Coin: Chance-Constrained Imitation Learning for Safe and Adaptive Resource Oversubscription under Uncertainty

Lu Wang
wlu@microsoft.com
Microsoft, Beijing, China

Mayukh Das
mayukhdas@microsoft.com
Microsoft, Bangalore, India

Fangkai Yang
Microsoft, Beijing, China

Chao Du
Microsoft, Beijing, China

Bo Qiao
Microsoft, Beijing, China

Hang Dong
Microsoft, Beijing, China

Chetan Bansal
Microsoft, Redmond, USA

Si Qin
Microsoft, Beijing, China

Saravan Rajmohan
Microsoft, Redmond, USA

Qingwei Lin
Microsoft, Beijing, China

Dongmei Zhang
Microsoft, Beijing, China

Qi Zhang
Microsoft, Beijing, China

ABSTRACT

We address the real problem of safe, robust, adaptive resource oversubscription in uncertain environments with our proposed novel technique of chance-constrained imitation learning. Our objective is to enhance resource efficiency while ensuring safety against congestion risk. Traditional supervised or forecasting models are ineffective in learning adaptive oversubscription policies, and conventional online optimization or reinforcement learning is difficult to deploy on real systems. Offline policy learning methods, such as Imitation Learning (IL) can leverage historical resource utilization telemetry data to learn effective policies if we can ensure robustness and safety from the underlying uncertainty in the domain, and thus the data. Our work investigates the nature of this uncertainty, how it can be quantified and proposes a novel chance-constrained IL that implicitly models such uncertainty in a principled manner via additional knowledge in the form of stochastic constraints on the associated risk, to learn provably safe and robust policies. We show empirically a substantial improvement (∼ 3-4×) in capacity efficiency and congestion safety in test as well as real deployments.

CCS CONCEPTS

• **Computing methodologies** → **Knowledge representation and reasoning**; **Apprenticeship learning**; • **Applied computing** → **Enterprise computing**.

KEYWORDS

Oversubscription, Uncertainty, Knowledge, Imitation Learning, Reinforcement Learning, Robustness, Safety, Chance-Constraints

ACM Reference Format:
Lu Wang, Mayukh Das, Fangkai Yang, Chao Du, Bo Qiao, Hang Dong, Chetan Bansal, Si Qin, Saravan Rajmohan, Qingwei Lin, Dongmei Zhang, and Qi Zhang. 2024. Coin: Chance-Constrained Imitation Learning for Safe and Adaptive Resource Oversubscription under Uncertainty. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24), October 21–25, 2024, Boise, ID, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3627673.3680060

1 INTRODUCTION

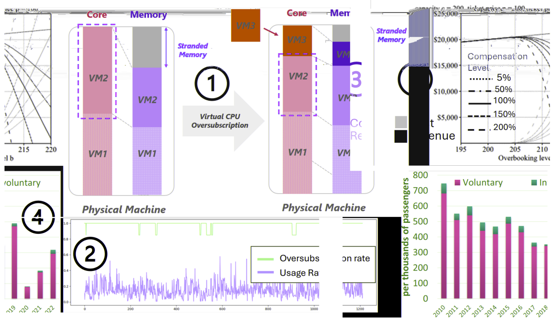


Figure 1: Resource oversubscription in real scenarios: 1. vCPU oversub in cloud saves stranded core/memory; 2. Naive oversub not aligned with usage; 3. cost vs revenue for involuntary bumping in overbooked flights [34]; 4. Voluntary vs Involuntary bumping frequency 2010-22 for overbooking

Resource *"Oversubscription"* is a prevalent strategy across the professional services industry, including cloud services [8, 23, 43], airlines [45], logistics [51], etc., to optimize their operating costs (capacity efficiency) via 'thin-provisioning. The key idea is that a system offers more resources to users or entities than its available capacity, assuming that not all users would simultaneously utilize their allocated capacity, in order to diminish the sum of unutilized resources and increase gains.

For instance, in cloud services, Virtual Machines (VMs) with some quantum of virtual resources such as virtual cores/memory etc. are hosted on Physical Machines (PMs). Virtual CPUs (vCPU) represent the share of the underlying physical CPU assigned to VMs and are the sellable billing units in cloud platforms. Oversubscription assigns fewer resources to a VM than what it requests, assuming that a VM will be allowed to use extra resources beyond its allocated limit if needed. This allows the platform to leverage unused physical capacity by packing more VMs in PMs (Fig. 1①) as an effective way to avoid unnecessary resource waste (stranded cores/memory) and maximize profits [3, 4]. CPU bottlenecks are more severe and prevalent [30], so in this work we focus on vCPU oversubscription for cloud domain. In the airline industry, overbooking (oversubscribing) fight tickets beyond aircraft capacity improves airline load factors and reduce revenue losses due to cancellations and no-shows [33, 45], while also adding risk of cost of compensations (Fig 1). Note that, oversubscription, resource allocation, scheduling, and optimal packing are interconnected concepts in resource management, each addressing distinct aspects. Oversubscription arises when demand surpasses supply, while resource allocation and scheduling optimize resource placement. Optimal packing minimizes fragmentation.

Oversubscription policy (margin/rate i.e. how much to overbook for an entity) is a crucial (often overlooked) scientific problem. Conservative oversubscription policies lead to resource wastage, whereas aggressive policies cause resource congestion, compromising reliability. *Therefore, the policies should be* ***adaptive*** *to the temporal dynamics of resources usage, but safe and robust from the underlying* ***uncertainty*** *of workload / utilization patterns in real platforms.* To that end, we need to address the following challenges - $\boxed{1}$ Appropriate data/information to learn adaptive policies $\boxed{2}$ Identify the nature and impact of the uncertainty $\boxed{3}$ Learn policies that are safe and robust to the risks due to this uncertainty but leads to optimum benefit as well.

Most existing research have limited outlook. Some cloud oversubscription work focus on resource allocation problem via online bin-packing [20] or reactive user migration for overload mitigation [27, 50]. Others leverage online Reinforcement learning (RL) [14], often with constraints [31, 36] to design adaptive solutions, but constrained multi-objective RL may not converge and are hard to train/deploy for real platforms.

We employ Imitation Learning (IL) instead, an offline policy learning strategy that can leverage historical resource usage telemetry data. Fig 1 illustrates the dynamic utilization patterns in some of Microsoft's internal services. We retrieve this telemetry data but their are two important considerations, $\boxed{1}$ There is no true label for the most accurate value of oversubscription ratio, we use usage a proxy label (fig. 1), $\boxed{2}$ Significant aleatoric uncertainty which may cause divergence or high unexplained variance making policies unsafe wrt. resource congestion risk. While Uncertainty Quantification (UQ) methods, [1], can explicitly represent uncertainty, they are not tractable in real-time and cannot capture the real-world interactions [11]. Under this uncertain, dynamic demand/usage data, to learn safe yet highly beneficial policies, we propose Coin, a Chance-cOnstrained Imitation learNing framework that exploits chance-constraints (stochastic) to implicitly model the underlying uncertainty. It learns adaptive policies that optimize resource efficiency and safety /robustness against congestion risk. Hard constraints, while strictly enforce safety, may lead to higher resource wastage. We make the following major contributions – **(1).** We characterize uncertainty information in oversubscription and design chance constraints on risk from domain knowledge. **(2).** We propose novel chance-constrained imitation learning to solve safe but beneficial oversubscription policy under uncertainty. **(3).** We demonstrate, empirically, on test+real scenarios (Cloud - 1st party/internal, 3rd party/public Azure, Airline), how Coin can learn safe yet effective policies. In the following sections, we outline related work (Sec. 2), define the optimal oversubscription problem, the telemetry/data and the nature of the uncertainty(Sec. 3.1), followed by the formalism of chance constraints on IL, algorithm and tractable implementation (Sec. 3.2 to 3.4) In sections 4.2, 4.3 & 4.4 we show extensive evaluations on testbeds and real deployment.

