A Survey of Markov Model in Reinforcement Learning

Tianhan Gao
Software College
Northeastern University
Shenyang, China
gaoth@mail.neu.edu.cn

Baicheng Chen
Software College
Northeastern University
Shenyang, China
2071264@stu.neu.edu.cn

Qingwei Mi
Software College
Northeastern University
Shenyang, China
2110491@stu.neu.edu.cn

***Abstract*—There was a famous mathematician from Russian in the early 20th century whose name is Andrey Andreyevich Markov. After he proposed the Markov process, the theories have been effectively developed in the past years and have been widely adopted in different fields. This paper both summarizes and does incomplete statistics on Markov models which are related to reinforcement learning. It will help the readers to quickly clarify the relationship between these theories and have an overall understanding of them.**

Keywords—Markov model, Reinforcement Learning, Markov chain, Markov process

I. Introduction

The theories about the Markov process will be called uniformly Markovian theories in this paper for convenience. Although there are still a lot of theories except Markovian theories in the field of reinforcement learning, Markovian theories can be always considered as the basis for the implementation of most of the algorithms in reinforcement learning. It will be better to know what the Markov model is before move on to other theories, and after that, it will be much easier to understand for example the Markov chain, process, decision process, etc. Among them, the Markov decision-making process (MDP) can be considered as the basis theory of reinforcement learning. Besides the theories about the Markov process, many variants have been proposed based on them, and people have come up with a lot of new ideas for the use of them. This paper presents incomplete statistics on the Markovian theories especially in the field of reinforcement learning, which mainly summarizes the variation and usage ideas of kinds of Markovian theories, so that the reader can both quickly understand the structure of the theories and reinforcement learning in other fields, and have a view of reinforcement learning from the perspective of Markovian theories.

In the early 20th century, Andrey Andreyevich Markov, who is a mathematician from Russian, worked on the Markov process[1] and published a paper on this subject in 1906[2]. Before his work, the Poisson process had been discovered, and it can be considered as a kind of Markov process which is continuous in time.

II. Markov Property

Because Markovian theories contain a lot of knowledge, learning them in a certain order will lead to better learning results. While the Markov model, process, and chain make up the bulk of Markovian theories, Markov property should be considered the most valuable to learn at the beginning. Because almost any theory that can be called a Markov theory should have or fit the Markov property.

A. Introduction

In probability theory and statistics, stochastic processes can be considered to have memorability. This kind of property was later summarized as the Markov Property named after Andrey Markov. It is a kind of property that can be used to define a special environment and the environment's state signal. In the field of reinforcement learning, ideally, researchers who want to predict the future will need the past state signal that retains all the information which can summarize the past. If a state signal successfully retains all the information which can summarize the past, then this signal is Markovian or has Markov Property.

B. Definition

The number of states and reward values assumed should not be infinite so that the problem can be calculated based on 'sum' of different kinds of data and “probabilities” rather than “integral” and the “probability density”, otherwise researchers will have to worry about the assumption where the number of states and reward values are unlimited, as the argument can be difficult to be extended to include continuous states and rewards.

Assuming the action is taken at time t, then what the environment will react at time t+1 will be the response. In the most general causal case, this response depends on what happened before. In this case, the dynamic [4] of the environment can be defined by specifying the full joint probability distribution:

$$Pr\{S_{t+1}=s', R_{t+1}=r|S_0,A_0,R_1,...S_{t-1},A_{t-1},R_t,S_t,A_t\} \tag{1}$$

for all r, s’, and all possible values of the past events: S_0, A_0, R_1, ..., S_{t-1}, A_{t-1}, R_t, S_t, A_t. On the other hand, if the state signal has Markov Property, then the response of the environment at t+1 depends only on the state and action representations at t, in which case the dynamics of the environment should be defined as:

$$p(s',r|s,a)\doteq Pr\{S_{t+1}=s', R_{t+1}=r|S_t=s,A_t=a\} \tag{2}$$

for all r, s’, s, and a.

C. Strong Markov property

Strong Markov property is similar to Markov property. The most important difference between them is that the Strong Markov property contains a stopping time, which is a specific type of “random time”. This kind of “random time” is always defined by a stopping rule which is a mechanism for deciding whether the process should be stopped or not. The rule is usually based on current or past state. And stopping time cannot be infinite in general.

