International Journal of Automation and Computing
www.ijac.net

18(3), June 2021, 410-421
DOI: 10.1007/s11633-021-1278-z

Optimal Policies for Quantum Markov Decision Processes

Ming-Sheng Ying[1,2,3] Yuan Feng[1] Sheng-Gang Ying[2]

[1] Centre for Quantum Software and Information, University of Technology Sydney, NSW 2007, Australia

[2] State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China

[3] Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China

Abstract: Markov decision process (MDP) offers a general framework for modelling sequential decision making where outcomes are random. In particular, it serves as a mathematical framework for reinforcement learning. This paper introduces an extension of MDP, namely quantum MDP (qMDP), that can serve as a mathematical model of decision making about quantum systems. We develop dynamic programming algorithms for policy evaluation and finding optimal policies for qMDPs in the case of finite-horizon. The results obtained in this paper provide some useful mathematical tools for reinforcement learning techniques applied to the quantum world.

Keywords: Quantum Markov decision processes, quantum machine learning, reinforcement learning, dynamic programming, decision making.

Citation: M. S. Ying, Y. Feng, S. G. Ying. Optimal policies for quantum Markov decision processes. *International Journal of Automation and Computing*, vol.18, no.3, pp.410–421, 2021. http://doi.org/10.1007/s11633-021-1278-z

1 Introduction

Markov decision process (MDP) offers a general framework for modelling sequential decision making where outcomes are random[1]. It stemmed from operations research and has been widely used in a broad range of areas, including manufacturing, economics, ecology, biology, automatic control and robotics. Since Kaelbling et al.[2] introduced MDPs, in particular partially observable Markov decision processes (POMDPs) into artificial intelligence (AI), they have been successfully applied in planning, scheduling, machine learning, to name just a few.

1.1 Quantum Markov decision processes

Recently, MDPs have been generalised into the quantum world in two slightly different ways:

1) The notion of quantum observable Markov decision process (QOMDP) was defined by Barry et al.[3] as a quantum generalisation of Kaelbling et al.'s POMDP[2]. The following two problems were studied there:

i) Policy existence problem for the infinite horizon: Given a QOMDP, a starting state, and a value V, whether there is a policy that achieves reward at least V?

ii) Goal-state reachability problem for the finite-horizon: Given a QOMDP, a starting state s, and a goal state s', whether there is a policy that can reach s' from s with probability 1?

The most interesting result in [3] is a computability separation between POMDPs and QOMDPs indicating that the goal-state reachability is decidable for POMDPs but undecidable for QOMDPs.

2) Another quantum generalisation of MDPs, called qMDP, was defined in [4]. A major difference between QOMDPs and qMDPs is that a policy in a QOMDP maps directly a (pure or mixed) quantum state to an action, whereas a policy in a qMDP maps the outcome of measurement on a quantum state to an action. It was proved that the goal-reachability problem for infinite-horizon qMDPs with probability 1 or $p < 1$ is EXP-hard or undecidable, respectively. The authors have employed quantum Markov chains (QMCs) as the semantic model in their research on static analysis of quantum programs[5, 6]. In particular, the termination problem of quantum programs can be reduced to reachability of QMCs. This observation lead the authors further to developing model checking techniques for quantum systems[7–9]. Essentially, the main results in [4] are extensions of the corresponding results of [9] to qMDPs.

1.2 Quantum machine learning

In the last five to ten years, a new research line has been rapidly emerging at the intersection of quantum physics and AI & machine learning[10, 11]. The interaction between these two areas is bidirectional:

1) Quantum physics helps to solve AI & machine learning problems via quantum computation.

Research Article
Manuscript received July 22, 2020; accepted January 13, 2021; published online March 20, 2021
Recommended by Associate Editor Jyh-Horng Chou
Colored figures are available in the online version at https://link.springer.com/journal/11633

2) AI & machine learning methodologies and techniques are employed to help solving problems in quantum physics.

For more detailed discussions about this area, the reader is referred to several excellent surveys[12–14].

Reinforcement learning is a basic machine learning paradigm, in which an agent learns behaviour through trial-and-error interactions with the dynamic environment[15, 16]. Several quantum reinforcement learning models have already been proposed either for enabling to apply reinforcement learning in the quantum world or for enhancing reinforcement learning by exploiting quantum advantage (see for example [17–20]). A question naturally arises here: How can the quantum Markov decision processes introduced in [3, 4] be used as a mathematical framework of quantum reinforcement learning?

1.3 Contributions of this paper

The main problem considered in [3, 4] is the reachability of quantum Markov decision processes (QOMDPs and qMDPs). However, a crucial step in classical reinforcement learning is to find an optimal behaviour of the agent that can usually be formulated as an optimal policy in MDPs. This paper solves the optimal policy problem for quantum Markov decision processes to provide a useful mathematical tool for decision making and reinforcement learning in the quantum world. In this paper, we focus on the case of finite horizon. The case of infinite horizon will be discussed in a forthcoming paper. We adopt a model that extends a qMDP defined in [4].

The paper is organised as follows: Quantum mechanics is briefly reviewed in Section 2 in a way that the AI community can easily understand. Several basic notions, including qMDP, policy and expected reward, are defined in Section 3. A backward recursion for the expected reward with a given policy is established, and an algorithm based on it for computing the expected reward is presented in Section 4. In Section 5, the Bellman principle of optimality is generalised to qMDPs and an algorithm for finding optimal policies for qMDPs is given. A key step in the algorithms presented in Sections 4 and 5 is the computation of quantum probabilities. For readability, we separate it from the other parts of the algorithms and solve it in Section 6. An illustrative example is shown in Section 7. The paper concludes with several remarks about further studies.

2 Preliminaries

For the convenience of the reader, we review the basics of quantum mechanics. In this paper, we only consider quantum systems of which the state spaces are finite-dimensional. So, their mathematical descriptions can be presented in the languages of vectors and matrices. We assume the reader is familiar with matrix algebra. All operations (e.g., addition, multiplication, scalar product) of vectors and matrices used in this paper are standard.

Quantum states. Following the convention in quantum theory, we use the Dirac notation to write

$$|\psi\rangle = (a_1, \cdots, a_n)^{\mathrm{T}}$$

for a column vector, i.e., an element in the n-dimensional complex vector space $\mathbf{C}^n$, where $\mathbf{C}$ is the field of complex numbers, and T stands for transpose. If the components of the vector satisfies the normalisation condition:

$$\sum_{i=1}^{n} |a_i|^2 = 1$$

then $|\psi\rangle$ is called a unit vector. The dual of $|\psi\rangle$ is the row vector $\langle\psi| = (a_1, \cdots, a_n)$. According to the basic postulates of quantum mechanics, a pure state of an n-level quantum system can be represented by a unit vector $|\psi\rangle \in \mathbf{C}^n$. For example, the state of a qubit (quantum bit) is a 2-dimensional vector:

$$(\alpha, \beta)^{\mathrm{T}} = \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \alpha|0\rangle + \beta|0\rangle$$

with $|\alpha|^2 + |\beta|^2 = 1$, where $|0\rangle = (1, 0)^{\mathrm{T}}, |1\rangle = (0, 1)^{\mathrm{T}}$ is a basis of the 2-dimensional space $\mathbf{C}^2$. Two example qubit states are

$$|\pm\rangle = \frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle).$$

More generally, a mixed state of an n-level system is described by an $n \times n$ positive semi-definite matrix $\rho = (\rho_{ij})$ with trace:

$$\mathrm{tr}(\rho) = \sum_i \rho_{ii} = 1$$

called a density matrix in $\mathbf{C}^n$. It turns out that each density operator ρ can be written in the form of

$$\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|$$

where $\{|\psi_i\rangle\}$ is a family of pure states, and $\{p_i\}$ is a probability distribution. So, mixed state ρ can be interpreted as follows: The system is in state $|\psi_i\rangle$ with probability p_i. For example, if a qubit is in state $|0\rangle$ with probability $\frac{2}{3}$ and in state $|1\rangle$ with probability $\frac{1}{3}$, then it can be depicted by the density matrix:

$$\rho_0 = \frac{2}{3}|0\rangle\langle 0| + \frac{1}{3}|-\rangle\langle -| = \frac{1}{6}\begin{pmatrix} 5 & -1 \\ -1 & 1 \end{pmatrix}.$$

Dynamics of quantum systems. For any matrix $A = (a_{ij})$, we write $A^\dagger$ for the transpose and conjugate of