2 RELATED WORK

Oversubscription: Cloud oversubscription is practiced for different resource types, *e.g.*, CPU [8], memory [6], power [25], etc. However, our evaluations focus on virtual CPU/core oversubscription as CPU is one of the most vital, but scarce, resources in the cloud [19]. Some industrial solutions [18] globally optimize resource used, under constrained while others [16] emphasize adaptive solutions for actual usage patterns. [5, 7, 25] employ ML/DL for VM usage patterns, while others [26] adopt RL for airline inventory control.

Chance-Constrained RL: Chance constrained RL has gained popularity in recent years. Some model-based chance-constrained RL methods improve the over-conservative policy with efficient evaluations [37, 38]. However model-based methods are challenging to deploy on real systems. Model-free chance-constrained RL methods primarily adopt penalty and Lagrangian techniques [15, 35]. However, the oversubscription problem involves a large number of chance constraints, making existing methods difficult to adapt.

Safe Imitation Learning: Most traditional imitation learning, inverse RL or even Constrained IL approaches [12, 24] struggle to induce safe policies in presence of uncertainty. Some safe IL approaches from uncertain or inconsistent demonstrations focus on explicitly predicting/estimating uncertainty via kernels or model-based formulation [13, 46] by computing "value at risk". Some use Monte Carlo policy rollouts or via ensembles (with dropouts) [9, 32]. While related in spirit, our stochastic constraints implicitly ensure safety against uncertainty instead of explicit uncertainty quantification; hence no extra sampling.

3 METHOD

3.1 Problem Setting

3.1.1 Optimal Oversubscription. In Oversubscription, the system provisions a $\zeta \sim [0, 1]$ fraction of the amount of resources requested by an entity in order to maximize the recovery of unused resources (COGS [21]savings/benefit), since more resources can be served from the same resource pool, [COGS benefit = $\max_\zeta \sum\sum(1-\zeta) \times$ Cost(resource ask)$_{/entity/pool}$]. For effective oversubscription we assume not all the entities will have peak resource usage at the same time. However, there is risk of overloading or resource congestion if there is simultaneous peak demand. Optimal oversubscription must optimize both COGS benefit and congestion risk together (Eqn 1).

$$\zeta^* = \arg\max_{\zeta} \underbrace{\sum\sum(1-\zeta) \times \text{Cost(resource ask)}_{/entity/pool}}_{\text{COGS savings/benefit}}$$
$$+ \arg\min_{\zeta} \underbrace{\sum (\text{Cost of congestion})_{\zeta}/\text{pool}}_{\text{Risk}} \quad (1)$$

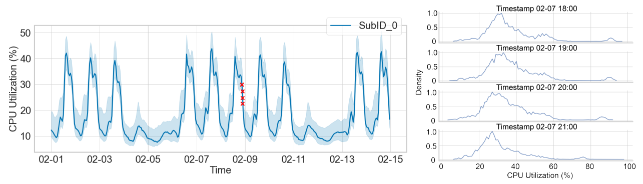


Figure 2: CPU utilization of a sampled user (left). The variance band represents 25th and 75th percentiles. Four points (in red) are sampled to show their probability densities.

There is no closed form solution to the problem. Also, ζ^* is not one value but different values across time based on dynamic usage patterns and conditions. Thus we learn a policy $\pi^* = P(\zeta^*|usage, time, \ldots)$ to maximize the COGS saving as well as minimize Risk. In oversubscription scenarios, where return/reward is a multi-objective COGS savings vs risk, it is hard for online RL to stably and sample-efficiently to learn a policy. In professional services industry we can often access and exploit historical usage telemetry (trajectories) which makes a strong case for offline methods such as Imitation Learning.

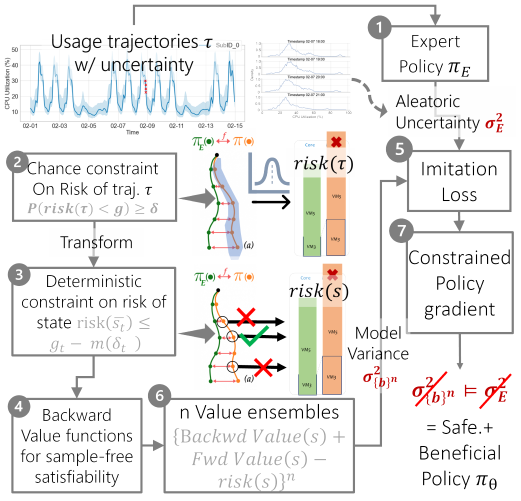
3.1.2 Data and uncertainty. We leverage historical telemetry data $\mathcal{D}$ which comprises the resource usage measurement trajectories across time characterized by the relevant features about the system load and demand. There are no true labels about ideal oversubscription margin/rate. Instead, resource usage rate $u = (used)/(requested)$, acts as the proxy label such that $\zeta_t \propto u_t$.

The aleatoric uncertainty (AU) [22] in this data comes from the domain itself, due to $\boxed{1}$ uncertainty of demand characteristics, $\boxed{2}$ statistical variance due to measurement granularity, $\boxed{3}$ noise during retrieval and aggregation or $\boxed{4}$ may be due to reactive mitigation. Figure 2 illustrates an example from cloud services domain, where the CPU usage rate for different subscribers/users at different time steps are not point estimates but are stochastic processes, approximately, characterized by Gaussian distributions (possibly skewed/mixture). AU makes the computation of the risk in Eqn. 1 arbitrarily erroneous which can lead to catastrophic congestion.