The Markov property means that it is sufficient to predict the future from the current state because the current state is the result of all previous states. In another word, it is not necessary to collect all the past states' information, just use the current state's information.

The Markov property understands time in the dimension of time, but the Strong Markov attribute understands time in the perspective of regular logic.

III. Markov Models in Reinforcement Learning

In probability theory, the Markov model is a stochastic model used to simulate pseudo-randomly changing systems. It assumes that the future state depends only on the current state and not on the events that occurred before it (The Markov Property). In general, this assumption makes the model accessible for inference and computation that it would otherwise be difficult to handle. Thus, in the field of prediction models and probabilistic prediction, it is desirable to assume that a given model exhibits the Markov Property [5].

A. The relationship between Markov model and Markov process

When the system state is both automatic and completely visible, the Markov model can be called a Markov chain. And the Markov process is a continuous-time version of the Markov chain.

B. Classification of the Markov models

There are four common Markov models in different scenarios, depending on whether each sequence state is observable, and whether the system is to be adjusted for the observations:

- Markov chain.
- Hidden Markov models.
- Markov decision process.
- Partially observable Markov decision process.

TABLE I. Classification of the Markov Models

	System state is completely visible	***System state is partially observable***
System is autonomous	Markov chain	Hidden Markov model
System is controlled	Markov decision process	Partially observable Markov decision process

1) Markov chain

a) Definition: The Markov property can be called "memorylessness" sometimes because it means that in a stochastic process that satisfies it, the predictions of the process can be made based solely on the process's present state and the predictions are as good as the one that could be made knowing all the process's history. And this kind of process can be called the Markov process. There are different definitions of a Markov chain. The most common is that a Markov chain is a Markov process having discrete-time in either countable or continuous state space. But some definitions are regardless of the nature of time and say that a Markov chain is a Markov process with a countable state space. For example, if you ignore time, you can define a Markov chain as a Markov process in a countable state space, and a Markov chain as a Markov process in discrete time if you ignore a state space.

b) Types of the Markov chain: Depending on different kinds of state spaces and discrete-time v. continuous time, there will be four kinds of Markov chains:

- (Discrete-time) Markov chains on a countable or finite-state space.
- Markov chains on a measurable state space (e. g., Harris chains).
- Continuous time Markov process or a Markov jump process.
- Any continuous stochastic process with the Markov property (for example, the Wiener process).

TABLE II. Types of Markov Chains

	Countable state space	***Continuous or general state space***
Discrete-time	(discrete-time) Markov chain on a countable or finite state space	Markov chain on a measurable state space (for example, Harris chain)
Continuous-time	Continuous-time Markov process or Markov jump process	Any continuous stochastic process with the Markov property (for example, the Wiener process)

The Markov process is a stochastic process where, given the present case, the future is independent of the past. Sometimes, the Markov process is also known as a version of the Markov chain with continuous time [6]. A simple representation of the Markov process is shown below:

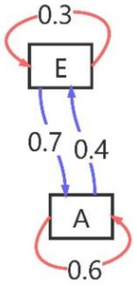

Fig. 1. Example of a Markov process with two states marked E and A. And each number represents the probability that the Markov process changes from one state to another, with the direction indicated by the arrow.

2) Hidden Markov model

The Hidden Markov model (HMM) is a statistical model that was first proposed by Baum L.E. He uses a Markov process that contains hidden and unknown parameters. In this model, the observed parameters are used to identify the hidden parameters. Its state cannot be directly observed but can be identified by observing the vector series [7]. The hidden Markov models are probabilistic frameworks where the observed data are modeled as a series of outputs generated by one of several internal states [8].

In general, when considering a Markov model, all its processes should be observable. However, an HMM is a doubly stochastic process with an underlying stochastic process that is not observable but can only be observed through another set of stochastic processes that produce the sequence of observed symbols [9].

If the state of a Markov model is only partially observable or noisily observable, this Markov model can be called a hidden Markov Model.