A, i.e., $A^\dagger = (b_{ij})$ with $b_{ij} = a^*_{ji}$ for all i, j. An $n \times n$ complex matrix U is called a unitary matrix if

$$U^\dagger U = I_{n\times n}$$

where I is the identity matrix. If the states of a closed quantum system at times t and t' are $|\psi\rangle$ and $|\psi'\rangle$, respectively, then they are related to each other by a unitary matrix U which depends only on the times t and t':

$$|\psi'\rangle = U|\psi\rangle.$$

If the system is in mixed state ρ, ρ' at time t, t', respectively, then,

$$\rho' = U\rho U^\dagger.$$

For example, the NOT gate and Hadamard gate on a qubit are respectively described by unitary matrices:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},\ H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

and the state ρ_0 is transferred by H into

$$H\rho_0 H^\dagger = \frac{1}{3} \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix}.$$

More generally, the dynamics of an open quantum system is described by a super-operator. The notion of super-operator can be introduced in several different (but equivalent) ways. Here, we choose to use the Kraus operator-sum representation, which is convenient for computation. A super-operator $\mathcal{E}$ transforms a density matrix to another and is defined by a family of $n \times n$ matrices $\{E_i\}$:

$$\mathcal{E}(\rho) = \sum_i E_i \rho E_i^\dagger$$

for each density matrix ρ, where it is required that

$$\sum_i E^\dagger E_i = I_{n\times n}.$$

It is obvious that $\mathcal{E}$ degenerates to a unitary matrix whenever $\{E_i\}$ is a singleton. For example, the bit flip action transfers the state of a qubit from $|0\rangle$ to $|1\rangle$ and vice versa, with probability $1 - p$, $0 \le p \le 1$. It is described by the super-operator:

$$\mathcal{E}(\rho) = E_0 \rho E_0 + E_1 \rho E_1$$

where

$$E_0 = \sqrt{p} I,\ E_1 = \sqrt{1-p} X$$

and I, X are the 2×2 unit matrix and the NOT gate, respectively. For example, state ρ_0 is transformed by $\mathcal{E}$ to

$$\mathcal{E}(\rho) = \begin{pmatrix} \frac{1}{6} + \frac{2p}{3} & -\frac{1}{6} \\ -\frac{1}{6} & \frac{5}{6} - \frac{2p}{3} \end{pmatrix}.$$

Quantum measurements. To acquire information about a quantum system, a measurement must be performed on it. A quantum measurement on an n-level system is described by a collection $M = \{M_m\}$ of $n \times n$ complex matrices satisfying the normalisation condition:

$$\sum_m M_m^\dagger M_m = I_{n\times n}$$

where the indices m stand for the measurement outcomes. We write $O(M) = \{m\}$ for the set of all possible outcomes of M. If the state of a quantum system is $|\psi\rangle$ immediately before the measurement, then the probability that result m occurs is

$$p(m) = \langle\psi|M_m^\dagger M_m|\psi\rangle$$

and the state of the system after the measurement is

$$\frac{M_m|\psi\rangle}{\sqrt{p(m)}}.$$

If the state of a quantum system was ρ before measurement, then the probability that result m occurs is

$$p(m) = \mathrm{tr}(M_m \rho M_m^\dagger) \tag{1}$$

and the state after the measurement is

$$\frac{M_m \rho M_m^\dagger}{p(m)}. \tag{2}$$

For example, the measurement on a qubit in the computational basis $\{|0\rangle, |1\rangle\}$ is $M = \{M_0, M_1\}$, where

$$M_0 = |0\rangle\langle 0| = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix},\ M_1 = |1\rangle\langle 1| = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}.$$

If we perform M on a qubit in state ρ_0, then the probability that we get outcome 0 is

$$p(0) = \mathrm{tr}(M_0 \rho_0) = \mathrm{tr} \begin{pmatrix} \frac{5}{6} & 0 \\ 0 & 0 \end{pmatrix} = \frac{5}{6}$$

and the probability of outcome 1 is $p(1) = \frac{1}{6}$. In the case that the outcome is 0, the qubit will be in state $|0\rangle$ after the measurement, and in the case that the outcome is 1, it will be in state $|1\rangle$.

Composite quantum systems. In the example presented in Section 7, we will need the notion of a tensor product of vector spaces. For each $1 \leq i \leq n$, let $\mathcal{H}_i$ be a vector space with $\{|\psi_{ij}\rangle\}$ as an orthonormal basis. Then the tensor product $\mathcal{H}_1 \otimes \cdots \otimes \mathcal{H}_n$ is the vector space with $\{|\psi_{1j_1}, \cdots, \psi_{nj_n}\rangle\}$ as an orthonormal basis. For example, the state space of two-qubits is $\mathbf{C}^2 \otimes \mathbf{C}^2$. A two-qubit system can be in a separable state

$$|\psi\rangle = |\psi_1\rangle \otimes |\psi_2\rangle = |\psi_1\rangle, |\psi_2\rangle$$

where $|\psi_1\rangle, |\psi_2\rangle$ are one-qubit states, e.g., $|0,0\rangle, |0,1\rangle$, $|1,+\rangle, |+,-\rangle$. It can also be in an entangled state that cannot be written as the product of two one-qubit states, like the EPR (Einstein-Podolsky-Rosen) pair or Bell state:

$$|\beta_{00}\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle).$$

3 Basic definitions

Recall from [1] that an MDP consists of decision epochs, states, actions, transition probabilities and rewards. The decision epochs are the points of time where decisions are made. In this paper, we only consider the case of finite horizon – the set $\mathcal{T}$ of decision epochs is finite. We write $\mathcal{S}$ for the set of possible states of the system and $\mathcal{A}$ for the set of allowable actions. At each decision epoch, the system occupies a state $s \in \mathcal{S}$, and the decision maker take an action a chosen from $\mathcal{A}$. As a result of taking action a in state s at decision epoch t, the decision maker receives a reward $r_t(s,a)$, and the system evolves as follows: At the next decision epoch, the system is in state s' with probability $p_t(s'|s,a)$.

A qMDP is a quantum generalisation of MDP where the dynamics of and the observation on the system are governed by the laws of quantum mechanics. Formally, we have:

Definition 1. *A* qMDP is a 7-tuple

$$\mathcal{P} = (\mathcal{T}, \mathcal{H}, \rho, \mathcal{A}, \{\mathcal{E}_t(\cdot|a) : t \in \mathcal{T}, a \in \mathcal{A}\}, \mathcal{M}, \{r_t : t \in \mathcal{T}\})$$

where:

1) $\mathcal{T} = \{1, 2, \cdots, N\}$ is the set of decision epochs.

2) $\mathcal{H} = \mathbf{C}^n$ is the state space of an n-level quantum system.

3) ρ is a density matrix in $\mathcal{H}$, called the starting state.

4) $\mathcal{A}$ is a set of action names.

5) For each $t \in \mathcal{T}$ and $a \in \mathcal{A}$, $\mathcal{E}_t(\cdot|a)$ is a super-operators in $\mathcal{H}$.

6) $\mathcal{M}$ is a set of quantum measurements in $\mathcal{H}$. We write:

$$\mathcal{O} = \bigcup_{M \in \mathcal{M}} [\{M\} \times O(M)].$$

7) For each $1 \leq t \leq N-1$, $r_t : \mathcal{O} \times \mathcal{A} \to \mathbf{R}$ (real numbers) is the reward function at decision epoch t, and $r_N : \mathcal{O} \to \mathbf{R}$ is the reward function at the final decision epoch N.

An MDP is a decision maker together with a classical (but stochastic) system on which the decision maker can take actions. In contrast, a qMDP consists of a decision maker and a quantum system of which the state space is $\mathcal{H} = \mathbf{C}^n$. The state of this quantum system is described by a density matrix. $\mathcal{A}$ and $\mathcal{M}$ are the sets of actions and measurements, respectively, allowable to perform on the system. For each decision epoch $t \in \mathcal{T}$, the decision maker acquires information about the system through performing a chosen measurement $M \in \mathcal{M}$. It is possible that different outcomes occur with certain probabilities. Each $(M,m) \in \mathcal{O}$ is called an observation, meaning that measurement M is performed and the outcome is m. For each action $a \in \mathcal{A}$, the super-operator $\mathcal{E}_t(\cdot|a)$ models the evolution of the system if a is taken on it between t and the next epoch. So, if the system is in state σ before action a, then it will be in state $\mathcal{E}_t(\sigma|a)$ after action a. Obviously, $\mathcal{E}_t(\cdot|a)$ can be seen as the quantum counterpart of the matrix

$$\left\{p_t(s'|s,a)\right\}_{s,s' \in \mathcal{S}}$$

of transition probabilities in a MDP. For each $(M,m) \in \mathcal{O}$ and $a \in \mathcal{A}$, $r_t(M,m,a)$ is the reward that the decision maker gains by taking action a at decision epoch t when the outcome of measurement M is m. Note that in a MDP, the reward depends on the state of the system. However, in a qMDP, the reward depends on the observation (M,m) about the system rather than directly on the state of the system, because usually the state of a quantum system cannot be fully known. Since at the final epoch $t = N$, no action will be taken, the domain of the reward function r_N is $\mathcal{O}$ but not $\mathcal{O} \times \mathcal{A}$.

