Inventory Control with Partially Observable States

Erli Wang[a,b], **Hanna Kurniawati**[a], and **Dirk P. Kroese**[b]

[a]*Research School of Computer Science, the Australian National University, Australia*
[b]*School of Mathematics and Physics, the University of Queensland, Australia*

Email: erli.wang@anu.edu.au

Abstract:

Consider a retailer who buys a range of commodities from a wholesaler and sells them to customers. At each time period, the retailer has to decide how much of each type of commodity to purchase, so as to maximize some overall profit. This requires a balance between maximizing the amount of high-valued customer demands that can be fulfilled and minimizing storage and delivery costs. Due to inaccuracies in inventory recording, misplaced products, market fluctuation, etc., the above purchasing decisions must be made in the presence of partial observability on the amount of stocked goods and on uncertainty in the demand. A natural framework for such an inventory control problems is the Partially Observable Markov Decision Process (POMDP).

Key to POMDP is that it decides the best actions to perform with respect to distributions over states, rather than a single state. Finding the optimal solution of a POMDP problem is computationally intractable, but the past decade has seen substantial advances in finding approximately optimal POMDP solutions and POMDP has started to become practical for many interesting problems.

Despite advances in approximate POMDP solvers, they do not perform well on most inventory control problems, due to the massive action space (i.e., purchasing possibilities) of most such problems. Most POMDP-based methods limit the problem to a one-commodity scenario, which is far from reality.

In this paper, we apply our recent POMDP-based method (QBASE) to multi-commodity inventory control. QBASE combines Monte Carlo Tree Search with quantile statistics based Monte Carlo, namely the Cross-Entropy method for optimization, to quickly identify good actions without sweeping through the entire action space. It enables QBASE to substantially scale up our ability to compute good solutions to POMDPs with extremely large discrete action spaces (in the order of a million discrete actions).

We compare our solution to several commonly used inventory control methods, such as the (s, S) method and other state-of-the-art POMDP solvers. The results are promising, as it demonstrates smarter purchasing behaviors. For instance, if we combine maximum likelihood with the commonly used (s, S) policy, the latter policy is very far from optimal when the uncertainty must be represented as a non-unimodal distribution. Furthermore, the state-of-the-art POMDP solver can only generate a sub-optimal policy of always keeping the stocks of all commodities at a relatively high level, to meet as many demands as possible. In contrast, QBASE can generate a better policy, whereby a small amount of sales are sacrificed to keep the inventory level of commodities with expensive storage cost and low value, to be as low as possible, which then lead to a higher profit.

Keywords: *inventory control problem, multi-commodity, partially observable Markov decision process, on-line POMDP solver*

1 INTRODUCTION

The goal of the inventory control problem is to maintain stocks of commodities (goods) at an appropriate level so that the inventory costs can be minimized, while revenues from demand fulfillment can be maximized. This problem has been well-studied for fully-observed systems; see, e.g., (Resh and Naor, 1963) and (Veinott Jr and Wagner, 1965). However, in practice, many issues cause the systems to only be partially observed. For instance, distortion of inventory data is a major problem in supply chain management, costing billions of dollars lost (Consulting, 1996). Moreover, Kang and Gershwin (2005) found 51% of inventory accuracy in a global retailer, while DeHoratius and Raman (2008) found that the accuracy of over $350K$ inventory recordings from over 37 retail stores from large retailers is only 35%. In summary, partial observability is mainly caused by inventory record inaccuracy and misplaced products. Such errors cause the inventory autonomous system to order insufficient products or too many products, which lowers profit.

Multiple approaches have been proposed to alleviate the above problems (DeHoratius et al., 2008). This paper focuses on one of the promising approaches, which is to apply a robust decision making framework that considers partial observability on the stocked good.

The Partially Observable Markov Decision Process (POMDP) is a general and mathematically principled framework for decision making in the presence of partial observability (Sondik, 1971). Key to POMDP is that, due to uncertainty in the effect of actions and in perception, the system never knows its exact state. Instead, one keeps track of a distribution over the state space, called a belief, and the objective is to choose the best action to perform for each belief. Although computing the optimal POMDP solution is computationally intractable (Papadimitriou and Tsitsiklis, 1987), approximate methods for solving POMDPs have advanced tremendously in the past decade (Pineau et al., 2003; Kurniawati et al., 2008; Silver and Veness, 2010; Somani et al., 2013; Wang et al., 2018; Sunberg and Kochenderfer, 2018).

