Journal of Artificial Intelligence and Data Mining (JAIDM), Vol. 13, No. 2, 2025, 145-157.

Shahrood University of Technology

Journal of Artificial Intelligence and Data Mining (JAIDM)
Journal homepage: http://jad.shahroodut.ac.ir

Research paper

Employing Chaos Theory for Exploration-Exploitation Balance in Reinforcement Learning

Habib Khodadadi [1] and Vali Derhami [2*]

1. *Department of Computer Engineering, Minab Branch, Islamic Azad University, Minab, Iran.*
2. *Computer Engineering Department, Yazd University, Yazd, Iran.*

Article Info

Article History:
Received 21 December 2024
Revised 10 February 2025
Accepted 18 February 2025

DOI:10.22044/jadm.2025.15292.2633

Keywords:
Action Selection, Chaos Theory, Exploration and Exploitation, Reinforcement Learning.

**Corresponding author: vderhami@yazd.ac.ir (V. Derhami)*

Abstract

The exploration-exploitation trade-off poses a significant challenge in reinforcement learning. For this reason, action selection methods such as ε-greedy and Soft-Max approaches are used instead of the greedy method. These methods use random numbers to select an action that balances exploration and exploitation. Chaos is commonly utilized across various scientific disciplines because of its features, including non-periodicity, unpredictability, ergodicity and pseudorandom behavior. In this paper, we employ numbers generated by different chaotic systems to select action and identify better maps in diverse states and quantities of actions. Based on our experiments on various environments such as the Multi-Armed Bandit (MAB), taxi-domain, and cliff-walking, we found that many of the chaotic methods increase the speed of learning and achieve higher rewards.

1. Introduction

Reinforcement learning is the process of determining the best action to take from a set of permitted options in a specific scenario, guided by the rewards and penalties received [1]. Unlike supervised learning algorithms, this type of learning only uses numerical evaluations and requires a high degree of exploration [2]. To make reinforcement learning effective, actions must be selected in a way that thoroughly explores the environment and properly exploits the knowledge acquired during the learning process. When estimating action-values in each state of a problem, there are two solutions for action selection at each time step: The initial approach is to choose the action that has the highest estimated value (greedy action), leveraging the current understanding and action-value assessments. The second solution is to select a non-greedy action, which focuses more on exploration. Exploration allows the agent to enhance its understanding of non-greedy actions, while exploitation focuses on maximizing expected rewards at each time step, ultimately resulting in a more advantageous solution over the long term [1].
A balance between exploration and exploitation is essential to achieve optimal results in reinforcement learning; however, a balance between exploration and exploitation is a challenge. This balance allows the agent to gain new information about the environment while simultaneously exploiting the current knowledge to achieve high rewards. An appropriate balance will reduce learning time, help the agent evade local optima, and lead to better solutions.
In general, the strategies for selecting action are classified into two categories: direct and indirect exploration methods. In direct exploration methods, it is assumed that some environmental information such as transition probability function and reward function is available, while undirected methods are completely dependent on the Q-values. Since the state transition probability function is usually not available in reinforcement learning problems, indirect exploration methods are often used. The implementation of indirect exploration methods (such as ε-greedy and Soft-Max [3]) is easier, and the two characteristics of "randomness of exploration" and "production of action based on random distributions" distinguish these methods from direct methods [1, 4]. Section 3 provides the different methods of selecting the

action and creating a balance between exploration and exploitation.

Chaos has also been used to establish a balance between exploration and exploitation, and chaotic numbers have been used in the ε-greedy method for exploration. Chaotic systems are considered efficient due to their unique characteristics, including sensitivity to initial values, pseudo-randomness, unpredictability, non-periodicity, and ability to inspect different segments in the state space. In most cases [5, 6], the logistic map is utilized to generate random numbers used in the ε-greedy action selection method. Experiments were conducted on the target capturing task to compare the chaotic state with the normal state. According to the results, the chaotic state yielded better results to find the target. These papers also concluded that the use of random ordinary numbers would be sensitive to ε. Previous studies have explored the use of the chaos theory in the shortcut maze problem. These studies compared the performance of logistic, tent, and chaotic neuron maps in ε-greedy action selection [7- 9]. The results of these studies indicated that the chaos theory was effective in improving performance. In another study, the tent map was compared to the logistic map and found to be less efficient [8]. Additionally, the logistic map was applied to the Licensed Assisted Access (LAA) problem and implemented in the ε-greedy action selection method [10].

