Forward-PECVaR Algorithm: Exact Evaluation for CVaR SSPs

Extended Abstract

Willy Arthur Silva Reis
University of São Paulo
Sao Paulo, Brazil
willy.reis@usp.br

Denis Benevolo Pais
University of São Paulo
Sao Paulo, Brazil
denis.pais@alumni.usp.br

Valdinei Freire
University of São Paulo
Sao Paulo, Brazil
valdinei.freire@usp.br

Karina Valdivia Delgado
University of São Paulo
Sao Paulo, Brazil
kvd@usp.br

KEYWORDS

Conditional Value at Risk; Stochastic Shortest Path; Sequential Decision Making; Probabilistic Planning

ACM Reference Format:
Willy Arthur Silva Reis, Denis Benevolo Pais, Valdinei Freire, and Karina Valdivia Delgado. 2023. Forward-PECVaR Algorithm: Exact Evaluation for CVaR SSPs: Extended Abstract. In *Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), London, United Kingdom, May 29 – June 2, 2023*, IFAAMAS, 3 pages.

1 INTRODUCTION

The Stochastic Shortest Path (SSP) problem models probabilistic sequential-decision problems where an agent must pursue a goal while minimizing a cost function. Because of the probabilistic dynamics, it is desired to have a cost function that considers risk. Conditional Value at Risk (CVaR) is a coherent risk measure [7] criterion that allows modeling an arbitrary level of risk by considering the expectation of a fraction α of worse trajectories [1, 4, 8].

Although an optimal policy is non-Markovian, solutions of CVaR-SSP can be found approximately with Value Iteration based algorithms such as CVaR Value Iteration with Linear Interpolation (CVaRVILI) [4] and CVaR Value Iteration via Quantile Representation (CVaRVIQ) [8]. These type of solutions depends on the algorithm's parameters such as the number of atoms and α_0 (the minimum α). To compare the policies returned by these algorithms, we need a way to exactly evaluate the stationary policies of CVaR-SSPs. Although there is an algorithm that evaluates these policies, this only works on problems with uniform costs [6].

In this work, we propose a new algorithm, Forward-PECVaR (ForPECVaR), that evaluates exactly stationary policies of CVaR-SSPs with non-uniform costs. We evaluate empirically CVaR Value Iteration algorithms that found solutions approximately regarding their quality compared with the exact solution, and the influence of the algorithm parameters in the quality and scalability of the solutions. Experiments in two domains show that it is important to use an α_0 smaller than the α target and an adequate number of atoms to obtain a good approximation. The full paper is available at https://arxiv.org/abs/2303.00672.

Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), A. Ricci, W. Yeoh, N. Agmon, B. An (eds.), May 29 – June 2, 2023, London, United Kingdom.

2 BACKGROUND

A Stochastic Shortest Path Problem [3] is described by a tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, P, c, \mathcal{G} \rangle$ where: $\mathcal{S}$ is a finite set of states; $\mathcal{A}$ is a finite set of actions; $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is a transition function that represents the probability that $s' \in \mathcal{S}$ is reached after the agent executes an action $a \in \mathcal{A}$ in a state $s \in \mathcal{S}$, i.e., $\Pr(s_{t+1} = s' | s_t = s, a_t = a) = P(s, a, s')$; $c : \mathcal{S} \times \mathcal{A} \to \mathcal{R}^+$ is a positive cost function that represents the cost of executing an action $a \in \mathcal{A}$ in a state $s \in \mathcal{S}$, i.e., $c_t = c(s_t, a_t)$; and $\mathcal{G}$ is a non-empty set of goal states that are absorbing, i.e., $P(s_{t+1} \in \mathcal{G} | s_t \in \mathcal{G}, a_t = a) = 1$ and $c(s_t \in \mathcal{G}, a_t = a) = 0$ for all $a \in \mathcal{A}$.

The solution to an SSP is a policy π that could be stationary ($\pi : \mathcal{S} \to \mathcal{A}$) or non-Markovian (history-dependent). Let the random variable $Z_M = \sum_{t=0}^{M} c(s_t, \pi(s_t))$ be the accumulated cost from time 0 up to time M. The value function of a policy π is defined by the total expected cost of reaching the goal from s_0: $V^\pi(s) = \lim_{M \to \infty} \mathrm{E}\left[Z_M | \pi, s_0 = s \right]$. The optimal value $V^*(s) = \min_\pi V^\pi(s)$ can be computed by solving the Bellman equation:

$$V^*(s) = \begin{cases} 0 & , \text{if } s \in \mathcal{G} \\ \min\limits_{a \in \mathcal{A}} \left[c(s, a) + \sum\limits_{s' \in \mathcal{S}} T(s, a, s') V^*(s') \right] & , \text{otherwise.} \end{cases} \quad (1)$$

The VaR (*Value at Risk*) and CVaR (*Conditional-Value-at-Risk*) metrics are widely used for portfolio management of financial assets. VaR measures the worst expected loss within a given α confidence level, where $\alpha \in (0, 1)$. The CVaR with a confidence level $\alpha \in (0, 1)$ measures the expected value of the α% of the worst expected losses.