3.1.3 Imitation Learning from uncertain Trajectories. In traditional imitation learning the goal is to minimize the divergence between agent policy $\pi_\theta = P(\hat{a}_t = \zeta_t|s_t)$(predicted actions) and the expert policy $\pi^E = P(a_t = u_t|s_t)$ (underlying policy induced by data) $[L = \pi^E(a_t|s_t) \ominus \pi_\theta(\hat{a}_t|s_t)]$, where $s_t \in S, a_t \in A$ are state and action spaces. Naïvely this is meant to maximize the benefit, $(1 - (\zeta \simeq u)) \times$ requested, but in presence of uncertainty, it becomes a harder problem. Considering AU in the data as a variance in the expert policy, $\sigma_E^2 = \mathbb{E}[((a \sim \pi^E) - \mathbb{E}_{a\sim\pi^E}[a|s])^2]$, the bias-variance decomposition of the generic loss, gives,

$$\begin{aligned} L &= \pi^E(a_t|s_t) \ominus \pi_\theta(\hat{a}_t|s_t) \\ &= \underbrace{(\mathbb{E}_{\hat{a}\sim\pi_\theta}[\hat{a}|s] - \mathbb{E}_{a\sim\pi^E}[a|s])^2}_{\text{bias}} + \underbrace{\sigma_\theta^2}_{\text{model var}} + \underbrace{(\sigma^E)^2}_{\text{irreducible}} \end{aligned} \tag{2}$$

AU manifests as irreducible variance component in the error [1]. This is unsafe because, **(1)** in out-of-distribution conditions we take $\zeta \simeq \mathbb{E}[u]$; leads to arbitrary resource congestion [12], **(2)**



Usage trajectories τ w/ uncertainty
1
Expert Policy π_E
Aleatoric Uncertainty σ_E^2
2
Chance constraint On Risk of traj. τ
$P(risk(\tau) < g) \geq \delta$
Transform
3
Deterministic constraint on risk of state $risk(\bar{s}_t) \leq g_t - m(\delta_t)$
4
Backward Value functions for sample-free satisfiability
5
Imitation Loss
6
n Value ensembles $\{Backwd\ Value(s) + Fwd\ Value(s) - risk(s)\}^n$
7
Constrained Policy gradient
$\pi_E(\bullet) \xleftrightarrow{f} \pi(\bullet)$
$risk(\tau)$
$risk(s)$
Core
VM5
VM3
(a)
Model Variance $\sigma^2_{\{b\}^n}$
$\cancel{\sigma^2_{\{b\}^n}} \vDash \cancel{\sigma^2_E}$
= Safe.+ Beneficial Policy π_θ

Figure 3: Solution Overview of Coin framework

for in-distribution cases, either we cannot measure risk (Eqn. 1) confidently or we lose benefit if $\zeta > u$ strictly using hard constraints. We enforce safety using constraints with 'probably approximate' satisfiability, (chance) on congestion risk (not action space unlike some constrained RL approaches with UQ [29]) and present efficient solution without trajectory sampling.

3.2 Problem and Solution Overview

We propose the Chance-cOnstrained Imitation learNing (Coin), where the training trajectory τ consisting of states and actions $(s_0, a_0, s_1, a_1, ..., s_T, a_T)$ are drawn from the expert policy π_E from $\mathcal{D}$ ①. Figure 3 illustrates the solution schematic. The oversubscription MDP is the standard $\langle S, A, \mathcal{R}, \mathcal{P}, \gamma \rangle$ where factored state space S comprises user's resource requests, historical usage, current resource status, etc. as features, and action space $A \in [0, 1]$ could be usage rate u or oversubscription rate ζ. We try to learn a policy $\pi_\theta(\hat{a}|s)$, where $\hat{a}$ is the predicted ζ. IL does not use explicit reward $\mathcal{R}$, but maximizes benefit via minimizing loss and mitigates risk through constraints. We find a policy that mimics the expert policy while probabilistically satisfying the chance constraints. We formulate our problem as:

$$\underbrace{\min_\pi L(\pi^E, \pi_\theta)}_{\text{Benefit Maximization}} \;;\; \text{s.t.} \underbrace{\Pr\left(\frac{1}{T}\sum_{t=0}^{T} c(s_t) \leq g\right) \geq 1 - \delta}_{\text{chance-constraint on risk ②}} \tag{3}$$
$$s_0 \sim \mathcal{P}_0, s_{t+1} \sim \mathcal{P}(s_t, a_t);\ a_t \sim \pi_\theta(\cdot|s_t);$$

where $L(\pi^E, \pi_\theta)$ is the objective function of mimicking the expert's policy (mean square error function for Behavior Cloning [39] or a JS divergence for adversarial IL). We adopt behavior cloning as the base learner for simplicity of implementation on real-world data. $c(s_t) = h^\top s_t$ is the instantaneous constraint cost, defined as a function of the risk in the task based on domain knowledge. $\mathcal{P}_0$ is the distribution of s_0, $\mathcal{P}(s'|s, a)$ is the state transition probability. Chance constraints should cumulatively satisfy with probability $1 - \delta$, where δ is the global upper bound of the probability of resource congestion. g is the upper bound on the cumulative cost, i.e., resource limit for the whole oversubscription action.

[1]For KL div. loss $(\sigma^E)^2$ is entropy - information theoretic measure of uncertainty

There is no theoretically viable way to check probabilistic satisfiability of chance constraints without brute-force sampling. Hence, we consider a linear dynamic system $\mathcal{P}(s'|s,a)$ under Gaussian distribution for converting the chance constraints into deterministic form. Such Gaussian assumption is valid and follows directly from Section 3.1. T is the time horizon.

Challenges: The constraint is stochastic, cumulative and depends on the distribution of entire trajectories. This optimization problem is harder since – **(I.)** Verifying satisfiability requires computing integral of a multi-variate probability distribution and the solution may be intractable or non-existent unless $c(s_t) \leq g_t$ is mutually exclusive across all states. **(II.)** Measuring the probabilities require sampling of trajectories at every iteration (infeasible for real problems).

Solution Strategies: We address these challenge via three mechanisms. *(1) Stochastic → Deterministic [Challenge I]*: Due to the difficulty of computing the integral, we transform the chance constraint in Equation 3 into a deterministic constraint defined on the states $\bar{s}_t$, where $\bar{s}_t = E[s_t]$ is the expected value of state (represents the major value of the states). *(2) Backward Value [Challenge II]*: Given the deterministic constraint, inspired by the previous work, we estimate the constraint value for each time step with a temporal difference (TD) [48] methods instead of sampling entire trajectories. This method updates the estimated value of chance constraint via bootstrapping. *(3) Value ensembles [Challenge II]*: We further design a practical tractable implementation of our proposed approach for real-world problems via value function ensembles. We annotate each section with module numbers in Fig. 3.

3.3 Satisfying the Chance Constraint

Unlike traditional constrained optimization, typical branch and bound strategies will not suffice in optimizing an IL objective. So we describe how to transform a chance constraint into a deterministic constraint to assert (stochastic) constraint satisfaction and construct the backward value function in the context of IL with the deterministic form.

3.3.1 Transforming to deterministic constraint. 3 With the assumption that the state transaction distribution is Gaussian distribution, i.e., $h_t^\top s_t \sim \mathcal{N}(h_t^\top \bar{s}_t, \sigma_t^2)$, we derive the deterministic version of chance constraint as follows:

$$c(\bar{s}_t) \leq g_t - m(\delta_t) \tag{4}$$

where $\bar{s}_t = E[s_t]$ is the nominal state of the agent, $c(\bar{s}_t) = h_t^\top \bar{s}_t$, $m(\cdot)$ is the inverse of the cumulative distribution function of univariate Gaussian distribution, g_t and δ_t are local cost and chance constraint probability per time step t.

$$m(\delta_t) = -\sqrt{\sigma_t^2}\Phi^{-1}(\delta_t), \tag{5}$$

where $\sigma_t = \sqrt{2h_t^\top \sum_{s_t} h_t}$ is the standard deviation and $\Phi^{-1}(\delta_t) = \mathtt{erf}^{-1}(\delta_t)$ is the inverse of the Gaussian fn.