During the last decade, researchers on replenishment systems have started to use the POMDP framework (DeHoratius et al., 2008; Zhou et al., 2010; Bayraktar and Ludkovski, 2010; Saghafian, 2018). However, most of their work focuses on single-commodity problems. The reason is that the size of the action space of a POMDP for inventory control problems grows exponentially in the number of commodities, and even state-of-the-art approximate POMDP solvers today (Silver and Veness, 2010) perform poorly on problems with large action (more than 50 discrete actions) spaces.

This paper proposes to alleviate the difficulty of solving multi-commodities inventory control problem in the presence of partial observability. We frame the problem as a POMDP problem. To keep the problem simple, we assume that the demand distribution is stationary. We apply our recent on-line POMDP solver, Quantile-Based Action Selector (QBASE) (Wang et al., 2018), to alleviate the difficulties of finding good POMDP solutions for extremely large action space. QBASE combines Monte Carlo Tree Search (Browne et al., 2012) with the Cross-Entropy Method for optimization (Rubinstein and Kroese, 2004), to quickly identify good actions without sweeping through the entire action space. We demonstrate the capability of this new solver on an inventory control problem, framed as a POMDP with up to one million discrete action. Comparison with commonly used techniques for inventory control and state-of-the-art POMDP solvers are promising.

2 POMDPs

A POMDP is described by a 6-tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, T, Z, R \rangle$, where $\mathcal{S}$ is the set of *states*, $\mathcal{A}$ is the set of *actions*, and $\mathcal{O}$ is the set of *observations*. At each step, the agent is in some hidden state $s \in \mathcal{S}$, takes an action $a \in \mathcal{A}$, and moves from s to another state $s' \in \mathcal{S}$ according to a transition density $T(s, a, s') = f(s' \mid s, a)$. The current state s' is then partially revealed via an observation o drawn from a conditional probability density $Z(s', a, o) = f(o \mid s', a)$ that represents uncertainty in sensing. After each step, the agent receives a reward $R(s, a)$, if it takes action a from state s.

Now, due to the uncertainty in the effect of action and in sensing, the agent never knows its exact state. Instead, it maintains an estimate of its current state in the form of a *belief* b, which is a probability density on $\mathcal{S}$. The initial belief of the agent is denoted as b_0. The agent updates its belief at the end of each step, in a Bayesian manner, based on the belief at the beginning of the step along with the action and observation that have just been performed and perceived in this step.

The objective of a POMDP agent is to maximize its expected total reward (called the *value function*), by following at each time step the best *policy* — a mapping from beliefs to actions. Each policy induces a value

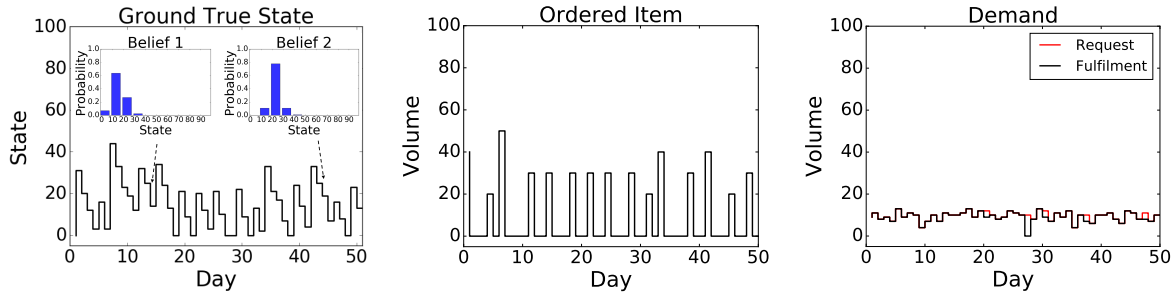


Figure 1. Illustration of ***InventoryControl***(1). (a) The ground true state is not fully observable. Instead, the agent maintains an estimate of the current state as a probability distribution over the state space. Although the ground truth state is the same (e.g., $s = 25$), the belief over the state space can be quite different. For example, the probability in the belief 1 masses in the [10,20], which leads the decision maker to order new items immediately the next day. The situation is different for the belief 2. The owner believes that there are still sufficient goods so will not replenish at the moment. (b) Considering the cost of ordering, the decision maker should prefer ordering a large number of items at once rather than frequently ordering a small amount. (c) The red line denotes the demand from the customer and the black line shows the actual fulfilment. The profits of the shop owner come only from the sale, therefore, one should avoid insufficient supply. The goal is to keep the stock state at an appropriate level. If the stock level is too high, there is a large holding cost. If the stock is too low, one may miss the sale.