A CVaR SSP [4] is defined by the tuple $\mathcal{M}_{CVaR} = \langle \mathcal{M}, \alpha \rangle$ where $\mathcal{M}$ is an SSP and $\alpha \in (0, 1]$ is the confidence level. Let Π_H be the set of all history-dependent policies. The objective in CVaR SSPs is to find $\mu \in \Pi_H$ [4]:

$$\min_{\mu \in \Pi_H} CVaR_\alpha \left(\sum_{t=0}^{\infty} c(s_t, a_t) | s_0 = s, \mu \right), \quad (2)$$

where $\mu = \{\mu_0, \mu_1, ...\}$ is the policy sequence that depends on the history with actions $a_t = \mu_t(h_t)$ for $t \in \{0, 1, ...\}$.

A dynamic programming formulation for the CVaR SSP problem was proposed by Chow et al. [4] by defining the CVaR value function V over an augmented state space $\mathcal{S} \times Y$, where $Y = (0, 1]$ is a continuous confidence level. Among the algorithms that solve CVaR

SSPs are CVaRVILI and CVaRVIQ. CVaRVILI makes a discretization of Y by generating a set of interpolation points (atoms) and then interpolating the value function across these points. CVaRVIQ is inspired by the use of the distributional approach of Bellemare et al. [2]. The connection between the function $yCVaR_y$ and the quantile function (VaR_y) of the distribution of Z that is a result of the convexity and piecewise linear properties of $yCVaR_y$ function are used to make faster computations than CVaRVILI.

3 FORPECVAR ALGORITHM

Theorem 1 shows how the CVaR value of a policy π can be expressed in a forward approach, instead of a backup operator. In Theorem 1, $P^{X,\pi}(s)$ is the probability of reaching a goal state paying at most X when following policy π. Intuitively, Theorem 1 indicates that the CVaR value of a policy π for $\alpha = 1 - P^{X,\pi}(s)$ can be calculated by the difference between the mean value ($\mathbb{E}[Z|s_0 = s, \pi]$) and the expected value of the best cases with cost at most X divided by the probability of not reaching a goal state paying at most X. X plays the role of $VaR_{\alpha=1-P^{X,\pi}(s)}$, as it will divide the Z distribution into $P^{X,\pi}(s)$ best cases and $1 - P^{X,\pi}(s)$ worst cases.

Theorem 1. *Let the random variable $Z = \lim_{T\to\infty} \sum_{t=0}^{T} c_t$ be the accumulated cost and π be a proper policy. Let $\mathcal{X}^{\pi}(s) = \{X \in \mathbb{R} | \Pr(Z = X|s_0 = s, \pi) > 0\}$ be the set of accumulated cost with nonzero probability. For an SSP, $\mathcal{X}^{\pi}(s)$ is countable . For all $X \in \mathcal{X}^{\pi}(s)$, we define: $y(X) = 1 - P^{X,\pi}(s)$. The CVaR value of a policy π of the augmented state $\Big(s, y(X)\Big)$ can be computed by:*

$$V^{\pi}\big(s, y(X)\big) = \frac{\mathbb{E}[Z|s_0 = s, \pi] - \mathbb{E}\left[Z|Z \leq X, s_0 = s, \pi\right] P^{X,\pi}(s)}{1 - P^{X,\pi}(s)}.$$

The ForPECVaR algorithm makes use of Theorem 1 to compute CVaR values $V^{\pi}(s_0, \alpha)$ for a proper policy π and an initial state s_0 considering a target α. The ForPECVaR algorithm constructs a tree from the initial augmented state (s_0, α) and expands leaves until a goal state is reached. Leaves with the smallest accumulated cost are expanded first so that the minimum cost trajectory is founded first. Globally, the ForPECVaR algorithm keeps the expected value of the best cases with cost at most X, i.e., $\mathbb{E}\left[Z|Z \leq X, s_0 = s, \pi\right]$, and $P^{X,\pi}(s_0, s')$.

4 EXPERIMENTS

We compared CVaRVILI [4] and CVaRVIQ [8] in terms of execution time and quality of the solution. Both algorithms return the policy and the CVaR value function (***approximate value***) for all augmented states $(s, y) \in \mathcal{S} \times Y$. The quality of the policy is evaluated exactly with ForPECVaR, whose value is referred to as ***exact value***. With the experiments, we want to answer the following questions: (1) What are the differences between the CVaRVILI and CVaRVIQ algorithms in terms of the approximate value, exact value, and execution time?; and (2) What is the influence of CVaRVIQ parameters (number of atoms $|Y|$ and α_0) on the approximate and exact values? Are there some insights about how to choose these parameter values for a problem?