Lemma 1 (Deterministic ≈ Chance constraint). *A feasible solution to the deterministic constraint in Equation 4 is always a feasible solution to the original chance constraint in Equation 3. (Proof excluded due to space constraints but can be easily derived from Eqn. 5)*

3.3.2 Backward value function for satisfiability estimation. 4 In (chance - constrained) IL, the policy needs to consider the cumulative constraints from time step 0 to current time step t to plan for the future. Now, given the derived deterministic constraint function in Equation 4, we observe that the constraint is estimated by sampling the whole trajectory, which is not sample efficient. To improve the sample efficiency, TD learning estimates the current time step's cumulative value via bootstrapping the previous time step's cumulative value, i.e., $V^\pi(s_t) \leftarrow r(s_t,a_t) + \gamma \sum_{s_{t=1}} V^\pi(s_{t+1})$, where $V^\pi(s)$ is the state value function [48]. This cumulative constraint value is named as backward value function [41], and is extremely sampling efficient. Inspired by the previous work, we design a new backward value function for the chance constraints. With this, we obtain an analytical solution step by step.

Lemma 2 (Backward value ∼ Forward Markov chain). *With the irreducible and aperiodic assumption in forward Markov chain $\mathcal{P}^\pi(s_{t+1}|s_t) = \sum_{a_t \in \mathcal{A}} \mathcal{P}(s_{t+1}|s_t)\pi_\theta(a_t|s_t)$, samples from the forward Markov chain $\mathcal{P}^\pi(s_{t+1}|s_t)$ can be used directly to estimate the statistics of the backward Markov chain $\overleftarrow{\mathcal{P}}^\pi(s_t|s_{t+1}) = \sum_{a_t \in \mathcal{A}} \overleftarrow{\mathcal{P}}(s_t, a_t|s_{t+1})$ [41]. For any $K \in \mathbf{N}$, where $\mathbf{N}$ is the set of natural numbers. Thus we can obtain the backward value function:*

$$\overleftarrow{V}^\pi(s_t) = \mathbf{E}_{\overleftarrow{\mathcal{P}}^\pi}\left[\sum_{k=0}^{K} c(s_{t-k})|s_t\right] \tag{6}$$

$$= \mathbf{E}_{\mathcal{P}^\pi, s_{t-K}\sim\eta^\pi(\cdot)}\left[\sum_{k=0}^{K} c(s_{t-k})|s_t\right] \tag{7}$$

where $\mathbf{E}_{\overleftarrow{\mathcal{P}}}[\cdot]$ indicates the expectation over backwards chain, η^π is the stationary distribution satisfying $\eta^\pi(s_{t+1}) = \sum_{s\in\mathcal{S}} \mathcal{P}^\pi(s_{t+1}|s_t)\eta^\pi(s_t)$.

Based on Lemma 2, we can get the backward value from the forward Markov chain instead of the backward Markov chain, which further simplifies the backward value estimation. We can then use the traditional TD learning setting to estimate the backward value function recursively as follows:

$$\overleftarrow{V}^\pi(s_t) = \mathbf{E}_{\overleftarrow{\mathcal{P}}^\pi}[c(s_t) + \overleftarrow{V}^\pi(s_{t-1})] \tag{8}$$

With the defined backward value function above, we can rewrite the derived deterministic constraint at each time step. Recall the derived deterministic constraint:

$$\sum_{t=0}^{T} c(\bar{s}_t) \leq g - m(\delta) \rightarrow \mathbf{E}_{\mathcal{P}^\pi}[\sum_{t=0}^{T} c(s_t)] \leq g - m(\delta) \tag{9}$$

Alternatively, for each time step $t \in [0,T]$ of the trajectory:

$$\underbrace{\mathbf{E}[\sum_{k=0}^{t} c(s_k)|s_0,\pi]}_{\text{backward from t}} + \underbrace{\mathbf{E}[\sum_{k=t}^{T} c(s_k)]}_{\text{forward from t}} - \mathbf{E}[c(s_t)] \leq g - m(\delta)$$

The trajectory-level constraints is not transformed into state-level,

$$\mathbf{E}_{s_t\sim\eta^\pi(\cdot)}[\overleftarrow{V}^\pi(s_t) + V^\pi(s_t) - c(s_t)] \leq g - m(\delta). \tag{10}$$

As proved by the previous work, the state-level constraint in Equation 10 with backward value function is an upper bound to the original constraint in Equation 9. Thus the policy satisfying the state-level constraint will also ensure the original constraint.

Given the formulation of the state-level constraint, we can obtain a policy improvement at each time step, where a safe feasible policy can be optimized by the estimated constraint value. The final objective, modifying Eqn 3 with the state-level chance constraints on IL, is as follows 5:

$$\begin{aligned}&\min_{\pi} \ L(\pi^E, \pi_\theta)\\ &\text{s.t. } \mathbf{E}_{s_t\sim\eta^\pi}[\overleftarrow{V}^\pi(s_t) + V^\pi(s_t) - c(s_t)] \le g_t - m(\delta_t)\end{aligned} \tag{11}$$

3.4 Practical Implementation Design

There are still two main challenges in optimizing the deterministic objective function in Eqn 11. **(A)** directly solving this constraint problem via Lagrangian-based methods or Lyapunov-based methods will introduce large extra computational complexity and **(B)** when we try to check whether the policy is feasible in each time step, we need to compare the constraint value with the threshold which requires calculating the standard derivation which is hard to estimate in large-scale datasets.

3.4.1 Chance constrained optimization via policy gradient. 7 For challenge **(A)**, we leverage the safety layers based method [10], a constraint projection approach, to conduct action correction at each time step. For each state, the safety layer projects the unconstrained action to the nearest action (in Euclidean norm) satisfying the necessary constraints. Following the derived analytical solution of previous work [41], let $d(s) = \nabla Q^\pi(s,a)|_{a=\pi_\theta(s)}$, where $Q^\pi(s,a)$ is the state-action value:

$$a* = \pi_\theta(s) - \lambda^* \cdot d(s), \tag{12}$$

where $\lambda^* = (\frac{-(g+c(s)-\overleftarrow{V}^\pi(s)-Q(s,\pi_\theta(s)))}{d(s)^T d(s)})^+$ is the analytical soln.

Based on the above, we can consider the constraint term as an additional loss, then conduct gradient descent to optimize the policy **if** the policy is not in feasible space, i.e., $\mathbf{E}_{s_t\sim\eta^\pi(\cdot)}[\overleftarrow{V}^\pi(s_t) + V^\pi(s_t) - c(s_t)] > g - m(\delta)$.

$$\theta \leftarrow \theta - \nabla_\theta \log \pi_\theta(s) \cdot Q^\pi(s,a). \tag{13}$$

Otherwise, we update the θ via the loss of traditional imitation learning as $\theta \leftarrow \theta - \nabla_\theta L(\pi^E, \pi_\theta)$

3.4.2 Ensemble Learning to estimate the variance of cost value. 6 To solve the second challenge, where we need to verify whether the policy is feasible in the current time step. We learn N backward value function and N forward value functions where they will sync with other during several steps of training to estimate the standard deviation of the costs.

$$\sigma_t = \sum_{n=1}^{N} b_t^n - \mathbb{E}[b_t^n], \tag{14}$$

where $b_t = \overleftarrow{V}^\pi(s_t) + V^\pi(s_t) - c(s_t)$ is value of state-level constraint.