function V, and the best policy gives rise to a value function V^* that satisfies:

$$V^*(b) = \max_{a \in A} \underbrace{\left[\sum_{s \in \mathcal{S}} R(s,a) b(s) + \gamma \sum_{o \in O} f(o \mid b, a) V^*(\tau(b,a,o)) \right]}_{\text{Q-value function: } Q(b,a)} \tag{1}$$

where $f(o \mid b, a)$ represents the expected probability of perceiving observation $o \in \mathcal{O}$ after performing action $a \in \mathcal{A}$ from belief b, and is computed as $\sum_{s', s \in \mathcal{S}} f(o \mid s', a)\ f(s' \mid s, a) b(s)$. The notation $\tau(b, a, o)$ represents the new belief of the agent, after it performs action a and perceives observation o, and is computed as $\tau(b,a,o)(s') \propto \sum_{s \in \mathcal{S}} Z(s', a, o)\ T(s, a, s')\ b(s)$. When the planning horizon is infinite, to ensure the problem is well defined, rewards at subsequent time steps are discounted by a factor $0 < \gamma < 1$.

3 PROBLEM FORMULATION

Think of an agent in ***InventoryControl***(K) as a shop owner, where the shop provides multiple commodities (K) that are for sale. In order to maximize the cumulative profit, the manager at each step (e.g., day) decides the type and quantity of items to order based on the prediction of future demand. Figure 1 shows a sample run of a computed policy for one commodity. The state of commodity k is denoted by s_k, where $s_k \in \{0, 1, 2, \ldots, 100\}, k = 1, 2, \ldots, K$. The state space is the Cartesian product of the stock level for all commodities. The action $a_k \in \{0, 10, \ldots, 90\}$ corresponds to the amount of stock that is ordered from suppliers for each commodity $k = 1, 2, \ldots, K$. Similarly, the action space is $\mathcal{A} = \{(a_1, \ldots, a_K)\}$, where $a_k \in \{0, 10, \ldots, 90\}, k = 1, 2, \ldots, K$. We assume the number of orderings is perfect and that all items ordered will be delivered in good condition on time (i.e., before the opening of a step).

At the start of the next step, inventory is either replenished by ordering certain goods due to the sales from the current stage or is not replenished. During this time period, the customer demand for each commodity is stochastic from some distribution, say Gaussian. This implies that the next state is computed as

$$\boldsymbol{s}' = \max\{\mathbf{0}, \boldsymbol{s} + \boldsymbol{a} - \boldsymbol{D}\}, \tag{2}$$

where $\boldsymbol{D} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$. The correlation coefficients ρ_{cor} in Σ are assumed to be greater than 0.

The observation space is $\mathcal{O} = \{(o_1, \ldots, o_K)\}$, where $k = 1, 2, \ldots, K$. According to industrial surveys, the inaccuracy of observations should be reflected by various types of errors. We assume that each observation o_k is perceived as a categorical variable taking values in the set $\{0, 1, 2\}$, corresponding to Low, Normal, and High stock levels. Since many retail chains do not track the location of items, the placement of items relies on

the employees' memories. Because of memory bias, employees tend to keep a good memory of stock levels when these are at either a low or high level. We model the observation distribution for the stock level s'_k as follows. Let $\mu(s'_k) = 0$ if $s'_k \in [0, 33)$, $\mu(s'_k) = 1$ if $s'_k \in [33, 66)$, and $\mu(s'_k) = 2$ if $s'_k \in [66, 100]$. Letting $Z \sim \mathcal{N}(\mu(s'_k), e_Z^2)$, we define $\mathbb{P}(o_k = 0 \,|\, s'_k) = \mathbb{P}(Z \leq 1/2)$, $\mathbb{P}(o_k = 1 \,|\, s'_k) = \mathbb{P}(1/2 < Z \leq 3/2)$, and $\mathbb{P}(o_k = 2 \,|\, s'_k) = \mathbb{P}(Z > 3/2)$.