We used a desktop machine running with 6 processors at 2.90 GHz and 24 GB of memory DDR4. We executed the experiments in the Gridworld domain used in [4] and [8] (with grids of 5×5, 8×9, and 14×16), and the River domain [5] (with grids of 10×3, 16×6, and 30×10) . We set $\epsilon = 0.001$ as the residual error. The parameters values used in the experiments are $|Y| \in \{7, 13, 25\}$, where $|Y| = N(s), \forall s \in \mathcal{S}$, and $\alpha_0 \in \{10^{-3}, 10^{-2}, 10^{-1}\}$.

Approximate and exact values of CVaRVILI and CVaRVIQ. The results show that the difference between the approximate values obtained by CVaRVILI and CVaRVIQ was less than 10^{-6} in all the points, and the difference between the exact values of both algorithms was less than 0.1 in all points. Additionally, CVaRVIQ was up to two orders of magnitude faster than CVaRVILI. For these algorithms, the number of atoms is more significant than the value of α_0 in the execution time.

Values of CVaRVIQ varying $|Y|$ and α_0. The results show that as more atoms are used while varying $|Y|$ with a fixed α_0, the difference between the approximate and exact values decreases. However, we observed a limitation in points closer to α_0, where the distance between the approximate and exact values was greater compared to points closer to 1, even when a high number of atoms were used. In the other experiment, varying α_0 with a fixed $|Y|$, we found that using an α_0 value smaller than the α_{target} can result in better approximations, provided that a sufficient number of atoms are used. In summary, to achieve a good approximation for a problem with a target value of α_{target}, we need to first select an appropriate number of atoms and then choose an α_0 value that is smaller than α_{target}.

Execution time of ForPECVaR. For a fixed $|Y|$, the lower the α_0 value, the longer it takes to evaluate the policy because it is necessary to reach the goal state with a higher probability. When fixing α_0, we see that with more atoms, the execution time is longer, because with more atoms, the policy is better, and the policy will tend to take safer actions, which will take more time to reach the goal state. We also compared the ForPECVaR with a Monte Carlo simulation (MC) using the same amount of time spent to run the algorithms in one problem of each domain. The evaluation difference exceeded 0.45 on average over 5 MC runs, which suggests that MC can not get accurate evaluations of the performance of the policies considering the same time spent by ForPECVaR.

5 CONCLUSION

Given the existence of many algorithms with approximation to solve CVaR MDPs problems, it is important to have exact algorithms to evaluate them and the influence of their parameters. In this work, we have presented ForPECVaR, an exact algorithm to evaluate any CVaR policy with a forward approach. In addition to the CVaR value, ForPECVaR also calculates the exact VaR value of the policies. Our experimental evaluation has demonstrated that the approximate algorithms CVaRVILI and CVaRVIQ return similar policies and values, but the second has a better execution time. The exact evaluation of the CVaRVIQ policy shows a limitation of the algorithms analyzed in relation to the approximation of the values and policies closest to the minimum confidence level α. We also showed that the simple approach of MC can not get accurate evaluations of policies considering the same time used by ForPECVaR.

ACKNOWLEDGMENTS

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001 and the Center for Artificial Intelligence (C4AI-USP), with support by FAPESP (grant #2019/07665-4) and by the IBM Corporation.

REFERENCES

[1] Nicole Bäuerle and Jonathan Ott. 2011. Markov decision processes with average-value-at-risk criteria. *Mathematical Methods of Operations Research* 74, 3 (2011), 361–379.

[2] Marc G Bellemare, Will Dabney, and Rémi Munos. 2017. A distributional perspective on reinforcement learning. In *International Conference on Machine Learning*. PMLR, 449–458.

[3] Dimitri P Bertsekas and John N Tsitsiklis. 1991. An analysis of stochastic shortest path problems. *Mathematics of Operations Research* 16, 3 (Aug. 1991), 580–595.

[4] Yinlam Chow, Aviv Tamar, Shie Mannor, and Marco Pavone. 2015. Risk-Sensitive and Robust Decision-Making: a CVaR Optimization Approach. In *NIPS*. 1522–1530.

[5] Valdinei Freire and Karina Valdivia Delgado. 2017. GUBS: A Utility-Based Semantic for Goal-Directed Markov Decision Processes. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems (AAMAS '17)*. 741–749.

[6] Tobias Meggendorfer. 2022. Risk-Aware Stochastic Shortest Path. In *Thirty-Sixth AAAI Conference on Artificial Intelligence*. AAAI Press, 9858–9867.

[7] R.Tyrrell Rockafellar and Stanislav Uryasev. 2002. Conditional value-at-risk for general loss distributions. *Journal of Banking & Finance* 26, 7 (2002), 1443–1471.

[8] Silvestr Stanko and Karel Macek. 2019. Risk-averse Distributional Reinforcement Learning: A CVaR Optimization Approach.. In *IJCCI*. 412–423.