3) Markov decision process

a) Definition: Markov decision process (MDP) is also a kind of Markov chain. In general, the one that tries to control the system will be called the agent. And there may be many agents in one system. The MDP state transitions are depending on the current state and the actions the agents take. Typically, an MDP is always being used to compute a policy of actions that will maximize some utility concerning expected rewards. And that is why nearly all the reinforcement learning algorithms are based on MDP. Sometimes, a MDP will be defined by the tuple whose number of parameters is larger than four for the computation of reinforcement learning (RL) because RL needs parameters to representing the learning rate, discount factor and so on. In general, A MDP will be represented by a four-parameter tuple$\{S, A, P_a, R_a\}$. S is a finite set of states, called the state space. A is a finite set of actions, called behavioral space, while as are a series of actions that can be performed under state S.

$$P_a(s,s')=Pr(s_{t+1}=s'|s_t=s,a_t=a) \qquad (3)$$

is the probability that taking behavior a at t in state s leads to state s' at time t+1. Ra(s, s') is the instantaneous reward (or the expected instantaneous reward) when the state s turns to s' due to behavior a. State and action space can be finite or infinite. Some processes with countable infinite states and action space can be reduced to processes with finite state and action space [10].

b) Reinforcement Learning

Deep reinforcement learning based on Markovian theories first gained widespread attention in 2013, when Google's Deepmind team first implemented image-based reinforcement learning AI [11]. The algorithm they used was Deep Q-learning (DQN).

Deep Q-learning is a method that tries to get the best strategy by using Q-learning and deep neural networks. Q-learning is a kind of model-free algorithm that tries to get the best policy by calculating the value of each action in each particular state.

After the DQN, algorithms related such as DDPG, AC, A3C, NAF, TRPO, PPO, TD3, SAC were invented in the single-agent field based on the Markovian theories. In the field of multi-agents, algorithms for example IQL, VDN, COMA, QMIX, QTRAN, QTRAN++, Qatten have also been invented.

The DQN cannot be straightforwardly applied to continuous domains since it relies on finding the action that maximizes the action-value function, which in the continuous-valued case requires an iterative optimization process at every step. Based on the deterministic policy gradient (DPG) [12] algorithm, countzero et al. present a model-free, off-policy actor-critic algorithm using deep function approximators that can learn policies in high-dimensional, continuous action spaces which is called the DDPG [13].

And there are some algorithms called Actor-Critic algorithms that can combine the strong points of actor-only and critic-only methods [14].

The Advantage Actor-Critic (A2C) algorithm replaces the original return in the critic network by advantage function. The Asynchronous advantage actor-critic (A3C) algorithm makes it possible to calculate asynchronously while each worker gets the data directly from the global network and interacts with the environment [15].

As a continuous variant of Q learning, the NAF can reduce the sample complexity for continuous control tasks and it can be regarded as an alternative to the more commonly used policy gradient and actor-critic methods [16].

The policy gradient algorithm has four challenges: (1) The large policy change will destroy the training. (2) It cannot map changes between policy and parameter space easily. (3) Improper learning rate causes vanishing or exploding gradient. (4) Low sample efficiency. The trust region policy optimization (TRPO) combines the MM algorithm, Trust region, and Importance sampling and will improve the performance in most cases [17].

PPO algorithm is a new kind of Policy Gradient algorithm. The performance of the Policy Gradient algorithm is very sensitive to the step size, but it is difficult to select the appropriate step size. If the difference between the old strategy and the new strategy is too large in the training process, it will be always difficult to calculate. PPO proposed a new objective function that can be updated in small batches by multiple training steps, which solved the problem that the step size in the Policy Gradient algorithm was difficult to determine. TRPO is also actually trying to solve this problem but it is much easier to do PPO than TRPO [18].

Although DDPG can sometimes achieve excellent performance, it is often not very easy to adjust the hyper-parameters and other things that can be adjusted. A common problem with DDPG is that Q functions learned will sometimes overestimate the Q values. It then causes the policy to break because it exploits an error in the Q function. Twin delayed DDPG (TD3) [19] is an algorithm that solves this problem by introducing three key tricks: clipped double-Q learning, “delayed” policy updates, and target policy smoothing.

