paper does not release pretrained models, image generators, scraped
1420 datasets, or other assets with high misuse risk. The experiments use synthetic simulation
1421 instances for pricing and resource-allocation algorithms.

1422 12. Licenses for existing assets

1423 Question: Are the creators or original owners of assets (e.g., code, data, models), used in
1424 the paper, properly credited and are the license and terms of use explicitly mentioned and
1425 properly respected?

1426 Answer: [N/A].

1427 Justification: The paper does not rely on external datasets, pretrained models, or third-party

1428 code assets as part of the core contribution or experiments. Existing scholarly works and

1429 algorithmic tools used for comparison or theoretical foundations are cited in the references.

1430 **13. New assets**

1431 Question: Are new assets introduced in the paper well documented and is the documentation

1432 provided alongside the assets?

1433 Answer: [N/A].

1434 Justification: The paper does not introduce a new dataset, benchmark, pretrained model, or

1435 software asset as a primary contribution. If anonymized simulation code is later released,

1436 documentation and license information should be included with the supplemental material.

1437 **14. Crowdsourcing and research with human subjects**

1438 Question: For crowdsourcing experiments and research with human subjects, does the paper

1439 include the full text of instructions given to participants and screenshots, if applicable, as

1440 well as details about compensation (if any)?

1441 Answer: [N/A].

1442 Justification: The paper does not involve crowdsourcing, human-subject experiments, user

1443 studies, surveys, or human data collection.

1444 **15. Institutional review board (IRB) approvals or equivalent for research with human**

1445 **subjects**

1446 Question: Does the paper describe potential risks incurred by study participants, whether

1447 such risks were disclosed to the subjects, and whether Institutional Review Board (IRB)

1448 approvals (or an equivalent approval/review based on the requirements of your country or

1449 institution) were obtained?

1450 Answer: [N/A].

1451 Justification: The paper does not involve human subjects, crowdsourcing, surveys, or user

1452 data. Therefore IRB approval or equivalent human-subjects review is not applicable.

1453 **16. Declaration of LLM usage**

1454 Question: Does the paper describe the usage of LLMs if it is an important, original, or

1455 non-standard component of the core methods in this research? Note that if the LLM is used

1456 only for writing, editing, or formatting purposes and does *not* impact the core methodology,

1457 scientific rigor, or originality of the research, declaration is not required.

1458 Answer: [N/A].

1459 Justification: The core methodology does not use LLMs as an algorithmic component,

1460 experimental subject, data source, or evaluation tool. Any use of LLMs for writing, editing,

1461 or formatting does not affect the scientific claims, theoretical results, or experimental

1462 conclusions.