At the end of the step, the shop owner will need to compute the profit. The reward is defined as the incomes of sales minus the costs in general. For a given stock level $\boldsymbol{s}$ and ordered action $\boldsymbol{a}$, a step profit is defined as $R(\boldsymbol{s}, \boldsymbol{a}) = \sum_{k=1}^{K} \left[p_k \min\{s_k, D_k\} - h_k \max\{0, s_k - \delta_k\} - I_{\{a_k > 0\}}(t_k + b_k\, a_k) \right]$, where $p_k (> 0)$ is the income of a unit stock, $h_k (> 0)$ is the holding cost, $t_k (> 0)$ is the transaction fee if the ordering action happens for commodity k and $b_k (> 0)$ is the cost price.

4 QBASE

To make the best decision, the Q-value must be estimated by considering many possible sequences of future beliefs, which are affected by the future actions selected and observations perceived. This causes the number of belief-contingent plans to increase exponentially with the planning horizon. When the number of commodities is large, solving POMDPs becomes computationally infeasible due to the large action spaces.

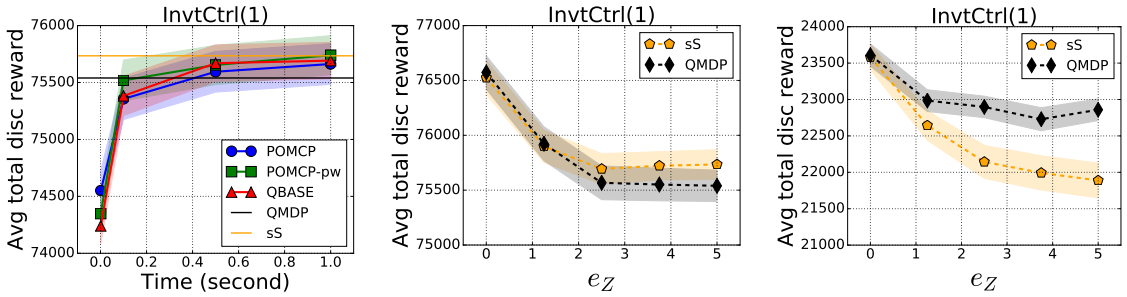
To overcome this issue, we adopt our development in the on-line POMDP algorithm called QBASE, which has the strength of handling problems with large action spaces. For completeness, we provide a brief summary of QBASE's main idea here. See Wang et al. (2018) for details. As an on-line solver, QBASE does a planning phase, executes one step, gets an observation and re-plans again. To find the best action to perform from a belief, QBASE constructs a *belief tree*, denoted as $\mathcal{T}$. A belief tree is a tree where the nodes are sampled beliefs. An edge labeled (a, o) from a belief b to a belief b' in $\mathcal{T}$ means there is an action $a \in \mathcal{A}$ and an observation $o \in \mathcal{O}$ such that $b' = \tau(b, a, o)$. QBASE represents each sampled belief as a set of particles (states) and estimates the value of each sampled belief via Monte Carlo backup. The best action is then the action that induces the best estimated value.

Key to QBASE is the way it samples actions when generating histories to construct the belief tree $\mathcal{T}$. Given a belief node to expand, QBASE maintains a probability distribution function over the action space $\mathcal{A}$ and uses this distribution to sample actions, so as to avoid full enumeration of the action space at the beginning. The question is, given a belief b, what distribution should QBASE use to sample the action space $\mathcal{A}$? QBASE adapts the distribution in two stages. First, it identifies a subset of the action space, where it expects to find good actions. This set contains actions from top $\varrho-$quantile of $\widehat{Q}$ (exploitation) and $|\mathcal{A}_s| - \varrho|\mathcal{A}|$ sampled uniformly (exploration) from the entire action space without considering the exploitation actions. In the second stage, the probability in the subset is updated via a slightly modification of the proportional Cross-Entropy method (Goschin et al., 2013), as this probability reflects QBASE s approximation on the relative differences in Q-values of the different actions. With these two stages, QBASE can avoid complete enumeration by using quantile-based stochastic optimization approach to focus going deeper on promising sub-trees first.

5 EXPERIMENTS

Our experiments have two goals: the first goal is to test all approaches on a single-commodity scenario to understand the properties of each method better. The second and ultimate goal is to test the effectiveness of QBASE on large inventory problems.