Lemma 3 (Stochasticity $\models$ Aleatoric uncertainty). *The irreducible variance due to aleatoric uncertainty σ^E (Eqn. 2) is entailed by the controlled model variance from the ensemble of value functions with state-level constraints. σ_t in Equation 14, $\sigma_t^2 \models (\sigma^E)^2$.*

Algorithm 1 Coin

```
Require: # epochs $M$, $\pi_\theta$, $\overleftarrow{V_\omega}^\pi(s_t)$, $Q_\phi(s_t,a_t)$, constraint threshold $g$, constraint probability
    $\delta$ from domain knowledge;
 1: Initialize actor $\pi_\theta$, $N$ target backward value function $\overleftarrow{V_\omega}^\pi(s_t)^n$, target forward value function
    $Q_\phi(s_t,a_t)^n$, constraint threshold $g$ and upper bound of resource congestion probability $\delta$;
 2: for $m = 0$ to $M$ do
 3:     Select the action $a_t$ from the the projection of action $a^*$ which is calculated by Equation 12,
        take action $a_t$ and observe the state $s_{t+1}$ and cost $c(s_t)$;
 4:     n ← n+1;
 5:     Calculate the backward value for the constraint with $N$ value functions;
 6:     $Q^n(s_t,a_t), V^n(s_t), \overleftarrow{V_\omega}^{\pi,n}(s_t)$;
 7:     Calculate the standard derivation of the value $\sigma$ and $m(\delta)$;
 8:     Calculate the deterministic constraint to check whether the current action is feasible:
 9:     if $\overleftarrow{V}^\pi(s_t) + V^\pi(s_t) - c(s_t)] \le g_t - m(\delta_t)$ then
10:         update the policy $\theta \leftarrow \theta - \nabla_\theta L(\pi^E, \pi_\theta)$
11:     else
12:         Use safeguard policy update to recover
13:         $\theta \leftarrow \theta - \nabla_\theta \log \pi_\theta(s_t) Q(s_t)$
14:     end if
15: end for
16: Update backward and forward constraint values here:
17: for $i \in \{t_{start}, ..., t\}$ do
18:     $\overleftarrow{R} \leftarrow c(s_t) + \gamma \overleftarrow{R}$
19:     $\overleftarrow{V}_\omega^{\pi,n} \leftarrow \phi - (\overleftarrow{R} - \overleftarrow{V}_\omega^{\pi,n})^2$          ▷ upd. backward const. val.
20: end for
21: for $i \in \{t-1, .., t_{start}\}$ do
22:     $R \leftarrow c(s_t) + \gamma R$
23:     $Q_\phi^n \leftarrow Q_\phi^n - (R - Q_\phi^n)^2$          ▷ upd. forward constraint value
24: end for
```

3.5 Algorithm

We integrate Coin on top of behavior cloning[2]. Details of the algorithm can be found in Algorithm 1. We discuss an alternate update for implementing the algorithms.

4 EXPERIMENTS

We evaluate Coin, based on the following questions:

Q1. Does Coin outperform baselines and learns safe but beneficial oversubscription policies from uncertain data?
Q2. Does Coin satisfy the constraints on risk to ensure safety?
Q3. Convergence stability of Coin under uncertainty?
Q4. How effective is Coin in practice in real deployments?**

Baselines: We compare Coin against the following baselines (1) Grid search with different oversub probabilities, where all users have the same oversub rate (2) Vanilla IL – Behavior Cloning/BC, (3) Online RL such as DDPG [28] (4) Multi-Agent RL or MA [43] and (5) IL with hard constraints. Most recent resource management or oversub frameworks in practice [17, 25, 26, 40, 44] leverage online RL. Thus our baseline choices are comprehensive.

Metrics: (1) **Cloud services:** The chance-safety of cloud service is evaluated via *physical Machine hot ratio* (short as **PM-Hot-R**). In addition, we set chance-safety ratio g (g $g \in \{\mathbf{0.75, 0.85, 0.95}\}$) to check whether the oversubscription policy satisfies the chance constraint with different g. PM-Hot-R indicates the maximum probability of a PM in the cluster being hot than δ. The remaining cores quantity is evaluated via *saved cores* (short as **S-Cores**). (2) **Airline ticket overbooking:** The chance-safety of Airline ticket overbooking is evaluated via *Ticket-Cost ratio* (short as **Ticket-Cost-R**). In addition, we set chance-safety ratio g ($g \in \{\mathbf{0.75, 0.85, 0.95}\}$) to check whether the oversubscription policy satisfies the chance constraint with different g. The profit is evaluated via *Profit.*

[2]Code: https://anonymous.4open.science/r/coin-B0FF/

4.1 Experiment Design

4.1.1 Evaluation Domains. We evaluate Coin on two major domains, vCPU oversubscription in cloud services and flight tickets overbooking in Airlines.

Cloud Oversubscription: In this domain we evaluate on two datasets: $\boxed{\text{i}}$ Microsoft's internal cloud platform and $\boxed{\text{ii}}$ a public dataset from Azure (AzurePublicDatasetV2[3]) [42]. *Usage/demand distributions in these two datasets differ, allowing us to evaluate our method under different conditions and treat them as two different test beds.* Internal Cloud dataset contains sampled traces in February 2022, consisting of 1.5 million virtual machines (VMs). The AzurePublicDatasetV2 contains data from the 2019 Azure VM workload, comprising information on ≈ 2.6 million VMs and 1.9 billion utilization readings. *[Although, we have deployed and tested our framework on Microsoft Internal (1st party) cloud services and observed substantial impact (discussed later), we highlight how our method solves the generalized oversubscription problem under uncertainty with flight overbooking data set, described next.]*

Airline ticket overbooking: While flight overbooking, *i.e.*, more bookings than the available seats, is a common practice, the **uncertainty** in ticket demands and no-shows result in inappropriate overbooking strategies, leading to "denied boardings" (Fig 1). In general, demands patterns are quarterly, peaking during tourist seasons [2, 47]. This motivates adaptive overbooking policies. We collect airline passengers' data from the overbooking reports of the U.S. Department of Transportation (DOT) covering 32 airline companies in the U.S. from 1998 to 2021[4], reported quarterly. Each quarter's data includes offloaded number of passengers (voluntary/involuntary) and the onboarded passengers.

4.1.2 Environment Settings. Apart from the real deployment on Internal cloud we have also created "gym" environments for evaluating our as well as baseline policies. Oversub Gym for cloud includes two environments built on two different cloud datasets (Internal cloud and Public Azure). We conduct experiments under both cold-start (empty environment where no previous VMs) and warm-start (environment with previously allocated VMs running) scenarios in the Internal Cloud dataset. The design of the environment involves the following critical aspects, namely, **(1) VM Allocation** to PMs **(2) Reset**, **(3) State**, **(4) Transition model** - state transitions, **(5) Action**, - oversubscription/usage rates and **(6)The chance constraints** with customizable g, (e.g., in cloud service, g = bound on **hot** PMs (w/ high usage)), and δ. For Airline ticket overbooking, the gym environment uses a GBDT simulator trained on the data.