Research on the application of chaos theory in action selection is sparse, primarily focusing on just one or two chaotic maps. Moreover, this exploration has largely been restricted to the ε-greedy action selection method and a handful of specific environments. Furthermore, no studies have analyzed the effects of different chaotic relationships on the action selection problem. However, several chaotic systems have been proposed in recent years, which are more efficient than previous ones. In this paper, we use different types of chaotic maps for action selection methods (ε-greedy and Soft-Max) in diverse environments to compare the resultant speed and efficiency with those of conventional methods.

The rest of this paper is organized as follows: Section 2 briefly introduces chaotic systems, and Section 3 presents methods of action selection and exploration-exploitation balance. The proposed method is described in Section 4, results and discussion are reported in Section 5 and 6, and finally, the conclusion is presented in Section 7.

2. Chaotic Systems

Chaotic systems are nonlinear dynamic systems that exhibit pseudorandom behavior and are very sensitive to initial conditions. Minor alterations in the primary conditions of these systems can result in significant transformations over time, a concept referred to as the butterfly effect in chaos theory.

For a system to be classified as chaotic, it must display the following characteristics [11]:

- **Sensitivity to initial conditions**: This trait of chaotic systems demonstrates that minor changes to the initial conditions can result in significantly different outcomes as time progresses.
- **Topological mixing or topological transitivity (ergodicity)**: is a characteristic that states chaotic variables will move through all states within a set range without repeating. This property can serve as an optimization tool to guarantee that solutions are not revisited in the search space, thereby helping algorithms avoid getting trapped in a local optimum. This feature leads to the generation of diverse and non-repeating numbers.
- **Topological density**: This refers to the characteristic that every point in a specified space can be approached by periodic orbits in an arbitrary manner.

Substituting chaos for random numbers has been shown to increase learning speed due to leveraging the special properties of chaos, especially its ergodicity property [11].

Many chaotic systems have been introduced to date and can be divided into two main categories. The first category includes chaotic systems that have specific physical interpretations, such as the Lorenz system [12]. The second category includes chaotic systems that have no specific physical interpretations and are merely mathematical models, such as the Chen chaotic system [13].

The governing equations of the Lorenz system can be seen in (1).

$$\begin{cases} \dot{x}=a(y-x) \\ \dot{y}=bx-y-xz \\ \dot{z}=xy-cz \end{cases} \tag{1}$$

The system is in a chaotic mode when $a = 10, b = 28$, and $c = 8/3$.

The logistic system [14] is another example of a chaotic system, characterized by the following governing equation:

$$X_{n+1}=\lambda X_n(1-X_n) \tag{2}$$

Where the system behaves chaotically for the values of λ within [3.56, 4].

Figure 1 demonstrates the behavior of logistic system with x_0=0.18 and λ =3.9999. The figure illustrates that the generated numbers are uniformly distributed across the range of 0 to 1. Over multiple iterations, various regions within this space are consistently explored; Therefore, using these chaotic numbers instead of random numbers can be justified, and in addition, it allows us to take advantage of the properties of chaos.

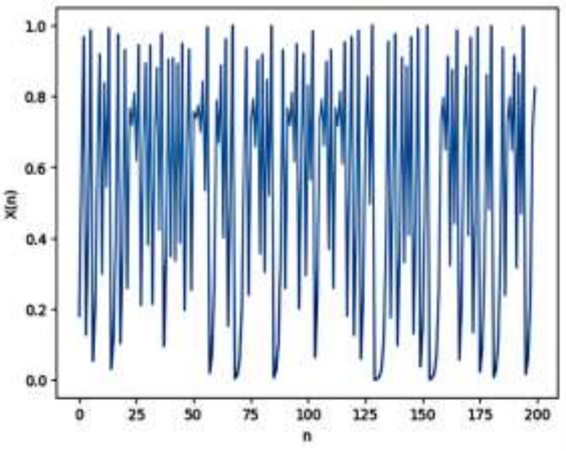

Figure 1. The chaotic behavior of a logistic system during the initial 200 iterations with X_0=0.18 and λ =3.9999.