We compare QBASE to commonly used inventory control methods and other state-of-the-art POMDP solvers. The (s, S) policy (Veinott Jr and Wagner, 1965) is a classical threshold policy for inventory system where the state of inventory is fully observable. We adapt the (s, S) policy to meet the partially observable settings assuming the true state to be the same as the most likely state in the belief. QMDP (Littman et al., 1995) is an off-line approximate POMDP algorithm. It considers the uncertainty in the current state and assumes the full observability at the subsequent states encountered in the future. POMCP (Silver and Veness, 2010) is considered to be one of the most successful on-line POMDP solvers today. It frames the action selection as a multi-armed bandit and adopts UCB1 (Auer et al., 2002) to handle the exploration-exploitation tradeoff. To ensure the convergence of POMCP, it requires sweeping over the whole action spaces to obtain initial values. This could lead POMCP to perform poorly when the size of action spaces is large. Instead of interacting with the entire action space directly, progressive widening (PW) (Couëtoux et al., 2011) gradually expands a subset of the action space and applies UCB1 over the subset of actions.



(a) (b) (c)

Figure 2. (a) Comparison of off-line and on-line solvers. (b) Gaussian $\mathcal{N}(15, 100/12)$ demand distribution. (c) Uniform $\mathcal{U}(0, 10)$ demand distribution.

The implementation of POMCP follows the authors' code [1]. We build all other on-line methods in POMCP's framework with improving the implementation to support a very large number of action spaces using C++. The experiments are run as a single thread process on an Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10GHz with 128GB RAM. To set the parameters, we execute a preliminary run to identify the best-performing ones. The default policy is computed as the fully observable version of the problem with the assumption that all items are independent. For a fair comparison, all solvers use the same rollout policy for each problem. In particular, the inventory system should be assumed to run forever, as there are no explicit terminal states. But for the convenience of computing the expected total discounted reward, the system will be measured over 50 stages.

5.1 Scenario A: Single-Commodity

Figure 2(a) presents the average expected discounted total reward with 95% confidence interval of the 1,000 simulation runs for each method when the demand distribution is Gaussian with $e_Z = 5$. Overall, given a sufficient planning time, QBASE and all other on-line solvers are able to find comparable performance to QMDP and (s, S). Even though (s, S) is guaranteed to find the optimal policy under fully observable settings, its performance can be fooled in partially observable scenarios. If the demand distribution is Gaussian, the uni-model property can help (s, S) to capture the hidden physical inventory level. It shows that (s, S) performs comparably to QMDP in Figure 2(b). However, Figure 2(c) shows that the performance of (s, S) policy deteriorates significantly in uniform demand problem when the error of observation is large.

QMDP, one of the off-line solvers, has to build the policy π for the entire state space. When the size of the state space and the action space are large, constructing policies in such a way can be computationally inefficient. However, on-line algorithms, such as QBASE and POMCP, interleave a planning phase with an execution step. In other words, they only need to figure out the best action (or policy) with respect to the current system state. It brings the benefit that the memory requirement of an on-line solver is dramatically reduced.

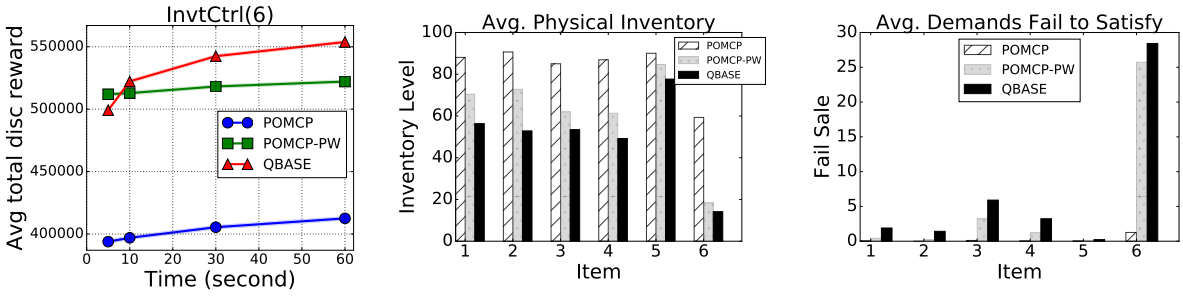
5.2 Scenario B: Multi-Commodity

Table 1 shows settings for a ***InventoryControl***(6) problem. $\widetilde{\boldsymbol{D}}_k$ denotes the marginal distribution of $\boldsymbol{D}$ for each type of item. For other parameters, we let $\rho_{cor} = 0.8$, $e_Z = 5$ and the initial state of all commodities to be 0. Considering the size of state spaces and action spaces are huge, we only output comparison across on-line solvers.

