Quantum Algorithms for Reinforcement Learning with a Generative Model

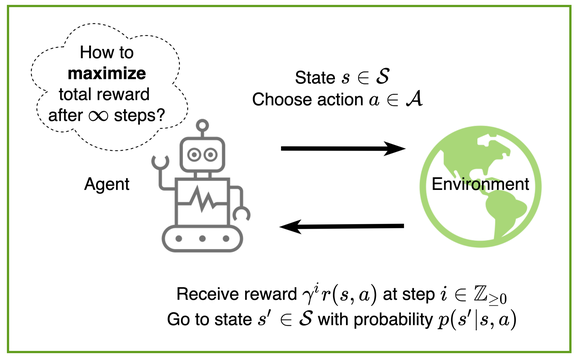
Daochen Wang[1], Aarthi Sundaram[2], Robin Kothari[2], Ashish Kapoor[2], Martin Roetteler[2]

[1]University of Maryland and [2]Microsoft Research

Objective

Rigorously quantify how reinforcement learning can be sped up by quantum computation.

Problem Setup



We consider a discounted MDP which is specified by a 5-tuple $(\mathcal{S}, \mathcal{A}, p, r, \gamma)$, where (i) $\mathcal{S}$ is the state space, (ii) $\mathcal{A}$ is the action space, (iii) $p(s'|s,a)$ are the transition probabilities, (iv) $r(s,a) \in [0,1]$ are the rewards, and (v) $\gamma \in [0,1)$ is the discount factor. Following classical convention, we assume that all parameters are known except p.

Goals: estimate the optimal Q-function (q^*), value function (v^*), and policy (π^*).

Quantum Generative Model

- A classical generative model G enables sampling from $p(\cdot|s,a)$ for any $(s,a) \in \mathcal{S} \times \mathcal{A}$ of our choice.
- We assume we have G as a classical circuit, which is justified whenever there exists a simulator for the environment; for example, if the environment is a video game or some other program.
- Then, it is a standard fact that we can efficiently convert G to a quantum circuit $\mathcal{G}$ that can sample from $p(\cdot|s,a)$ *in superposition*. We call $\mathcal{G}$ the **quantum generative model**.
- We design quantum algorithms that use $\mathcal{G}$ to achieve our goals; the number of times they call $\mathcal{G}$ is their quantum sample complexity.

Main Results

Quantum computing allows for speedups in terms of the parameters ϵ, $\Gamma \coloneqq (1-\gamma)^{-1}$, and $A \coloneqq |\mathcal{A}|$, but not $S \coloneqq |\mathcal{S}|$. For simplicity, all bounds are for maximum failure probability δ being constant (the dependence on δ is still logarithmic like for classical algorithms). The two quantum upper bounds that are optimal in Γ only hold for restricted $\epsilon \in O(1/\sqrt{\Gamma})$; otherwise the range of ϵ is unrestricted.

Goal: output an ϵ-accurate estimate of	Classical sample complexity	Quantum sample complexity[1]	
	Upper and lower bound	Upper bound	Lower bound
q^*	$\frac{SA\Gamma^3}{\epsilon^2}$	$\frac{SA\Gamma^{1.5}}{\epsilon}$	$\frac{SA\Gamma^{1.5}}{\epsilon}$
v^*, π^*	$\frac{SA\Gamma^3}{\epsilon^2}$	$\min\{\frac{SA\Gamma^{1.5}}{\epsilon}, \frac{S\sqrt{A}\Gamma^3}{\epsilon}\}$	$\frac{S\sqrt{A}\Gamma^{1.5}}{\epsilon}$

[1]This equals the quantum *time* complexity up to log factors assuming access to quantum random access memory (QRAM) which is the quantum analogue of classical RAM.

Upper Bounds

Main idea: apply quantum mean estimation [1] and maximum finding [2] to the algorithm in Ref. [3].

- Quantum mean estimation estimates $\mathbb{E}[X]$ to accuracy ϵ using $\tilde{O}(\sqrt{\mathrm{Var}[X]}/\epsilon)$ quantum samples of X. This can be thought of as a *quadratically* more efficient version of Chebyshev's inequality.
- Quantum maximum finding finds the maximum of a size-n list using $\tilde{O}(\sqrt{n})$ quantum queries to it. This is also quadratically more efficient than classical maximum finding which requires $\Omega(n)$ classical queries.

They can already speed up the red parts of the following basic version of value iteration which computes v^*:

$v \leftarrow \mathbf{0} \in \mathbb{R}^A$
for $\ell = 1, 2, \ldots, \tilde{O}(\frac{1}{1-\gamma})$ **do**
  for $s \in \mathcal{S}$ **do**
    $v(s) \leftarrow \max_{a \in \mathcal{A}}\{r(s,a) + \gamma \mathbb{E}[v(s') \mid s' \sim p(\cdot|s,a)]\}$
  end
end

However, this is highly sub-optimal, so we instead speed up the more sophisticated value iteration in Ref. [3].

Lower Bounds

We prove our lower bounds by considering hard MDP instances that are similar to those described in Ref. [4]. However, our argument is different. We reduce the computation of standard Boolean functions to the estimation of q^*, v^*, and π^* so that known lower bounds on the former computations imply our results.

Conclusion

To the best of our knowledge, ours is the first work to rigorously study quantum algorithms for solving MDPs. We show that quantum computers can offer quadratic speedups in terms of ϵ, Γ, and A in calculating q^*, v^*, and π^*. We show our algorithms are either optimal, or optimal assuming Γ or A is constant, for certain ranges of ϵ. Our work leaves open many interesting questions, including:

- Can we find quantum algorithms that are optimal in all parameters for an unrestricted range of ϵ?
- Is it possible to circumvent our quantum lower bounds by changing the output requirement?
- Our quantum algorithms are model-free. Can we develop model-based quantum algorithms?

References

[1] Ashley Montanaro. Quantum speedup of Monte Carlo methods. *Proceedings of the Royal Society A*, 2015.

[2] Christoph Dürr and Peter Høyer. A Quantum Algorithm for Finding the Minimum. *Manuscript*, 1996.

[3] Aaron Sidford, Mengdi Wang, Xian Wu, Lin Yang, and Yinyu Ye. Near-Optimal Time and Sample Complexities for Solving Markov Decision Processes with a Generative Model. *NeurIPS 31*, 2018.

[4] Mohammad Gheshlaghi Azar, Rémi Munos, and Hilbert J. Kappen. On the Sample Complexity of Reinforcement Learning with a Generative Model. *29th ICML*, 2012.

Acknowledgments

We especially thank Wen Sun for introducing us to reinforcement learning. We also thank Aaron Sidford, Mengdi Wang, and Xian Wu for helpful discussions. DW acknowledges funding by the Army Research Office (grant W911NF-20-1-0015) and NSF award DMR-1747426. Part of this work was performed while DW was an intern at Microsoft.

Contact Information

Daochen Wang
⟨wdaochen@gmail.com⟩
Aarthi Sundaram
⟨aarthi.sundaram@microsoft.com⟩