Despite their pseudorandom behavior, chaotic systems are useful for many applications. Chaos theory has demonstrated significant effectiveness compared to random data across numerous scientific disciplines and various areas of machine learning. For instance, it has enhanced and accelerated the process of identifying global optima in many evolutionary algorithms, and numerous encryption methods are grounded in chaotic principles. This superior performance stems from the distinctive characteristics of chaotic systems, such as their sensitivity to initial conditions and ergodicity. Ongoing research aims to develop more robust chaotic systems, with new models continually being proposed.

Chaotic systems exhibit different levels of sensitivity to initial conditions and control parameters, and not all systems possess the same attributes or chaotic rates. The sensitivity of a chaotic system can be measured using the Lyapunov exponent, with higher values indicating stronger chaotic attributes. For example, the logistic map has a maximum Lyapunov exponent of approximately 0.68, while the SPL map has a value of 1.52 [15]. The Appendix presents the chaotic maps employed in this paper, which vary in their degrees of sensitivity and chaotic rates.

To obtain a chaotic number in this paper, each time a random number is given as an initial value to the system, and the resultant number is then used. Moreover, in most of the chaotic systems of Appendix, the generated numbers are in the range of 0 to 1, and in cases where the number generation range is different from the foregoing interval, it will be mapped onto the 0–1 interval. Furthermore, in the cases where a chaotic system generates more than one number in each iteration, the mean of resultant numbers will be used in the ε-greedy method. In the systems with derivatives, the Runge–Kutta methods are adopted.

3. Action Selection Methods and Exploration–Exploitation Balance

In this section, after a short introduction about reinforcement learning, there will be some content about traditional and new algorithms that are about the balance between exploration and exploitation.

The central concept of reinforcement learning involves leveraging value functions to identify suitable policies. Dynamic programming is one approach within reinforcement learning that employs the Bellman equation to determine the value of each state in the environment, as well as the value of state-action pairs (see equations 3 and 4) [1].

$$V^{\pi}(S) = \sum_{a} \pi(s,a) \sum_{s'} P^{a}_{ss'} \left[R^{a}_{ss'} + \gamma V^{\pi}(s') \right] \quad (3)$$

To calculate the value of each state, the values of other states are used.

$$Q^{\pi}(s,a) = \sum_{s'} P^{a}_{ss'} \left[R^{a}_{ss'} + \gamma V^{\pi}(s') \right] \quad (4)$$

In these equations, action a is chosen from the available set of actions in state s, and the next states s' are members of the set of states. $V^{\pi}(S)$ represents the value of state s according to policy π, while $Q^{\pi}(s,a)$ indicates the value of taking action a in state s under the policy π, $P^{a}_{ss'}$ and $R^{a}_{ss'}$ are the transition probability and the expected reward value to the next state, respectively. Also, $\pi(s,a)$ is the probability of selecting action a in state s and finally, γ is the discount factor [1].

There are multiple algorithms in reinforcement learning, one of the most frequently utilized algorithms is Q-learning. In the Q-learning algorithm, the Q-table is used to store values for all state-action pairs. The rows of the table correspond to different states, while the columns correspond to various actions. Each entry in the table represents an estimate of the optimal value for its associated state-action pair. At every stage of the agent's movement within the environment, this table needs to be updated, and new estimated values should replace the previous values using the received rewards. The update rule of this algorithm is given by equation (5) [4].

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha(r_{t+1} + \gamma \max_a Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)) \quad (5)$$

In this algorithm, α and r denote the learning rate and the instant reward, respectively. Here, the learned state-action value function Q serves as a direct approximation of the optimal state-action value function, regardless of the policy that governs behavior.

For effective learning, actions should be selected in such a way that the environment is properly explored and penalties are avoided. It is not possible to complete these two tasks at the same time and a balance must be established between exploration and exploitation. One of the simplest methods of action selection is the greedy policy, in which the action with the highest value is chosen in each state. Adopting a greedy policy for action selection can limit the agent to a small segment of the environment space and prevent exploration of other segments to find better solutions. Therefore, other action selection methods are used in practice.