Table 1. Settings of ***InventoryControl***(6)

Item	p_k	h_k	δ_k	$\widetilde{\boldsymbol{D}}_k$	t_k	b_k
1	200	5	20	$\mathcal{N}(15, 2.5^2)$	1000	10
2	200	5	20	$\mathcal{N}(10, 2^2)$	1000	10
3	300	5	0	$\mathcal{N}(20, 5^2)$	2000	0
4	300	5	0	$\mathcal{N}(15, 2.5^2)$	2000	0
5	300	5	0	$\mathcal{N}(15, 2.5^2)$	500	0
6	200	20	0	$\mathcal{N}(15, 2.5^2)$	500	10

[1] http://www0.cs.ucl.ac.uk/staff/d.silver/web/Applications.html



(a) (b) (c)

Figure 3. (a) Performance of on-line solvers with different planning time. All experiments are repeated 1,000 times and presented the average expected discounted total reward with 95% confidence interval for each method. (b) Average physical inventory level for each commodity. (c) Average number of sale loss.

Table 2. Simulation Results

Scenario	Time/Step (sec)	Method	Avg. Total Profit (Avg. Undisc. Tot. Reward)
InventoryControl(6)			
$\|\mathcal{S}\| = 10^{12}$, $\|\mathcal{A}\| = 10^6$, $\|\mathcal{O}\| = 729$	30	POMCP	636794.5 ± 2174.3
		POMCP-PW	816575.4 ± 2003.8
		QBASE	855365.5 ± 1846.2

In Table 2, QBASE achieves the best performance. As the size of the problem increases, the performance gap between QBASE and other two methods increases. Figure 3(a) presents how the planning time per step affects performance in the scenarios. The trend is promising. As we allow additional planning time, QBASE can significantly improve on POMCP. This is because POMCP explodes on expanding new actions in the initialization stage, which means that the tree cannot go deeper to explore any useful information. To overcome this issue, QBASE and POMCP-PW confirm that partial enumeration methods work well for problems with large action spaces. Interestingly, POMCP-PW gets the best performance in ***InventoryControl***(6) when the planning time is small (t = 5s). However, by providing extra budgets, POMCP-PW can only slightly improve performance further. One possible reason is that the PW explores a new action at the cost of the number of simulations growing exponentially. It causes the number of actions can be sampled in practice is limited. In contrast, the probability of sampling a starved action in QBASE will be always fixed. This does not prevent QBASE to expand the tree wider to keep increasing the performance when additional planning time is provided.

Figure 3(b) and (c) show the behavior from the computed policy by each method. The general information shows that POMCP is more averse to lose customer's demands. However, according to performance, its policy is not the best one. Due to the large action spaces, POMCP can not capture the information about the costs and potential sales in the subsequent steps. Therefore, it naively keeps the inventory at an unnecessarily high level to avoid any possible missed sales at the current step. Instead, QBASE prefers to keep the physical inventory as low as possible while it may miss some sales. It seems not to be a good idea, but it returns a much better performance than POMCP's. For example, the 6th item has relatively low stock and a very high failure to satisfy the demand. This is because the price per sale is only 200 while the holding costs and cost prices are expensive, which prevents the system to maintain at a high inventory level.

6 CONCLUSIONS

Industrial investigations indicate that errors in inventory recordings are common and often unavoidable. Such errors result in dramatical wastes and cost to the industry. Although rigorous mathematical framework — POMDPs— for inventory control under partially observed inventory is well-known, it has not been used in practice due to the high computational complexity in applying such a framework to realistic inventory control problems, where multiple commodities are being considered. This paper alleviates the issue. In particular, this paper demonstrates the applicability of our recently developed method, QBASE, in generating good POMDP strategies for multi-commodity inventory control problems. The results are promising, indicating that our proposed solution can find less conservative inventory control strategies that yield higher profits, compared to existing solutions. Future avenues abound, including in scaling up the method further and in applying them to real industrial problems. We hope this paper has provided a step forward towards robust inventory control necessary in many industries.