Now we start to examine the behaviour of a qMDP by introducing the following:

Definition 2. *Let* $1 \leq t \leq N$. Then a sequence

$$h_t = (M_1, m_1, a_1, \cdots, M_{t-1}, m_{t-1}, a_{t-1}, M_t, m_t)$$

is called a history of t epochs if $(M_1, m_1), \cdots, (M_{t-1}, m_{t-1}), (M_t, m_t) \in \mathcal{O}$ and $a_1, \cdots, a_{t-1} \in \mathcal{A}$.

History h_t records the activities of the decision maker: For each $j \leq t$, she/he performed measurement M_j on the system, got outcome m_j, and then took action a_j on it. It is assumed that measurement M_j happened before action a_j. If a_j was taken before M_j, then the result would be different because a measurement usually changes the state of a quantum system. We write $tail(h_t) = (M_t, m_t)$. The set of histories of t epochs is denoted H_t. Obviously, if $h_t \in H_t$, $a_t, a_{t+1}, \cdots, a_{t+(k-1)} \in \mathcal{A}$ and (M_{t+1}, m_{t+1}), $(M_{t+2}, m_{t+2}), \cdots, (M_{t+k}, m_{t+k}) \in \mathcal{O}$, then

$$(h_t, a_t, M_{t+1}, m_{t+1}, a_{t+1}, M_{t+2}, m_{t+2}, ..., a_{t+(k-1)}, M_{t+k}, m_{t+k}) \in H_{t+k}$$

for $1 \leq k \leq N-t$.

A policy specifies the rule to be used by the decision maker to choose the measurements and actions performed at all decision epochs. For any nonempty set X, we write $\mathcal{D}(X)$ for the set of probability distributions over X.

Definition 3. *A* randomised history-dependent policy is a sequence $\pi = (\alpha_0, \beta_1, \alpha_1, \cdots, \beta_{N-1}, \alpha_{N-1})$, where:

1) $\alpha_0 \in \mathcal{D}(\mathcal{M})$.

2) $\alpha_t : H_t \times \mathcal{A} \to \mathcal{D}(\mathcal{M})$ for $t = 1, \cdots, N-1$.

3) $\beta_t : H_t \to \mathcal{D}(\mathcal{A})$ for $t = 1, \cdots, N-1$.

For each $M \in \mathcal{M}$, $\alpha_0(M)$ is the probability that M is chosen at the beginning of the decision process. For each $1 \leq t \leq N-1$, $h_t \in H_t$, $a \in \mathcal{A}$ and $M \in \mathcal{M}$, $\beta_t(h_t)(a)$ is the probability that action a is chosen to take between decision epoch t and $t+1$ given history h_t, and $\alpha_t(h_t, a)(M)$ is the probability that measurement M is chosen to perform at epoch $t+1$ given history h_t and that action a was taken between epoch t and $t+1$. In particular, π is a deterministic (history-dependent) policy if α_0, $\alpha_t(h_t, a)$ and $\beta_t(h_t)$ are all single-point distributions; that is, $\alpha_0 \in \mathcal{M}$, and

$$\alpha_t : H_t \times \mathcal{A} \to \mathcal{M},\ \beta_t : H_t \to \mathcal{A}$$

for $t = 1, \cdots, N-1$.

Now let π be a randomised history-dependent policy, $1 \leq t \leq N$ and

$$h_t = (M_1, m_1, a_1, \cdots, M_{t-1}, m_{t-1}, a_{t-1}, M_t, m_t) \in H_t.$$

Then repeated applications of (1) and (2) yield the probability of h_t under π:

$$p^{\pi}(h_t) = \alpha_0(M_1) \prod_{j=1}^{t-1} [p_j \times \beta_j(h_j)(a_j) \times \alpha_j(h_j, a_j)(M_{j+1})] \times p_t$$

where $h_j = (M_1, m_1, a_1, \cdots, M_{j-1}, m_{j-1}, a_{j-1}, M_j, m_j)$ for $1 \leq j \leq t$, and

$$\begin{cases} p_1 = \mathrm{tr}\left(M_{1m_1} \rho M_{1m_1}^{\dagger}\right) \\ p_j = \mathrm{tr}\left(M_{jm_j} \mathcal{E}_{j-1}(\rho_{j-1}|a_{j-1}) M_{jm_j}^{\dagger}\right),\ 1 < j \leq N \end{cases} \tag{3}$$

$$\begin{cases} \rho_1 = M_{1m_1} \rho M_{1m_1}^{\dagger} / p_1 \\ \rho_j = M_{jm_j} \mathcal{E}_{j-1}(\rho_{j-1}|a_{j-1}) M_{jm_j}^{\dagger} / p_j,\ 1 < j \leq N. \end{cases} \tag{4}$$

Intuitively, the system starts in state ρ. At the initial epoch, measurement M_1 is chosen by policy π with probability $\alpha_0(M_1)$ to perform on the system, outcome m_1 is obtained with probability p_1, and the state of the system is changed to ρ_1. Then action a_1 is chosen by π with probability $\beta_1(h_1)(a_1)$, and the system is transformed into state $\mathcal{E}_1(\rho_1|a_1)$. In general, at epoch j, measurement M_j is chosen with probability $\alpha_{j-1}(h_{j-1}, a_{j-1})(M_j)$ to perform on state $\mathcal{E}_{j-1}(\rho_{j-1}|a_{j-1})$, outcome m_j occurs with probability p_j, and the state of the system becomes ρ_j. Furthermore, action a_j is chosen with probability $\beta_j(h_j)(a_j)$ and it transforms the system into state $\mathcal{E}_j(\rho_j|a_j)$.

The following lemma gives a more compact representation of probability function $p^{\pi}(\cdot)$.

Lemma 1.

$$p^{\pi}(h_t) = \alpha_0(M_1) \prod_{j=1}^{t-1} [\beta_j(h_j)(a_j) \times \alpha_j(h_j, a_j)(M_{j+1})] \times \mathrm{tr}(\sigma_t)$$

where

$$\begin{cases} \sigma_1 = M_{1m_1} \rho M_{1m_1}^{\dagger} \\ \sigma_j = M_{jm_j} \mathcal{E}_{j-1}(\sigma_{j-1}|a_{j-1}) M_{jm_j}^{\dagger} (1 < j \leq n). \end{cases} \tag{5}$$

Proof. By a routine calculation. □

Finally, we can define the reward received by the decision maker in a qMDP. For each randomised history-dependent policy π, if π is used in the decision process, then the expected total reward over the decision making horizon is

$$v_N^{\pi} = \sum_{h_N \in H_N} p^{\pi}(h_N) \times r(h_N) \tag{6}$$

where $r(h_N)$ is the reward received along history h_N, i.e.,

$$r(h_N) = \sum_{t=1}^{N-1} r_t(M_t, m_t, a_t) + r_N(M_N, m_N)$$

if $h_N = (M_1, m_1, a_1, \cdots, M_{N-1}, m_{N-1}, a_{N-1}, M_N, m_N)$.

4 Policy evaluation

As in the case of MDPs, a direct computation of the reward in a qMDP based on defining emuation (6) is very inefficient. In this section, we establish a backward recursion for the reward function so that dynamic programming can be used in policy evaluation for qMDPs. To this end, we first introduce a conditional probability function. Let π be a randomised history-dependent policy, $1 \leq t \leq N$ and

$$h_t = (M_1, m_1, a_1, \cdots, M_{t-1}, m_{t-1}, a_{t-1}, M_t, m_t) \in H_t$$
$$f_t = (a_t, M_{t+1}, m_{t+1}, \cdots, a_{N-1}, M_N, m_N) \in (\mathcal{A} \times \mathcal{O})^{N-t}.$$

Clearly, the concatenation (h_t, f_t) of h_t and f_t is in H_N. By repeated applications of (1) and (2), we obtain the conditional probability of f_t under π on h_t:

$$p^{\pi}(f_t|h_t) = \prod_{j=t}^{N-1} [\beta_j(h_j)(a_j) \times \alpha_j(h_j, a_j)(M_{j+1}) \times p_{j+1}] \tag{7}$$

where

$$h_j = (h_t, a_t, M_{t+1}, m_{t+1}, \cdots, a_{j-1}, M_j, m_j)$$

and p_j's are defined by (3). Similar to Lemma 1, we have:

Lemma 2.

$$p^\pi(f_t|h_t) = \prod_{j=t}^{N-1} [\beta_j(h_j)(a_j) \times \alpha_j(h_j, a_j)(M_{j+1})] \times \frac{\mathrm{tr}(\sigma_N)}{\mathrm{tr}(\sigma_t)} \tag{8}$$

where σ_j's are defined by (5).