4.2 Experimental Results

4.2.1 Microsoft Internal Cloud. We train our model with 1800 episodes and 10 seeds. Three different safety-levels, i.e., g are considered as 0.75, 0.85, and 0.95 respectively. Results of Coin with the chance constraint probability $1 - \delta = 0.95$ against 7 baselines, the following observations can be made: (1) Coin satisfies chance constraint with all different chance-safety g and achieves the highest saving than the other methods that also satisfy the constraints, i.e., Grid-0.6, MA, IL and IL with hard constraints. These results indicate that the chance constraint is a crucial factor in ensuring performance and safety together. (2) Coin shows significantly more saved cores (benefit) than IL+hard constraint, which indicates hard constraints are too conservative. (3) DDPG and MA shows worse performance in benefit, which indicates that RL is not effective in oversubscription under uncertainty. These results highlight how Coin is extremely effective in designing profitable oversubscription policies in internal cloud system where COGS benefit is very important, answering **Q1** affirmatively while stochastically ensuring safety against congestion (hot node) risk (**Q1**).

4.2.2 Azure Public Cloud. The average usage rate of public Azure is much higher than the internal Cloud. As shown in Table 2, we can see that Grid fails to satisfy the safety constraint due to that most of the users have large usage rates. Four methods, i.e., Coin, IL, IL+hard constraint and MA, satisfy the chance constraint with different chance-safety g. Our method Coin consistently achieves the largest benefit. The IL based models achieve better performance than RL based models, because optimizing RL objectives in the highly stochastic environment, i.e., cloud service, is hard.

The results also suggest that the chance constraints can help improve the performance of IL models. These results highlight how Coinis extremely effective in designing profitable oversubscription policies in external cloud system where COGS benefit is very important, while seamlessly satisfying the stochastic congestion constraints, thus answering **Q1** & **Q2** affirmatively.

Table 1: Results in Internal Cloud

Method	PM-Hot-R	S-Cores	0.75	0.85	0.95
Grid-0.2	95.4	80.0	✗	✗	✗
Grid-0.4	100	60.0	✗	✗	✗
Grid-0.6	0.0	40.0	✓	✓	✓
MA	0.0	38.4	✓	✓	✓
DDPG	6.7	35.6 ± 1.2	✓	✓	✓
IL (BC)	0.8 ± 1.1	49.9 ± 3.3	✓	✓	✓
IL + hard constraint	0.73 ± 1.5	37.9 ± 4.6	✓	✓	✓
Coin	1.5 ± 0.97	66.8 ± 13.5	✓	✓	✓

Table 2: Results in Azure public Cloud

Method	PM-Hot-R	S-Cores	0.75	0.85	0.95
Grid-0.2	100.0	80.0	✗	✗	✗
Grid-0.4	100.0	60.0	✗	✗	✗
Grid-0.6	100.0	40.0	✗	✗	✗
MA	0.0	5.7	✓	✓	✓
DDPG	3.89	8.4	✓	✓	✓
IL (BC)	1.0 ± 0.0	7.6 ± 2.2	✓	✓	✓
IL+hard constraint	0.0 ± 0.0	5.4 ± 2.1	✓	✓	✓
Coin	1.67 ± 3.2	15.78 ± 0.13	✓	✓	✓

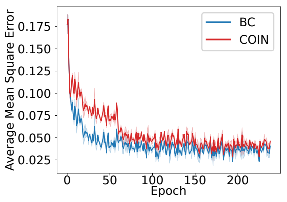
4.2.3 Airline Tickets Overbooking. As the overbooking rate of each airline company and the actual demands are not reported, we generate overbooking rate and flight demands based on known common industry practice [49] to create a semi-synthetic dataset. With the popularity of e-tickets, no-shows are less compared to the paper-ticket period [33]. So we assume the no-show rate to decay with years. Then the demand is the aggregation of bumped, no-shows, and onboard passengers. Our GBDT-based simulator is trained on the above semi-synthetic data. The results are shown in Table 3. We observe that, (1) the Grid methods fail, where none of them

[3]https://github.com/Azure/AzurePublicDataset
[4]https://www.bts.gov/denied-confirmed-space

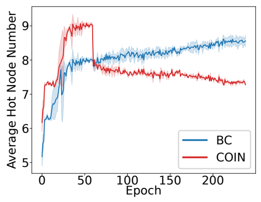
Table 3: Oversubscription results in Airline

Method	Ticket-Cost-R	Profit	0.75	0.85	0.95
Grid-0.2	100.0	16.24	✗	✗	✗
Grid-0.4	100.0	12.18	✗	✗	✗
Grid-0.6	100.0	8.12	✗	✗	✗
MA	1.7%	7.2	✓	✓	✓
DDPG	1.28%	6.9	✓	✓	✓
IL (BC)	0.9% ± 0.0	7.8 ± 2.2	✓	✓	✓
IL+hard constraint	0.0 ± 0.0	5.4 ± 2.1	✓	✓	✓
Coin	0.3% ± 3.2	9.78 ± 0.13	✓	✓	✓

satisfy the chance constraints, which indicates that global oversubscription rates are not suitable for this task. (2) Coin shows siginificant improvements in profits compared to baselines while satisfying the chance constraints on Ticket-Cost ratio (safety risk). This highlightd how Coin is extremely effective in designing profitable oversubscription policies while ensuring balanced safety in Ticket booking answering **Q1** & **Q2** affirmatively.

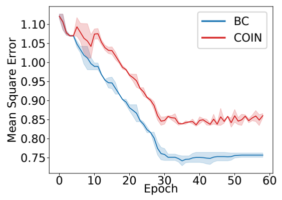


(a) Avg. Mean Square Err

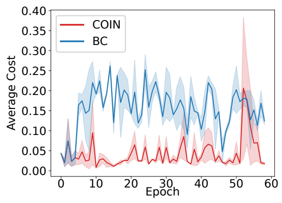


(b) Average Hot nodes

Figure 4: Convergence curves for vCPU Oversubscription



(a) Mean Square Error.



(b) Average Cost

Figure 5: Convergence curves for Airline Overbooking

4.3 Convergence Study

Our policy learner should have stable convergence properties under uncertainty **[Q3]**, such that (1) the overhead cost of training policies does not exceed the savings in operational cost and (2) The policy is well balanced and ensures both "safety" and saving. Convergence analysis on both vCPU as well as Air ticket domains under uncertainty between Coinand a stable baseline (Behavior Cloning or BC) is presented in Figure 4(a) & 4(b).

Coin converges to a stable mean squared error almost at the same epoch range as baseline BC in the Cloud domain. However BC drops faster towards the earlier epochs. This indicates that training with stochastic constraints may be a bit slower (Similar observation in case of hot nodes Fig 4(b) even though Coin is significant better post convergence). Figure 5(b) shows that BC continuously has the higher average costs than Coin on airline data.