There are several algorithms based on MDP and are famous in the field of multi-agent reinforcement learning.

The independent q learning (IQL) [20] algorithm regards the other agents directly as a part of the environment, which means that each agent in the environment is in its single-agent task. It is impossible to guarantee convergence and the agents will easily get lost in the endless exploration because the environment is non-stationary for any one of the agents. But this algorithm`s performance is still relatively acceptable in practice.

In cooperative multi-agent reinforcement learning, each agent chooses actions based on its local observations to maximize team rewards. The Value-Decomposition Networks for Cooperative Multi-Agent Learning (VDN) [21] proposes a way to decompose the team's reward signal to each agent through back-propagation.

There is a credit assignment problem in MARL because the immediate reward of each agent are the same which means the agents who have made a huge contribution and those who have not much contribution will get the same rewards. To solve the problem, the Counterfactual Multi-Agent Policy Gradients (COMA) [22] algorithm uses a centralized critic to estimate the Q-function and decentralized actors to optimize the agents' policies. In addition, to address the challenges of multi-agent credit assignment, it uses a counterfactual baseline that marginalizes out a single agent's action, while keeping the other agents' actions fixed. COMA also uses a “critic” representation that allows the counterfactual baseline to be computed efficiently in a single forward pass.

The full factorization of VDN is not necessary to extract decentralized policies. The Monotonic Value Function Factorization for Deep Multi-Agent Reinforcement Learning (QMIX) [23] is a novel value-based method that can train decentralized policies in a centralized end-to-end fashion.

QMIX employs a network that estimates joint action values as a complex non-linear combination of per-agent values that condition only on local observations. It enforces a monotonicity constraint on the value`s relationship between all the agents and one single agent.

VDN and QMIX address only a fraction of factorizable MARL tasks due to their structural constraint in factorization such as additivity and monotonicity. QTRAN [24] guarantees more general factorization than VDN or QMIX, thus covering a much wider class of MARL tasks than previous methods.

4) Partially observable Markov decision process

While doing reinforcement learning research in most computer games, the agents will know with full certainty the state of the environment. In another word, the agent's sensors will allow it to perfectly monitor the state at all times, where the state captures all aspects of the environment relevant for optimal decision making. However, this kind of situation will rarely happen in the real world. For example, in many robotic applications, the robot's onboard sensors may not be able to enable the robot to unambiguously identify its location or pose. Furthermore, a robot's sensors are often limited to observing its direct surroundings, and there will always be features of the environment's state beyond the robot`s visibility which can be called the hidden state. Another source of uncertainty regarding the true state of the system is imperfections in the robot's sensors. For instance, let us suppose a robot uses a camera to identify the person it is interacting with. The face-recognition algorithm processing the camera images is likely to make mistakes sometimes and report the wrong identity. Although in some domains the issues resulting from imperfect sensing might be ignored, the severe performance degradation caused by it is inevitable. The POMDP captures the partial observability in a probabilistic observation model, which relates possible observations to states [25].

5) Semi-Markov process

The difference between the semi-Markov process and the Markov process is the type of time for which the state is defined. In the Markov process, the state is defined at the jump times. But in the semi-Markov process, the state is defined for every given time. The semi-Markov process is an actual stochastic process that evolves over time [26].

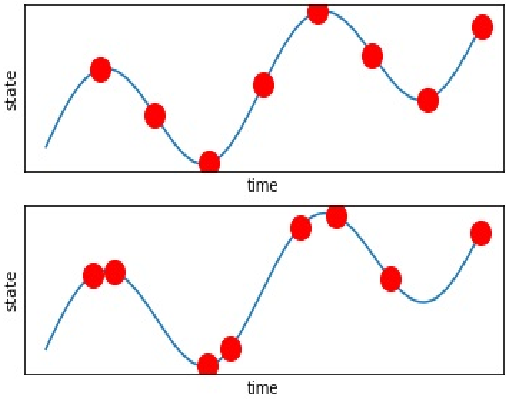

Fig. 2. The difference between Markov process and semi-Markov process

IV. Conclusion

Since the birth of Markovian theories, there has been more than one hundred years of research history. Markovian theories have strongly promoted the development of science and technology. Especially in the field of reinforcement learning, the application of Markov decision process theory has greatly promoted the development of reinforcement learning in recent years. On this basis, a large number of reinforcement learning algorithms have been proposed, although these algorithms are far from perfect. There is still much to be explored in Markovian theories. And the popularity of reinforcement learning will in turn promote the further development of them.