Proof. By a routine calculation. □

Using the conditional probability function $p^\pi(\cdot|h_t)$, we can compute the expected reward in the tail of a decision process. More precisely, for each randomised history-dependent policy π, function

$$u_t^\pi : H_t \to R$$

is defined to be the expected total reward obtained by using policy π at decision epochs $t, t+1, \cdots, N$; i.e., for every $h_t \in H_t$,

$$u_t^\pi(h_t) = \sum_{f_t \in (\mathcal{A}\times\mathcal{O})^{N-t}} p^\pi(f_t|h_t) \times r(f_t) \tag{9}$$

where

$$r(f_t) = \sum_{j=t}^{N-1} r_j(M_j, m_j, a_j) + r_N(M_N, m_N).$$

Theorem 1 presents a backward recursion that shows how to compute the conditional reward u_t^π at decision epoch t from the conditional reward u_{t+1}^π at the next epoch $t+1$.

Theorem 1. (Backward Recursion) For each $1 \le t \le N-1$, we have:

$$u_t^\pi(h_t) = \sum_{a_t \in \mathcal{A}} \sum_{M_{t+1} \in \mathcal{M}} \beta_t(h_t)(a_t) \times \alpha_t(h_t, a_t)(M_{t+1}) \times \Big[r_t(M_t, m_t, a_t) + \sum p_{t+1} \times u_{t+1}^\pi(h_t, a_t, M_{t+1}, m_{t+1}) \Big] \tag{10}$$

where the third $\sum$ is over $m_{t+1} \in O(M_{t+1})$.

Proof. Straightforward. □

The aim of policy evaluation is to compute the total reward v_N^π. The following lemma gives a representation of v_N^π in terms of the conditional reward u_1^π at the first decision epoch.

Lemma 3.

$$v_N^\pi = \sum_{(M_1, m_1) \in \mathcal{O}} \alpha_0(M_1) \times \mathrm{tr}(M_{1m_1} \rho M_{1m_1}^\dagger) \times u_1^\pi(M_1, m_1) \tag{11}$$

Proof. Routine. □

Combining Theorem 1 and Lemma 3 enables us to develop a dynamic programming algorithm for evaluating v_N^π; see Algorithm 1. Note that there is an essential difficulty in step 2 of this policy evaluation algorithm, namely the computation of quantum probabilities p_{t+1}. The same difficulty arises in the next section where the optimal policies for qMDPs are considered. So, this problem will be carefully addressed in Section 6.

Algorithm 1. Policy evaluation

1) Set $t = N$ and $u_N^\pi(h_N) = r_N(M_N, m_N)$ for $h_N \in H_N$ with $tail(h_N) = (M_N, m_N)$.

2) Substitute $t-1$ for t. Compute $u_t^\pi(h_t)$ for $h_t \in H_t$ with $tail(h_t) = (M_t, m_t)$ using (10).

3) If $t = 1$, go to step 4. Otherwise, return to Step 2).

4) Compute v_N^π by (11).

Note that for each t, the computation of p_t according to Theorem 3 takes time $O(tn^6)$ where n is the dimension of the Hilbert space. Furthermore, as $|H_t| = |\mathcal{O}|^t|\mathcal{A}|^{t-1}$, and for each $h_t \in H_t$, the computation of $u_t^\pi(h_t)$ in (10) requires $|\mathcal{O}| \times |\mathcal{A}|$ multiplications each of which needs to compute $p_{t+1} = p_{t+1}(h_{t+1})$, the total complexity of Algorithm 1 is $O(|\mathcal{O}|^{N+1}|\mathcal{A}|^N n^6)$.

5 Optimality of policies

Now we turn to consider how to compute optimal policies. The optimal expected total reward over the decision making horizon is defined by

$$v_N^* = \sup_\pi v_N^\pi.$$

For any $1 \le t \le N$ and $h_t \in H_t$, $u_t^*(h_t)$ is defined to be the optimal expected total reward from decision epoch t onward when the history up to time t is h_t, i.e.,

$$u_t^*(h_t) = \sup_\pi u_t^\pi(h_t)$$

where π traverses over all randomised history-dependent policies. Similar to Lemma 3, Lemma 4 shows that the optimal total reward v_N^* can be represented in terms of the optimal reward u_1^π at the first decision epoch.

Lemma 4.

$$v_N^* = \sup_{M_1 \in \mathcal{M}} \left\{ \sum_{m_1 \in O(M_1)} \mathrm{tr}(M_{1m_1} \rho M_{1m_1}^\dagger) \times u_1^*(M_1, m_1) \right\} \tag{12}$$

Proof. By a routine calculation. □

Lemma 4 provides a method for computing the optimal total reward v_N^* through computing the optimal reward u_1^* at the first decision epoch, which can be computed by backward recursion based on the following quantum generalisation of the Bellman optimality equa-

tions:

$$[b]u_t(h_t) = \sup_{a_t\in\mathcal{A}} \sup_{M_{t+1}\in\mathcal{M}} \left[r_t(M_t, m_t, a_t) + \sum p_{t+1} \times u_{t+1}(h_t, a_t, M_{t+1}, m_{t+1})\right] \tag{13}$$

for $t = 1, \cdots, N-1$, where the summation is over $m_{t+1} \in O(M_{t+1})$, and p_{t+1} is given by (3), and

$$u_N(h_N) = r_N(M_N, m_N) \tag{14}$$

if

$$h_N = (M_1, m_1, a_1, \cdots, M_{N-1}, m_{N-1}, a_{N-1}, M_N, m_N).$$

Theorem 2 shows that the optimal expected reward $u_t^*(h_t)$ at decision epoch t can be computed by solving the optimal equations.

Theorem 2. (The principle of optimality) Let $u_t : H_t \to R(t = 1, \cdots, N)$ be a solution of the optimality equations (13) and (14). Then,

$$u_t(h_t) = u_t^*(h_t)$$

for all $t = 1, \cdots, N$ and $h_t \in H_t$.

Proof. First, we show that $u_t^*(h_t) \le u_t(h_t)$ by backward induction on t. By definition, it holds for $t = N$. Now assume that it holds for $t+1$. Then using Lemma 4.3.1 in [21] and Theorem 1, we obtain:

$$\begin{aligned}
&u_t^*(h_t) = \sup_\pi u_t^\pi(h_t) = \sup_\pi \sum_{a_t} \sum_{M_{t+1}} \beta_t^\pi \\
&(h_t)(a_t) \times \alpha_t^\pi(h_t, a_t)(M_{t+1}) \times \\
&[r_t(M_t, m_t, a_t) + \sum_{m_{t+1}} p_{t+1} \times \\
&u_{t+1}^\pi(h_t, a_t, M_{t+1}, m_{t+1})] \le \\
&\sup_\pi \sup_{a_t} \sup_{M_{t+1}} [r_t(M_t, m_t, a_t) + \sum_{m_{t+1}} p_{t+1} \times \\
&u_{t+1}^\pi(h_t, a_t, M_{t+1}, m_{t+1})] = \\
&\sup_{a_t} \sup_{M_{t+1}} [r_t(M_t, m_t, a_t) + \sum_{m_{t+1}} p_{t+1} \times \\
&\sup_\pi u_{t+1}^\pi(h_t, a_t, M_{t+1}, m_{t+1})] = \\
&\sup_{a_t} \sup_{M_{t+1}} [r_t(M_t, m_t, a_t) + \sum_{m_{t+1}} p_{t+1} \times \\
&u_{t+1}^*(h_t, a_t, M_{t+1}, m_{t+1})] \le \\
&\sup_{a_t} \sup_{M_{t+1}} [r_t(M_t, m_t, a_t) + \sum_{m_{t+1}} p_{t+1} \times \\
&u_{t+1}(h_t, a_t, M_{t+1}, m_{t+1})] = u_t(h_t).
\end{aligned}$$

Note that the last inequality comes from the induction hypothesis for $t+1$.