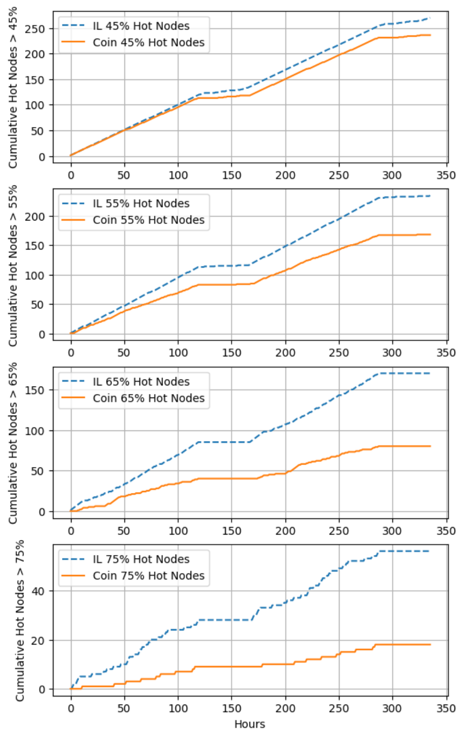


Figure 6: A/B test Results on Microsoft internal services

4.4 Coin in Practice - Internal Deployment

We have deployed our framework on Microsoft internal cloud services for 3 regions. It determines the appropriate oversubscription rate for each vCPU request made by users to reduce resource wastage while ensuring safety against uncertain demand.

We compare the performance of Coin against the strongest baseline, behavior cloning on production deployment and and made observations over a 350-hours period. This service has been deployed in approximately 300 clusters. Figure 6 shows the AB tests to analyze safety. The y-axis shows the cumulative number of hot nodes during the test time at different hot thresholds (45%, 55%, 65%, and 75%). We observe that our method can consistently achieve smaller hot node ratio compared with the baseline. *Again Coin improves vCPU utilization by **saving** 31.05× **more vCores** compared to the existing static naïve oversub. policy in production (answering **Q4**).* Note, other adaptive baselines cannot be tested on production.

5 CONCLUSION

We presented our 'first of its kind' chance-constrained imitation learning framework that can implicitly model uncertainty and learn balanced safe yet profitable oversubscription policies. This leads to substantial COGS savings with nominal congestion risk in test and real deployment. Adaptive oversubscription with stochastic bounds on risk is very important in real systems since industry cannot adopt strict risk avoidance (hard constraints) yet make reasonable profits. As future work we plan to design an end-to-end oversubscription framework that can not only leverage utilization patterns but can also exploit workload forecasts.

REFERENCES

[1] Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U Rajendra Acharya, et al. 2021. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion* 76 (2021), 243–297.

[2] Nilabhra Banerjee, Alec Morton, and Kerem Akartunalı. 2020. Passenger demand forecasting in scheduled transportation. *European Journal of Operational Research* 286, 3 (2020), 797–810.

[3] Salman A Baset, Long Wang, and Chunqiang Tang. 2012. Towards an understanding of oversubscription in cloud. In *2nd USENIX Workshop on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services (Hot-ICE 12)*.

[4] David Breitgand and Amir Epstein. 2012. Improving consolidation of virtual machines with risk-aware bandwidth oversubscription in compute clouds. In *2012 Proceedings IEEE INFOCOM*. IEEE, 2861–2865.

[5] Faruk Caglar and Aniruddha Gokhale. 2014. iOverbook: intelligent resource-overbooking to support soft real-time applications in the cloud. In *2014 IEEE 7th International Conference on Cloud Computing*. IEEE, 538–545.

[6] Jie Chen, Chun Cao, Ying Zhang, Xiaoxing Ma, Haiwei Zhou, and Chengwei Yang. 2018. Improving cluster resource efficiency with oversubscription. In *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, Vol. 1. IEEE, 144–153.

[7] Jie Chen, Chun Cao, Ying Zhang, Xiaoxing Ma, Haiwei Zhou, and Chengwei Yang. 2018. Improving cluster resource efficiency with oversubscription. In *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, Vol. 1. IEEE, 144–153.

[8] Maxime C Cohen, Philipp W Keller, Vahab Mirrokni, and Morteza Zadimoghaddam. 2019. Overcommitment in cloud services: Bin packing with chance constraints. *Management Science* 65, 7 (2019), 3255–3271.

[9] Yuchen Cui, David Isele, Scott Niekum, and Kikuo Fujimura. 2019. Uncertainty-Aware Data Aggregation for Deep Imitation Learning. In *2019 International Conference on Robotics and Automation (ICRA)*. 761–767.

[10] Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. 2018. Safe exploration in continuous action spaces. *arXiv preprint arXiv:1801.08757* (2018).

[11] Pim De Haan, Dinesh Jayaraman, and Sergey Levine. 2019. Causal confusion in imitation learning. *Advances in Neural Information Processing Systems* 32 (2019).

[12] Christopher Diehl, Janis Adamek, Martin Krüger, Frank Hoffmann, and Torsten Bertram. 2022. Differentiable Constrained Imitation Learning for Robot Motion Planning and Control. *arXiv preprint arXiv:2210.11796* (2022).

[13] Peter Englert, Alexandros Paraschos, Jan Peters, and Marc Peter Deisenroth. 2013. Model-based imitation learning by probabilistic trajectory matching. In *2013 IEEE International Conference on Robotics and Automation*. 1922–1927.

[14] Javier Garcıa and Fernando Fernández. 2015. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research* 16, 1 (2015), 1437–1480.

[15] Peter Geibel and Fritz Wysotzki. 2005. Risk-sensitive reinforcement learning applied to control under constraints. *Journal of Artificial Intelligence Research* 24 (2005), 81–108.

[16] Rahul Ghosh and Vijay K Naik. 2012. Biting off safely more than you can chew: Predictive analytics for resource over-commit in iaas cloud. In *2012 IEEE Fifth International Conference on Cloud Computing*. IEEE, 25–32.

[17] A Gosavi, N Bandla, and T Das. 2002. Airline seat allocation among multiple fare classes with overbooking. *IIE Transactions* 34, 9 (2002), 729–742.

[18] Venu Govindaraju, Vijay Raghavan, and Calyampudi Radhakrishna Rao. 2015. *Big data analytics*. Elsevier.

[19] Ori Hadary, Luke Marshall, Ishai Menache, Abhisek Pan, Esaias E Greeff, David Dion, Star Dorminey, Shailesh Joshi, Yang Chen, Mark Russinovich, and Thomas Moscibroda. 2020. Protean: VM Allocation Service at Scale. In *14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20)*. USENIX Association, 845–861. https://www.usenix.org/conference/osdi20/presentation/hadary

[20] Rachel Householder, Scott Arnold, and Robert Green. 2014. On cloud-based oversubscription. *arXiv preprint arXiv:1402.4758* (2014).

[21] Ke-Wei Huang and Mengqi Wang. 2009. Firm-level productivity analysis for software as a service companies. *ICIS 2009 proceedings* (2009), 21.

[22] Eyke Hüllermeier and Willem Waegeman. 2021. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning* 110 (2021), 457–506.