Acknowledgment

This paper is supported by the Fundamental Research Funds for the Central Universities under Grant Number: N2017003.

References

[1] P. A. Gagniuc, *Markov chains: From theory to implementation and experimentation*, 1st ed. Nashville, TN: John Wiley & Sons, 2017.

[2] A. A. Markov and N. M. Nagorny, *The theory of algorithms*. Dordrecht, Netherlands: Kluwer Academic, 2010.

[3] S. M. Ross, *Stochastic Processes*, 2nd ed. Nashville, TN: John Wiley & Sons, 1996.

[4] R. S. Sutton and A. G. Barto, *An Reinforcement Learning: Introduction*. Mit Press, 2012.

[5] P. A. Gagniuc, “Markov chains: From theory to implementation and experimentation.” .

[6] R. Jarrow and P. Protter, “A short history of stochastic integration and mathematical finance: The Early Years, 1880–1970.” .

[7] Y. Lan, D. Zhou, H. Zhang, and S. Lai, “Development of early warning models.” .

[8] M. H. Swat, G. L. Thomas, J. M. Belmonte, A. Shirinifard, D. Hmeljak, and J. A. Glazier, “Multi-scale modeling of tissues using compucell3d.” .

[9] L. Rabiner and B. Juang, “An introduction to hidden Markov models,” *IEEE ASSP mag.*, vol. 3, no. 1, pp. 4–16, 1986.

[10] A. Wrobel, “On Markovian decision models with a finite skeleton,” *Zeitschrift für Operations Research*, vol. 28, no. 1, pp. 17–27, 1984.

[11] V. Mnih *et al.*, “Playing Atari with deep reinforcement learning,” *arXiv [cs.LG]*, 2013.

[12] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, “Deterministic policy gradient algorithms,” in *Proceedings of the 31st International Conference on Machine Learning*, 2014, vol. 32, pp. 387–395.

[13] T. P. Lillicrap *et al.*, “Continuous control with deep reinforcement learning,” *arXiv [cs.LG]*, 2015.

[14] K. V. R and T. J. N, “Actor-critic algorithms[C]//Advances in neural information processing systems.” pp. 1008–1014, 2000.

[15] V. Mnih *et al.*, “Asynchronous methods for deep reinforcement learning,” *arXiv [cs.LG]*, 2016.

[16] L. S., S. T., I., and S. Levine, “Continuous deep q-learning with model-based acceleration.” pp. 2829–2838, 2016.

[17] L. J., A. S., J. P., M., and P. Moritz, “Trust region policy optimization.” pp. 1889–1897, 2015.

[18] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal Policy Optimization Algorithms,” *arXiv [cs.LG]*, 2017.

[19] S. Fujimoto, H. van Hoof, and D. Meger, “Addressing function approximation error in actor-critic methods,” *arXiv [cs.AI]*, 2018.

[20] M. A., “Multiagent cooperation and competition with deep reinforcement learning,” *PloS one*, vol. 12, no. 4, p. 0172395, 2017.

[21] L. P., “Value-decomposition networks for cooperative multi-agent learning.” 2017.

[22] F. J., A. G., N. T., N., and S. Whiteson, “Counterfactual multi-agent policy gradients,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1), 2018.

[23] T. Rashid, M. Samvelyan, C. S. de Witt, G. Farquhar, J. Foerster, and S. Whiteson, “QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning,” *arXiv [cs.LG]*, 2018.

[24] K. K., K. D., W. J., D. E. Hostallero, and Y. Yi, “Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning.” pp. 5887–5896, 2019.

[25] M. T, *Partially observable Markov decision processes*. Berlin, Heidelberg: Springer, 2012.

[26] S. Z, *Hidden Semi-Markov models: theory, algorithms and applications*. Morgan Kaufmann, 2015.