Secondly, we show that $u_t(h_t) \le u_t^*(h_t)$. For given t and h_t, and for any $\epsilon > 0$, by the definition of u_t, $u_{t+1}, \cdots, u_N$, we have

$$\begin{aligned}
&\exists a_t^*, M_{t+1}^* \text{ s.t. } u_t(h_t) - \epsilon \le r_t(M_t, m_t, a_t^*) + \\
&\sum_{m_{t+1}} p_{t+1} \times u_{t+1}(h_t, a_t^*, M_{t+1}^*, m_{t+1}) \\
&\forall m_{t+1}, \exists a_{t+1}^*, M_{t+2}^* \\
&\text{s.t. } u_{t+1}(h_{t+1}) - \epsilon \le r_{t+1}(M_{t+1}, m_{t+1}, a_{t+1}^*) + \\
&\sum_{m_{t+2}} p_{t+2} \cdot u_{t+2}(h_{t+1}, a_{t+1}^*, M_{t+2}^*, m_{t+2}) \\
&\forall m_{N-1}, \exists a_{N-1}^*, M_N^* \\
&\text{s.t. } u_{N-1}(h_{N-1}) - \epsilon \le r_{N-1}(M_{N-1}, m_{N-1}, a_{N-1}^*) + \\
&\sum_{m_N} p_N \cdot u_N(h_{N-1}, a_{N-1}^*, M_N^*, m_N).
\end{aligned}$$

Here,

$$h_j = (h_t, a_t^*, M_{t+1}^*, m_{t+1}, \cdots, a_{j-1}^*, M_j^*, m_j)$$

for $t < j \le n$. We choose a deterministic policy

$$\pi^* = (\alpha_0, \beta_1, \alpha_1, \cdots, \beta_{N-1}, \alpha_{N-1})$$

such that $\alpha_j(M_j, m_j) = M_{j+1}^*$ and $\beta_j(M_j, m_j) = a_{j+1}^*$. Then, we obtain:

$$\begin{aligned}
&u_t(h_t) \le r_t(M_t, m_t, a_t^*) + \sum_{m_{t+1}} p_{t+1} \times \\
&u_{t+1}(h_t, a_t^*, M_{t+1}^*, m_{t+1}) + \epsilon \le \\
&r_t(M_t, m_t, a_t^*) + \sum_{m_{t+1}} p_{t+1} \times [r_{t+1}(M_{t+1}^*, m_{t+1}, a_{t+1}^*) + \\
&\sum_{m_{t+2}} p_{t+2} \times u_{t+2}(h_{t+1}, a_{t+1}^*, M_{t+2}^*, m_{t+2}) + \epsilon] + \epsilon = \\
&r_t(M_t, m_t, a_t^*) + \sum_{m_{t+1}} p_{t+1} \times \\
&r_{t+1}(M_{t+1}^*, m_{t+1}, a_{t+1}^*) + \sum_{m_{t+1}, m_{t+2}} p_{t+1} \times \\
&p_{t+2} \times u_{t+2}(h_{t+1}, a_{t+1}^*, M_{t+2}^*, m_{t+2}) + 2\epsilon \le \\
&Q + \sum_{m_{t+1}, m_{t+2}, \cdots, m_N} p_{t+1} \times p_{N+2} \times \cdots \times p_N \times \\
&u_N(h_{N-1}, a_{N-1}^*, M_N^*, m_N) + (N-t)\epsilon = \\
&Q + \sum_{m_{t+1}, m_{t+2}, \cdots, m_N} p_{t+1} \times \\
&p_{N+2} \times \cdots \times p_N \times r_N(M_N^*, m_N) + (N-t)\epsilon \overset{*}{=} \\
&\sum_{m_{t+1}, m_{t+2}, \cdots, m_{N-1}} p_{t+1} \times \\
&p_{t+2} \times \cdots \times p_N \times [r_t(M_t, m_t, a_t^*) + r_{t+1}(M_{t+1}^*, m_{t+1}, a_{t+1}^*) \\
&+ \cdots + r_{N-1}(M_{N-1}^*, m_{N-1}, a_{N-1}^*) + r_N(M_N^*, m_N^*)] = \\
&u_t^{\pi^*}(h_t) + (N-t)\epsilon \le \\
&u_t^*(h_t) + (N-t)\epsilon
\end{aligned}$$

where

$$Q = r_t(M_t, m_t, a_t^*) + \sum_{m_{t+1}} p_{t+1} \times r_{t+1}(M_{t+1}^*, m_{t+1}, a_{t+1}^*) +$$
$$\sum_{m_{t+1}, m_{t+1}} p_{t+1} \times p_{t+2} \times r_{t+2}(M_{t+2}^*, m_{t+2}, a_{t+2}^*) + \cdots +$$
$$\sum_{m_{t+1}, m_{t+2}, \ldots, m_{N-1}} p_{t+1} \times p_{t+2} \times \cdots \times p_{N-1} \cdot$$
$$r_{N-1}(M_{N-1}^*, m_{N-1}, a_{N-1}^*).$$

Note that the equality " $\stackrel{*}{=}$" follows from that

$$\sum_{m_{t+1}, m_{t+2}, \ldots, m_N} p_{t+1} \times p_{t+2} \times \cdots \times p_N \times r_{t+j}(M_{t+j}^*, m_{t+j}, a_{t+j}^*) =$$
$$\sum_{m_{t+1}, m_{t+2}, \ldots, m_{t+j}} p_{t+1} \times p_{t+2} \times \cdots \times p_{t+j} \times r_{t+j}(M_{t+j}^*, m_{t+j}, a_{t+j}^*)$$

for $j = 1, \cdots, N-t-1$, and similar equalities for r_t and r_N. Finally, arbitrariness of ϵ leads to $u_t(h_t) \leq u_t^*(h_t)$. □

It can be seen from the above proof that for any $\epsilon > 0$, we can find a deterministic policy

$$\pi^\epsilon = (\alpha_0^\epsilon, \beta_1^\epsilon, \alpha_1^\epsilon, \cdots, \beta_{N-1}^\epsilon, \alpha_{N-1}^\epsilon)$$

such that

$$r_t(M_t, m_t, \beta_t^\epsilon(M_t, m_t)) + \sum_{m_{t+1}} p_{t+1} \times$$
$$u_{t+1}^*(h_t, \beta_t^\epsilon(M_t, m_t), \alpha_t^\epsilon(M_t, m_t), m_{t+1}) + \frac{\epsilon}{N-1} \geq u_t^*(h_t)$$

for $t = 1, 2, \cdots, N-1$. Then π^ϵ is an ϵ-optimal policy in the sense that $v_N^{\pi^\epsilon} \geq v_N^* - \epsilon$. In particular, if both $\mathcal{A}$ and $\mathcal{M}$ are finite and $O(M)$ is finite for every $M \in \mathcal{M}$, then there is a deterministic policy

$$\pi = (\alpha_0, \beta_1, \alpha_1, \cdots, \beta_{N-1}, \alpha_{N-1})$$

such that

$$r_t(M_t, m_t, \beta_t(M_t, m_t)) + \sum_{m_{t+1}} p_{t+1} \times$$
$$u_{t+1}^*(h_t, \beta_t(M_t, m_t), \alpha_t(M_t, m_t), m_{t+1}) = u_t^*(h_t)$$

for $t = 1, 2, \cdots, N-1$. Based on this observation, a dynamic programming algorithm can be developed for computing the optimal expected reward v_N^* and finding an optimal policy for the case that $\mathcal{A}$, $\mathcal{M}$ and all $O(M)$ with $M \in \mathcal{M}$ are finite.

Algorithm 2. Finding optimal policy

1) Set $t = N$ and $u_N^*(h_N) = r_N(M_N, m_N)$ for $h_N \in H_N$ with $tail(h_N) = (M_N, m_N)$.

2) Substitute $t-1$ for t. Compute $u_t^*(h_t)$ for $h_t \in H_t$ with $tail(h_t) = (M_t, m_t)$ using (13) (with u_t and u_{t+1} be replaced by u_t^*, u_{t+1}^*, respectively).

Set

$$(\beta_t(M_t, m_t), \alpha_t(M_t, m_t)) \in \arg \max_{(a,M) \in \mathcal{A} \times \mathcal{M}}$$
$$\left\{ r_t(M_t, m_t, a) + \sum_{m \in O(M)} p_{t+1} \times u_{t+1}(h_t, a, M, m) \right\}$$

3) If $t = 1$, go to Step 4). Otherwise, return to Step 2).