[23] Pierre Jacquet, Thomas Ledoux, and Romain Rouvoy. 2024. ScroogeVM: Boosting Cloud Resource Utilization with Dynamic Oversubscription. *IEEE Transactions on Sustainable Computing* (2024), 1–13. https://doi.org/10.1109/TSUSC.2024.3369333

[24] Kento Kawaharazuka, Yoichiro Kawamura, Kei Okada, and Masayuki Inaba. 2021. Imitation Learning With Additional Constraints on Motion Style Using Parametric Bias. *IEEE Robotics and Automation Letters* 6, 3 (2021), 5897–5904. https://doi.org/10.1109/LRA.2021.3087423

[25] Alok Gautam Kumbhare, Reza Azimi, Ioannis Manousakis, Anand Bonde, Felipe Frujeri, Nithish Mahalingam, Pulkit A Misra, Seyyed Ahmad Javadi, Bianca Schroeder, Marcus Fontoura, et al. 2021. {Prediction-Based} Power Oversubscription in Cloud Platforms. In *2021 USENIX Annual Technical Conference (USENIX ATC 21)*. 473–487.

[26] Ryan J. Lawhead and Abhijit Gosavi. 2019. A bounded actor–critic reinforcement learning algorithm applied to airline revenue management. *Engineering Applications of Artificial Intelligence* 82 (2019), 252–262.

[27] Zhihua Li. 2019. An adaptive overload threshold selection process using Markov decision processes of virtual machine in cloud data center. *Cluster Computing* 22, 2 (2019), 3821–3833.

[28] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2015. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971* (2015).

[29] Owen Lockwood and Mei Si. 2022. A Review of Uncertainty for Deep Reinforcement Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*.

[30] Nihar R Mahapatra and Balakrishna Venkatrao. 1999. The processor-memory bottleneck: problems and solutions. *Crossroads* 5, 3es (1999), 2.

[31] Nina Mazyavkina, Sergey Sviridov, Sergei Ivanov, and Evgeny Burnaev. 2021. Reinforcement learning for combinatorial optimization: A survey. *Computers & Operations Research* 134 (2021), 105400.

[32] Kunal Menda, Katherine Driggs-Campbell, and Mykel J. Kochenderfer. 2019. EnsembleDAgger: A Bayesian Approach to Safe Imitation Learning. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 5041–5048. https://doi.org/10.1109/IROS40897.2019.8968287

[33] Amin Nazifi, Katja Gelbrich, Yany Grégoire, Sebastian Koch, Dahlia El-Manstrly, and Jochen Wirtz. 2021. Proactive handling of flight overbooking: how to reduce negative eWOM and the costs of bumping customers. *Journal of Service Research* 24, 2 (2021), 206–225.

[34] Amin Nazifi, Katja Gelbrich, Yany Grégoire, Sebastian Koch, Dahlia El-Manstrly, and Jochen Wirtz. 2021. Proactive Handling of Flight Overbooking: How to Reduce Negative eWOM and the Costs of Bumping Customers. *Journal of Service Research* 24, 2 (2021), 206–225. https://doi.org/10.1177/1094670520933683

[35] Santiago Paternain, Miguel Calvo-Fullana, Luiz FO Chamon, and Alejandro Ribeiro. 2019. Learning safe policies via primal-dual methods. In *2019 IEEE 58th Conference on Decision and Control (CDC)*. IEEE, 6491–6497.

[36] Santiago Paternain, Luiz Chamon, Miguel Calvo-Fullana, and Alejandro Ribeiro. 2019. Constrained reinforcement learning has zero duality gap. *Advances in Neural Information Processing Systems* 32 (2019).

[37] Baiyu Peng, Jingliang Duan, Jianyu Chen, Shengbo Eben Li, Genjin Xie, Congsheng Zhang, Yang Guan, Yao Mu, and Enxin Sun. 2022. Model-Based Chance-Constrained Reinforcement Learning via Separated Proportional-Integral Lagrangian. *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[38] Baiyu Peng, Yao Mu, Yang Guan, Shengbo Eben Li, Yuming Yin, and Jianyu Chen. 2021. Model-based actor-critic with chance constraint for stochastic system. In *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, 4694–4700.

[39] Dean A Pomerleau. 1991. Efficient training of artificial neural networks for autonomous navigation. *Neural computation* (1991).

[40] Mohammad A. Salahuddin, Ala Al-Fuqaha, and Mohsen Guizani. 2016. Reinforcement learning for resource provisioning in the vehicular cloud. *IEEE Wireless Communications* 23, 4 (2016), 128–135. https://doi.org/10.1109/MWC.2016.7553036

[41] Harsh Satija, Philip Amortila, and Joelle Pineau. 2020. Constrained markov decision processes via backward value functions. In *International Conference on Machine Learning*. PMLR, 8502–8511.

[42] Mohammad Shahrad, Rodrigo Fonseca, Inigo Goiri, Gohar Chaudhry, Paul Batum, Jason Cooke, Eduardo Laureano, Colby Tresness, Mark Russinovich, and Ricardo Bianchini. 2020. Serverless in the Wild: Characterizing and Optimizing the Serverless Workload at a Large Cloud Provider. In *2020 USENIX Annual Technical Conference (USENIX ATC 20)*. USENIX Association, 205–218. https://www.usenix.org/conference/atc20/presentation/shahrad

[43] Junjie Sheng, Lu Wang, Fangkai Yang, Bo Qiao, Hang Dong, Xiangfeng Wang, Bo Jin, Jun Wang, Si Qin, Saravan Rajmohan, et al. 2022. Learning Cooperative Oversubscription for Cloud by Chance-Constrained Multi-Agent Reinforcement Learning. *arXiv preprint arXiv:2211.11759* (2022).

[44] Syed Arbab Mohd Shihab, Caleb Logemann, Deepak-George Thomas, and Peng Wei. 2019. Autonomous airline revenue management: A deep reinforcement learning approach to seat inventory control and overbooking. *arXiv preprint arXiv:1902.06824* (2019).

[45] E. Shlifer and Y. Vard. 1975. An Airline Overbooking Policy. *Transportation Science* 9, 2 (1975), 101–114.

[46] João Silvério, Yanlong Huang, Fares J. Abu-Dakka, Leonel Rozo, and Darwin G. Caldwell. 2019. Uncertainty-Aware Imitation Learning using Kernelized Movement Primitives. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

[47] Erma Suryani, Shuo-Yan Chou, and Chih-Hsien Chen. 2010. Air passenger demand forecasting and passenger terminal capacity expansion: A system dynamics framework. *Expert Systems with Applications* 37, 3 (2010), 2324–2339.

[48] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.

[49] Yoshinori Suzuki. 2006. The net benefit of airline overbooking. *Transportation Research Part E: Logistics and Transportation Review* 42, 1 (2006), 1–19.

[50] Hui Wang and Huaglory Tianfield. 2018. Energy-aware dynamic virtual machine consolidation for cloud datacenters. *IEEE Access* 6 (2018), 15259–15273.

[51] Tingsong Wang, Zheng Xing, Hongtao Hu, and Xiaobo Qu. 2019. Overbooking and delivery-delay-allowed strategies for container slot allocation. *Transportation Research Part E: Logistics and Transportation Review* 122 (2019), 433–447. https://doi.org/10.1016/j.tre.2018.12.019