ACKNOWLEDGEMENT

This work was supported by the Australian Research Council Centre of Excellence for Mathematical and Statistical Frontiers (ACEMS) under grant number CE140100049. This work was partially supported by ONR NICOP Project Number N62909-17-1-2046. Erli Wang would also like to acknowledge the support from UQ through the UQ International Scholarships scheme.

REFERENCES

Auer, P., Cesa-Bianchi, N., and Fischer, P. (2002). Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256.

Bayraktar, E. and Ludkovski, M. (2010). Inventory management with partially observed nonstationary demand. *Annals of Operations Research*, 176(1):7–39.

Browne, C. B., Powley, E., Whitehouse, D., Lucas, S. M., Cowling, P. I., Rohlfshagen, P., Tavener, S., Perez, D., Samothrakis, S., and Colton, S. (2012). A survey of Monte Carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43.

Consulting, A. (1996). Where to look for incremental sales gains: The retail problem of out-of-stock merchandise. *The Coca-Cola Retailing Research Council, Atlanta, GA*.

Couëtoux, A., Hoock, J.-B., Sokolovska, N., Teytaud, O., and Bonnard, N. (2011). Continuous upper confidence trees. In *International Conference on Learning and Intelligent Optimization*, pages 433–445. Springer.

DeHoratius, N., Mersereau, A. J., and Schrage, L. (2008). Retail inventory management when records are inaccurate. *Manufacturing & Service Operations Management*, 10(2):257–277.

DeHoratius, N. and Raman, A. (2008). Inventory record inaccuracy: an empirical analysis. *Management Science*, 54(4):627–641.

Goschin, S., Weinstein, A., and Littman, M. L. (2013). The Cross-Entropy method optimizes for quantiles. In *International Conference on Machine Learning*, pages 1193–1201.

Kang, Y. and Gershwin, S. B. (2005). Information inaccuracy in inventory systems: stock loss and stockout. *IIE transactions*, 37(9):843–859.

Kurniawati, H., Hsu, D., and Lee, W. S. (2008). SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In *Robotics: Science and Systems*.

Littman, M. L., Cassandra, A. R., and Kaelbling, L. P. (1995). Learning policies for partially observable environments: Scaling up. In *Machine Learning Proceedings 1995*, pages 362–370. Elsevier.

Papadimitriou, C. H. and Tsitsiklis, J. N. (1987). The complexity of Markov decision processes. *Mathematics of operations research*, 12(3):441–450.

Pineau, J., Gordon, G., and Thrun, S. (2003). Point-based Value Iteration: An anytime algorithm for POMDPs. In *International Joint Conference on Artificial Intelligence*.

Resh, M. and Naor, P. (1963). An inventory problem with discrete time review and replenishment by batches of fixed size. *Management Science*, 10(1):109–118.

Rubinstein, R. Y. and Kroese, D. P. (2004). *The Cross-Entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation and machine learning*. Springer.

Saghafian, S. (2018). Ambiguous partially observable markov decision processes: Structural results and applications. *Journal of Economic Theory*, 178:1–35.

Silver, D. and Veness, J. (2010). Monte-Carlo planning in large POMDPs. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2164–2172.

Somani, A., Ye, N., Hsu, D., and Lee, W. S. (2013). DESPOT: Online POMDP planning with regularization. In *Advances in Neural Information Processing Systems*, pages 1772–1780.

Sondik, E. J. (1971). *The optimal control of partially observable Markov processes*. PhD thesis, Stanford University.

Sunberg, Z. and Kochenderfer, M. (2018). Online algorithms for POMDPs with continuous state, action, and observation spaces. In *Internatonal Conference on Automated Planning and Scheduling*, pages 259–263.

Veinott Jr, A. F. and Wagner, H. M. (1965). Computing optimal (s, s) inventory policies. *Management Science*, 11(5):525–552.

Wang, E., Kurniawati, H., and Kroese, D. P. (2018). An on-line planner for POMDPs with large discrete action space: A quantile-based approach. In *Internatonal Conference on Automated Planning and Scheduling*, pages 273–277.

Zhou, E., Fu, M. C., and Marcus, S. I. (2010). Solving continuous-state pomdps via density projection. *IEEE Transactions on Automatic Control*, 55(5):1101–1116.