4) Compute v_N^* by (12).

As in Algorithm 1, the problem of computing the quantum probability p_{t+1} arises in Step 2) of this algorithm. Furthermore, it is easy to see that Algorithm 2 has the same complexity as Algorithm 1.

6 Computation of quantum probabilities

Now we present a method for computing the quantum probabilities p_t needed in both Algorithms 1 and 2. First of all, an elegant formula for p_t can be easily derived from its defining equation (3) by induction on t.

Lemma 5. For each $h_t = (M_1, m_1, a_1, \cdots, M_{t-1}, m_{t-1}, a_{t-1}, M_t, m_t) \in H_t$, we have:

$$p_t = p_t(h_t) = \frac{\mathrm{tr}(\sigma_t)}{\mathrm{tr}(\sigma_{t-1})}$$

where σ_j's are defined by (5).

Proof. By induction on t. □

However, it is hard to compute probability p_t directly using the above lemma because $(t-1)$-fold iterations of super-operators occurs in σ_t. The matrix representation of a super-operator is usually easier to manipulate than the super-operator itself. Suppose super-operator $\mathcal{E}$ has the representation:

$$\mathcal{E}(\rho) = \sum_i E_i \rho E_i^\dagger$$

for all density matrices ρ. Then the matrix representation of $\mathcal{E}$ is the $n^2 \times n^2$ matrix:

$$M = \sum_i E_i \otimes E_i^*$$

where A^* stands for the conjugate of matrix A, i.e., $A^* = (a_{ij}^*)$ with a_{ij}^* being the conjugate of complex number a_{ij}, whenever $A = (a_{ij})$. We write:

$$|\Phi\rangle = \sum_j |jj\rangle$$

for the (unnormalized) maximally entangled state, where $\{|j\rangle\}$ is an orthonormal basis of $\mathcal{H} = \mathbf{C}^n$.

Lemma 6. ([6], Lemma 2.1) Let I be the $n \times n$ unit matrix and M the matrix representation of super-operator $\mathcal{E}$. Then for any density matrix ρ, we have:

1) $(\mathcal{E}(\rho) \otimes I)|\Phi\rangle = M(A \otimes I)|\Phi\rangle$.

2) $\mathrm{tr}(\rho) = \langle\Phi|\rho \otimes I|\Phi\rangle$.

For every j, we write N_j for the matrix representation of super-operator $\mathcal{E}_j(\cdot|a_j)$, and

$$L_j = M_{jm_j} \otimes M^*_{jm_j}.$$

Then a combination of the above two lemmas yields an elegant formula for computing the quantum probability p_t through ordinary matrix multiplications:

Theorem 3.

$$p_t = \frac{\langle\Phi|N_t L_{t-1} N_{t-1} \cdots L_1 N_1(\rho \otimes I)|\Phi\rangle}{\langle\Phi|N_{t-1} L_{t-2} N_{t-2} \cdots L_1 N_1(\rho \otimes I)|\Phi\rangle}.$$

Proof. This theorem can be easily proved by combining Lemmas 5 and 6. □

7 An illustrative example

We now give an example to illustrate the ideas introduced in the previous sections. Suppose a quantum robot ♣ is walking in an $(n_h + 1) \times (n_v + 1)$ grid environment shown in Fig. 1 (when $n_h = 3$ and $n_v = 2$). Initially, the robot is at the $S = (0, 0)$ location. At each decision epoch, it can choose to move horizontally or vertically, each implemented by a (one-dimensional) Hadamard quantum walk[21]. After each move, the robot's location information is (partially) obtained by making a measurement detecting its positions. If the robot is found outside the grid, it gets a penalty (a negative reward $-r$), and then restarts from the original point S; If it reaches the target slot $T = (n_h, n_v)$, then it stays there and a reward R is received; For other cases, no reward or penalty incurs. We assume that $n_h + n_v > 0$ and $R > r > 0$.

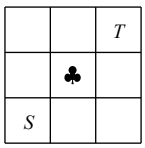


Fig. 1 A quantum robot walking in a grid (with $n_h = 3$ and $n_v = 2$

Formally, let $\mathcal{H}_h$ and $\mathcal{H}_v$ be two 2-dimensional vector spaces with $\{|0\rangle_h, |1\rangle_h\}$, $\{|0\rangle_v, |1\rangle_v\}$, respectively, as an orthonormal basis. They will serve as the state spaces of the coins for the horizontal and vertical walks, respectively. The location space $\mathcal{H}_p$ is the vector space with $\{|i, j\rangle_p : (i, j) \in \overline{G}\}$ as an orthonormal basis, where

$$\overline{G} = \{-1, \cdots, n_h + 1\} \times \{-1, \cdots, n_v + 1\}.$$

The shift operators for the horizontal and vertical walks are defined by

$$S_{hp} = \sum_{(i,j)\in G} \sum_{k\in\{0,1\}} |k\rangle_h\langle k| \otimes |i - (-1)^k, j\rangle_p\langle i, j|$$

and

$$S_{pv} = \sum_{(i,j)\in G} \sum_{k\in\{0,1\}} |i, j - (-1)^k\rangle_p\langle i, j| \otimes |k\rangle_v\langle k|$$

respectively, where

$$G = \{(i, j) \in \overline{G} \mid 0 \le i \le n_h, 0 \le j \le n_v, (i, j) \ne (n_h, n_v)\}.$$

Note that we use subscripts to indicate which subsystems the corresponding operators are performed on. For example, S_{hp} acts only on systems $\mathcal{H}_h$ and $\mathcal{H}_p$. For $\star \in \{h, v\}$, let $\mathcal{W}^\star$ be the quantum-walk super-operator in the grid $\overline{G}$ along direction $\star$, except that it stops when reaching the target slot or getting out of the grid, i.e., for any quantum state ρ,

$$\mathcal{W}^\star(\rho) = U^\star \rho\, U^{\star\dagger} + \sum_{(i,j)\in\overline{G}\setminus G} P_{i,j}\rho P_{i,j}$$

where

$$U^h = (S_{hp} \otimes I_v)(H_h \otimes I_p \otimes I_v)$$
$$U^v = (I_h \otimes S_{pv})(I_h \otimes I_p \otimes H_v)$$
$$P_{i,j} = I_h \otimes |i, j\rangle_p\langle i, j| \otimes I_v$$

and H, I stand for the Hadamard matrix and the unit matrix, respectively. Let $\mathcal{E}_{reset}$ be the super-operator which resets the robot to the initial state (i.e., the position to $(0, 0)$, and the coin states to $|0\rangle$) when it walks outside the grid. To be specific, $\mathcal{E}_{reset}$ has the Kraus operators:

$$\{E^{in}, E^{out}_{i,j,k,l} \mid (i, j) \in \overline{G}\setminus G^{in}, k, l \in \{0, 1\}\}$$

where $G^{in} = G \cup \{(n_h, n_v)\}$, and

$$E^{in} = I_h \otimes \sum_{(i,j)\in G^{in}} |i, j\rangle_p\langle i, j| \otimes I_v$$
$$E^{out}_{i,j,k,l} = |0\rangle_h\langle k| \otimes |0, 0\rangle_p\langle i, j| \otimes |0\rangle_v\langle l|.$$

Now, for $t \in \mathcal{T}$, let

$$\mathcal{E}_t(\cdot|\star) = \mathcal{E}_{reset} \circ \mathcal{W}^\star$$

be a composed super-operator obtained by first applying $\mathcal{E}_{reset}$ and then $\mathcal{W}^\star$. Moreover, let $\mathcal{M} = \{\Pi_p\}$ be a set consisting of a single measurement Π on system $\mathcal{H}_p$, where $\Pi = \{\Pi^!, \Pi^\times, \Pi^?\}$, $\Pi^! = |n_h, n_v\rangle\langle n_h, n_v|$,

$$\Pi^? = \sum_{(i,j)\in G} |i, j\rangle\langle i, j|$$

and $\Pi^\times = I_p - \Pi^! - \Pi^?$. Finally, we define:

$$r(m) = \begin{cases} R, & \text{if } m = ! \\ -r, & \text{if } m = \times \\ 0, & \text{if } m = ?. \end{cases}$$

For each $1 \leq t \leq N-1$, $M \in \mathcal{M}$, and $a \in \{h, v\}$, let $r_t(M, m, a) = r_N(M, m) = r(m)$.