The Greedy with Optimistic Initialization: This method, like the greedy method, selects larger action-values in each state, but the initial value of all action-values is determined optimistically. Initial action values used as a way to increase exploration and all actions are tried several times. This method is suitable for stationary problems, but not effective for nonstationary problems [3].

The ε-Greedy Method: This policy determines whether to explore or exploit based on a specified threshold value *ε*. In the ε-greedy method, with a probability of 1-*ε* (where ε is a positive real number between 0 and 1), the action with the highest value is selected (Exploitation phase), and with a probability of *ε*, all actions can be chosen (Eexploration phase). The results indicate that this method outperforms the greedy algorithm in terms of efficiency [1].

The Upper Confidence Bound (UCB): In this method, unlike ε-greedy, non-greedy actions are selected based on their potential for being optimal. One of the ways to select the action is in the form of Equation (6) [3].

$$A_t = \arg\max_a [Q_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}}] \quad (6)$$

Where $\ln t$ denotes the natural logarithm of *t*, $N_t(a)$ represents the count of how many times action *a* has been chosen before time *t*, and the positive constant (*c > 0*) regulates the level of exploration. The second term represents the degree of uncertainty associated with the estimated value of *a* [3]. The UCB algorithms keep an upper confidence bound for each action, ensuring that the expected reward for each action is, with high probability, lower than this bound. At every time step, the agent optimistically chooses the action with the highest UCB [16].

The Soft-Max Method: In this method, Equation (7) is used for action selection.

$$P(s,a) = \frac{\exp(\frac{Q(s,a)}{T})}{\sum_{b\in A}\exp(\frac{Q(s,b)}{T})} \quad (7)$$

In Equation (7), the temperature coefficient (*T*) is considered large at the beginning of learning to strike a balance between exploration and exploitation. However, this coefficient decreases as the learning process progresses in order to use the previous experience. In the implementation of this method, the mechanism of the roulette wheel can be used and by creating a random number in the range of 0 to 1 and checking its position in the roulette wheel, the selected action can be determined [1].

The authors of [17] made conventional ε-greedy method adaptive by using the temporal difference error obtained from the value function of states. The main idea is that when the changes in the value function are high, more exploration should be done, and on the contrary, when the changes in the value function are small or unchanged, less exploration is needed. In each state, there is a separate ε that is updated after each action. At the beginning of the learning process, ε is initialized by 1 for all states. The idea of using differences of values was then integrated with the soft-max method, something which yielded favorable results [18].

Fuzzy logic has been used to create a balance between exploration and exploitation. An adaptive learning rate was introduced to strike a balance between exploration and exploitation in the fuzzy reinforcement learning [19]. This learning rate is set by considering the “fuzzy visit” of the current status.

The cuckoo search algorithm was adopted for action selection [4]. In this algorithm, the action selection problem is framed as an optimization challenge, where the candidate solutions are represented by Q values. The objective function is also the Q function. During each iteration of the Q-learning algorithm, the cuckoo search algorithm refines the combinations of Q values and actions, enabling the selection of the action that corresponds to the highest optimized Q value. This algorithm was implemented and tested on different problems such as the MAB and cliff-walking to determine its efficiency.

Analyzing the MAB problem, another study [20] aimed to find the optimal temperature in the soft-max method. A high temperature parameter will promote exploration, whereas a low temperature parameter will focus solely on exploitation. The temperature was considered constant in the soft-max equation. An evaluation function was first proposed to measure temperature, and the optimal temperature was then determined in an iterative process. The results of this method indicated improvements in the entire set of receivable rewards.

The process of identifying the optimal policy in Q-learning is reconfigured into a search for the best solution within an optimization problem [21]. In this method, the value of ε is generated randomly in each state and Simulated Annealing (SA) was used in ε-greedy. The probability of choosing each action in the current state is determined based on the SA algorithm, and the balance between exploration and exploitation can be controlled with the help of the temperature parameter. The efficiency of this idea was tested on a maze problem. It proved to be superior to ε-greedy and Boltzmann exploration.