With the notations presented above, the robot-walking system can be modelled by a qMDP:

$$\mathcal{P} = (\mathcal{T}, \mathcal{H}, \rho, \mathcal{A}, \{\mathcal{E}_t(\cdot|a) : t \in \mathcal{T}, a \in \mathcal{A}\}, \mathcal{M}, \{r_t : t \in \mathcal{T}\})$$

where $\mathcal{H} = \mathcal{H}_h \otimes \mathcal{H}_p \otimes \mathcal{H}_v$, $\rho = |0\rangle_h\langle 0| \otimes |0,0\rangle_p\langle 0,0| \otimes |0\rangle_v\langle 0|$, and $\mathcal{A} = \{h, v\}$. As $\mathcal{M}$ contains only a single measurement, we can simply denote a history in H_t by $h_t = (m_1, a_1, m_2, \cdots, a_{t-1}, m_t)$. Furthermore, it is easy to see that $m_1 = ?$ in any history h_t.

In the remainder of this section, we compute the optimal expected total reward v_N^* as well as (one of) the corresponding policies in several simple cases. Our strategy is to first calculate for any $h_t = (m_1, a_1, m_2, \cdots, a_{t-1}, m_t) \in H_t$, $1 \leq t \leq N$, the probability $p_t = p_t(m_t \mid h_{t-1}, a_{t-1})$ defined in (3). Then (13) is recursively employed to calculate $u_t(h_t)$. Finally, we get $v_N^* = u_1(?)$ from Lemma 4 and Theorem 2.

Case $N = 1$: This case is trivial, as no decision is needed to make, and $v_1^* = r(?) = 0$.

Case $N = 2$: From (13), we have for any $h_2 = (?, a_1, m_2)$, $u_2(h_2) = r(m_2)$, and

$$u_1(?) = \max_{a_1 \in \{h,v\}} \{p_2(! \mid ?, a_1) \cdot R - p_2(\times \mid ?, a_1) \cdot r\}$$

where

$$p_2(! \mid ?, a_1) = \begin{cases} 1/2, & \text{if } n_{a_1} = 1 \ \wedge \ n_{\overline{a}_1} = 0 \\ 0, & \text{otherwise} \end{cases}$$

$\overline{a}_1$ denotes the direction other than a_1, and

$$p_2(\times \mid ?, a_1) = \begin{cases} 1/2, & \text{if } n_{a_1} > 0 \\ 1, & \text{if } n_{a_1} = 0. \end{cases}$$

Thus,

$$v_2^* = u_1(?) = \begin{cases} (R-r)/2, & \text{if } n_h + n_v = 1 \\ -r/2, & \text{otherwise} \end{cases}$$

and one of the optimal policies could be taking $a_1 = h$ if $n_h \geq 1$, and $a_1 = v$ otherwise.

Case $N = 3$: For any $h_2 = (?, a_1, m_2)$,

$$u_2(h_2) = r(m_2) + \max_{a_2 \in \{h,v\}} \{p_3(! \mid h_2, a_2) \times R - p_3(\times \mid h_2, a_2) \times r\}.$$

Note that when $m_2 = \times$, the robot is reset to the initial state. Thus $p_3(m \mid h_2, a_2) = p_2(m \mid ?, a_2)$ where p_2 is computed in the previous case. Similarly, as the robot is terminated when $m_2 = !$, $p_3(m \mid h_2, a_2) = 1$ if $m = !$ and 0 otherwise. Furthermore, it is not hard to show that

$$p_3(! \mid ?, a_1, ?, a_2) = \begin{cases} 1/2, & \text{if } a_1 = a_2 \ \wedge \ n_{a_1} = 2 \ \wedge \ n_{\overline{a}_1} = 0 \\ 1/2, & \text{if } a_1 \neq a_2 \ \wedge \ n_h = n_v = 1 \\ 0, & \text{otherwise} \end{cases}$$

and

$$p_3(\times \mid ?, a_1, ?, a_2) = \begin{cases} 1/2, & \text{if } a_1 = a_2 \ \wedge \ n_{a_1} = 1 \ \wedge \ n_{\overline{a}_1} > 0 \\ 1/2, & \text{if } a_1 \neq a_2 \ \wedge \ n_h > 0 \ \wedge \ n_v > 0 \\ 1, & \text{if } a_1 \neq a_2 \ \wedge \ n_{a_1} > 1 \ \wedge \ n_{a_2} = 0 \\ 0, & \text{otherwise.} \end{cases}$$

With these we know that for any $a_1 \in \{h, v\}$, $u_2(?, a_1, !) = 2R$, $u_2(?, a_1, \times) = -r + v_2^*$, and

$$u_2(?, a_1, ?) = \max_{a_2} \{p_3(! \mid ?, a_1, ?, a_2) \cdot R - p_3(\times \mid ?, a_1, ?, a_2) \times r\} = \begin{cases} R/2, & \text{if } n_{a_1} = 2 \ \wedge \ n_{\overline{a}_1} = 0 \\ (R-r)/2, & \text{if } n_h = n_v = 1 \\ -r/2, & \text{if } n_{a_1} = 1 \ \wedge \ n_{\overline{a}_1} > 1 \\ 0, & \text{otherwise} \end{cases}$$

and one of the optimal policies could be taking $a_2 = \overline{a}_1$ if $n_h = n_v = 1$, and $a_2 = a_1$ otherwise. Furthermore,

$$u_1(?) = \max_{a_1} \left\{ \sum_{m \in \{!, \times, ?\}} p_2(m \mid ?, a_1) \cdot u_2(?, a_1, m) \right\}.$$

Let $T(m) = p_2(m \mid ?, a_1) \cdot u_2(?, a_1, m)$. We compute:

$$T(!) = \begin{cases} R, & \text{if } n_{a_1} = 1 \ \wedge \ n_{\overline{a}_1} = 0 \\ 0, & \text{otherwise} \end{cases}$$

$$T(\times) = \begin{cases} (R-3r)/2, & \text{if } n_{a_1} = 0 \ \wedge \ n_{\overline{a}_1} = 1 \\ -3r/2, & \text{if } n_{a_1} = 0 \ \wedge \ n_{\overline{a}_1} > 1 \\ (R-3r)/4, & \text{if } n_{a_1} = 1 \ \wedge \ n_{\overline{a}_1} = 0 \\ -3r/4, & \text{otherwise} \end{cases}$$

$$T(?) = \begin{cases} R/4, & \text{if } n_{a_1} = 2 \ \wedge \ n_{\overline{a}_1} = 0 \\ (R-r)/4, & \text{if } n_h = n_v = 1 \\ -r/4, & \text{if } n_{a_1} = 1 \ \wedge \ n_{\overline{a}_1} > 1 \\ 0, & \text{otherwise.} \end{cases}$$

Thus,

$$v_3^* = u_1(?) = \begin{cases} (5R-3r)/4, & \text{if } n_{\min} = 0 \wedge n_{\max} = 1 \\ (R-3r)/4, & \text{if } n_{\min} = 0 \wedge n_{\max} = 2 \\ (R-4r)/4, & \text{if } n_{\min} = 1 \wedge n_{\max} = 1 \\ -3r/4, & \text{otherwise} \end{cases}$$

where $n_{\max} = \max\{n_h, n_v\}$, $n_{\min} = \min\{n_h, n_v\}$, and one

of the optimal policies could be taking a_1 such that $n_{a_1} = n_{\max}$. Furthermore, $a_2 = \overline{a}_1$ if $n_h = n_v = 1$; otherwise, take $a_2 = a_1$.

8 Concluding remarks

In this paper, we studied the optimal policy for qMDPs of finite-horizon. It is shown that the problem can by solved by dynamic programming together with matrix multiplications for computing quantum probabilities. We hope that the mathematical framework of qMDPs developed in [3, 4] and this paper can provide certain theoretical foundations for quantum reinforcement learning[17–20], quantum robot planning[22–24] and other decision-making tasks in the quantum world.

For future studies, one of the most interesting problems is to settle the complexity of the optimal policy problem (as well as other problems) for qMDPs (in comparison with that for classical MDPs and POMDPs[25, 26]). Another interesting problem is to find quantum algorithms (rather than classical algorithms as considered in the present paper) for solving qMDPs, in particular, speeding up the computation of quantum probabilities. Since the state space of a qMDP is a continuum and thus doomed-to-be infinite, it will be useful to extend analysis techniques for MDPs with infinite state spaces, e.g., bisimulation and metrics[27, 28], to the quantum case.

Acknowledgements

This work has been partly supported by National Key R&D Program of China (No. 2018YFA0306701), the Australian Research Council (Nos. DP160101652 and DP180100691), National Natural Science Foundation of China (No. 61832015) and the Key Research Program of Frontier Sciences, Chinese Academy of Sciences.

Open Access

References

[1] M. L. Puterman. *Markov Decision Processes*: *Discrete Stochastic Dynamic Programming*, Hoboken, USA: John Wiley, 2005.

[2] L. P. Kaelbling, M. L. Littman, A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, vol. 101, no. 1–2, pp. 99–134, 1998. DOI: 10.1016/S0004-3702(98)00023-X.

[3] J. Barry, D. T. Barry, S. Aaronson. Quantum partially observable Markov decision processes. *Physical Review A*, vol. 90, no. 3, Article number 032311, 2014. DOI: 10.1103/PhysRevA.90.032311.

[4] S. G. Ying, M. S. Ying. Reachability analysis of quantum Markov decision processes. *Information and Computation*, vol. 263, pp. 31–51, 2018. DOI: 10.1016/j.ic.2018.09.001.

[5] M. S. Ying. *Foundations of Quantum Programming*, Amsterdam, Netherlands: Morgan Kaufmann, 2016.

[6] M. S. Ying, N. K. Yu, Y. Feng, R. Y. Duan. Verification of quantum programs. *Science of Computer Programming*, vol. 78, no. 9, pp. 1679–1700, 2013. DOI: 10.1016/j.scico.2013.03.016.

[7] J. Guan, Y. Feng, M. S. Ying. Decomposition of quantum Markov chains and its applications. *Journal of Computer and System Sciences*, vol. 95, pp. 55–68, 2018. DOI: 10.1016/j.jcss.2018.01.005.

[8] M. S. Ying, Y. Feng. *Model Checking Quantum Systems*: *Principles and Algorithms*, Cambridge, USA: Cambridge University Press, 2021.

[9] S. G. Ying, Y. Feng, N. K. Yu, M. S. Ying. Reachability probabilities of quantum Markov chains. In *Proceedings of the 24th International Conference on Concurrency Theory*, Springer, Buenos Aires, Argentina, pp.334–348, 2013. DOI: 10.1007/978-3-642-40184-8_24.

[10] D. Powell. Quantum boost for artificial intelligence. *Nature*, to be published.

[11] M. S. Ying. Quantum computation, quantum theory and AI). *Artificial Intelligence*, vol. 174, no. 2, pp. 162–176, 2010. DOI: 10.1016/j.artint.2009.11.009.

[12] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, S. Lloyd. Quantum machine learning. *Nature*, vol. 549, no. 7671, pp. 195–202, 2017. DOI: 10.1038/nature23474.

[13] V. Dunjko, H. J. Briegel. Machine learning & artificial intelligence in the quantum domain: A review of recent progress. *Reports on Progress in Physics*, vol. 81, no. 7, Article number 074001, 2018. DOI: 10.1088/1361-6633/aab406.

[14] S. D. Sarma, D. L. Deng, L. M. Duan. Machine learning meets quantum physics. *Physics Today*, vol. 72, no. 3, pp. 48–54, 2019. DOI: 10.1063/PT.3.4164.

[15] L. P. Kaelbling, M. L. Littman, A. W. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, vol. 4, pp. 237–285, 1996. DOI: 10.1613/jair.301.

[16] R. S. Sutton, A. G. Barto. *Reinforcement Learning*: *An Introduction*, Cambridge, USA: MIT Press, 1998.

[17] D. Y. Dong, C. L. Chen, Z. H. Chen. Quantum reinforcement learning. *In Proceedings of the 1st International*

Conference on Advances in Natural Computation, Springer, *Changsha, China*, pp.686--689, 2005. DOI: 10.1007/11539117_97.

[18] D. Y. Dong, C. L. Chen, H. X. Li, T. J. Tarn. Quantum reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B* (*Cybernetics*), vol. 38, no. 5, pp. 1207–1220, 2008. DOI: 10.1109/TSMCB.2008.925743.

[19] V. Dunjko, J. M. Taylor, H. J. Briegel. Quantum-enhanced machine learning. *Physical Review Letters*, vol. 117, no. 13, Article number 130501, 2016. DOI: 10.1103/PhysRevLett.117.130501.

[20] V. Dunjko, J. M. Taylor, H. J. Briegel. Advances in quantum reinforcement learning. *Proceedings of 2017 IEEE International Conference on Systems, Man, and Cybernetics*, IEEE, Banff, Canada, pp.282–287, 2017. DOI: 10.1109/SMC.2017.8122616.

[21] A. Ambainis, E. Bach, A. Nayak, A. Vishwanath, J. Watrous. One-dimensional quantum walks. In *Proceedings of the 33rd ACM Symposium on Theory of Computing*, ACM, Heraklion, Greece, pp.37–49, 2001. DOI: 10.1145/380752.380757.

[22] P. Benioff. Some foundational aspects of quantum computers and quantum robots. *Superlattices and Microstructures*, vol. 23, no. 3-4, pp. 407–417, 1998. DOI: 10.1006/spmi.1997.0519.

[23] P. Benioff. Quantum robots and environments. *Physical Review A*, vol. 58, no. 2, pp. 893–904, 1998. DOI: 10.1103/PhysRevA.58.893.

[24] D. Y. Dong, C. L. Chen, C. B. Zhang, Z. H. Chen. Quantum robot: Structure, algorithms and applications. *Robotica*, vol. 24, no. 4, pp. 513–521, 2006. DOI: 10.1017/S0263574705002596.

[25] M. Mundhenk, J. Goldsmith, C. Lusena, E. Allender. Complexity of finite-horizon Markov decision process problems. *Journal of the ACM*, vol. 47, no. 4, pp. 681–720, 2000. DOI: 10.1145/347476.347480.

[26] C. H. Papadimitriou, J. N. Tsitsiklis. The complexity of Markov decision processes. *Mathematics of Operations Research*,vol. 12,no. 3,pp. 441–450,1987.DOI:10.1287/moor.12.3.441.

[27] N. Ferns, P. S. Castro, D. Precup, P. Panangaden. Methods for computing state similarity in Markov decision processes. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*, AUAI, Cambridge, USA, pp.174–181, 2006.

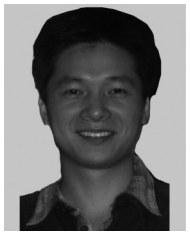
[28] N. Ferns, P. Panangaden, D. Precup. Metrics for Markov decision processes with infinite state spaces. *In Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence*, *Edinburgh*, *Scotland*, pp. 201–208, 2005.

Ming-Sheng Ying is a Distinguished Professor and Research Director of the Center for Quantum Software and Information at the University of Technology Sydney, Australia. He is also Deputy Director for Research (adjunct position) at the Institute of Software at the Chinese Academy of Sciences, and holds the Cheung Kong Chair Professorship at Tsinghua University, China. He has published books: *Model Checking Quantum Systems: Principles and Algorithms* (2021) (with Yuan Feng), *Foundations of Quantum Programming* (2016) and *Topology in Process Calculus: Approximate Correctness* and *Infinite Evolution of Concurrent Programs* (2001). He received a China National Science Award in Natural Science (2008). He has served on the editorial board of several publications including *Artificial Intelligence*. He is currently Editor-in-Chief of *ACM Transactions on Quantum Computing*.

His research interests include quantum computation, theory of programming languages, and logics in AI.

Email: mingsheng.ying@uts.edu.au (Corresponding author)

ORCID iD: 0000-0003-4847-702X



Yuan Feng received the B.Sc. degree in mathematics from Department of Applied Mathematics, Tsinghua University, China in 1999, and received the Ph. D. degree in computer science from Department of Computer Science and Technology, Tsinghua University, China in and 2004. He is currently a professor at Centre for Quantum Software and Information (QSI), University of Technology Sydney (UTS), Australia.

His research interests include quantum programming theory, quantum information and quantum computation, and probabilistic systems.

E-mail: yuan.feng@uts.edu.au

ORCID iD: 0000-0002-3097-3896

Sheng-Gang Ying received the B. Sc. degree in physics from Department of Physics, Tsinghua University, China in 2010, and received the Ph. D. degree in computer science from Department of Computer Science and Technology, Tsinghua University, China in 2015, He is currently an associate researcher at State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, China.

His research interests include quantum programming theory, quantum Markov systems.

E-mail: yingsg@ios.ac.cn

ORCID iD: 0000-0002-5052-5142