The probabilistic action selection method is also provided [22]. The Probabilistic Q-Learning (PQL) enhances the exploration strategy by employing a probabilistic action selection method. This method produces a probability distribution for the action set corresponding to each state, guiding the exploration policy. Here, action a_i in state s is selected with probability $p(s, a_i)$, which probability is updated along with the value function updating and actions will be selected based on a variable probability distribution. In addition, in this paper fidelity-based PQL is introduced which extracts more information from the structure and behavior of the system for a better balance between exploration and exploitation.

Moreover, heuristic functions were employed in the action selection process [23]. In this study and other similar cases, instead of selecting an action only based on values or action-values, this work is done with the help of a combination of heuristic functions and action-values.

The paper [24] presents an innovative reinforcement learning algorithm known as "Go-Explore." This approach tackles the issue of exploration in environments with sparse rewards by initially returning to states that have already been discovered before venturing out from those points. This strategy effectively addresses the shortcomings of conventional exploration methods, which frequently encounter difficulties in complex or misleading environments. The authors illustrate that Go-Explore delivers state-of-the-art results across a range of challenging tasks, such as Atari games and robotic control, underscoring its potential to propel advancements in artificial intelligence and machine learning.

The paper [25] introduces an innovative approach to reinforcement learning that utilizes intrinsic rewards based on masked input modeling. The authors present the MIMEx framework, which boosts an agent's learning capabilities by enabling it to predict masked segments of its input data. This strategy fosters enhanced exploration and comprehension of the environment. The findings of the research indicate that this method significantly improves performance across a range of tasks, underscoring the value of intrinsic rewards in facilitating effective learning processes.

The authors of [14] explored the application of the chaos theory in dynamic programming, specifically using the logistic chaos system to overcome global updates of all states. This method involves executing policy evaluation once in each stage of policy iteration and updating only a few states proposed by the chaotic system. The policy improvement stage then follows, using similar procedures in the value iteration method, which resulted in better outputs than the conventional method.

4. Proposed Method

As mentioned in the introduction, by using numbers generated by chaotic systems instead of random numbers, the exploration process can be improved while maintaining exploitation efficiency. The ε-greedy method is a popular action selection approach that can benefit from the chaotic characteristic of generated numbers to strike a more appropriate balance between exploration and exploitation. Previous studies have shown that using chaotic maps can result in better exploration-exploitation balances in maze environments, although they only analyzed two or three chaotic maps. This study aims to use several chaotic maps in different environments, such as the Multi-Armed Bandit (MAB) and OpenAI Gym environments [26], and compare their efficiencies in action selection. After preprocessing and initialization, the numbers generated by different chaotic maps replace random numbers in the ε-greedy action selection process. The chaotic ε-greedy method can be expressed as Equation (8). Instead of choosing the desired action based on uniform random numbers, this is done using chaos. Here, $Q(S, b)$ refers to the value of the action-value in state S, and

$P(S,a)$ refers to the probability of selecting action a in state S.

$$P(S,a)=\begin{cases}1-\varepsilon & a=\arg\max_{b\in A}Q(S,b)\\ \frac{\varepsilon}{N} & otherwise\end{cases} \tag{8}$$

In this relation, N represents the total number of actions, and A is the set of actions in state S. By generating a chaotic number, if this number is less than the value ε, one of the actions with a lower value is selected. If the generated chaotic number is greater than the value ε, the action with the highest value is selected. The parameter ε theoretically ranges between 0 and 1. In this paper, various values of ε are tested, and the results will be discussed in the following section.

In the soft-Max method, it is also possible to get help from the benefits of chaos by generating a chaotic number and choosing the action in the roulette wheel based on it; in the traditional Soft-Max method, this is done with the help of random number generation.

For instance, Table 1 illustrates the probabilistic scenario of selecting five candidate actions in a given problem.

Table 1. An example of the roulette wheel mechanism and its application in action selection. (n=1 to 5) [1].

$Q(s,a_n)$	$P(s,a_n)$	Action selection interval
-0.0446	0.2542	[0,0.2542]
-1.1493	0.0842	[0.2543,0.3384]
0.1709	0.3154	[0.3385,0.6538]
-0.3211	0.1928	[0.6539,0.8466]
-0.5498	0.1534	[0.8467,1]

The probability values in the second column of the table are computed based on action-value parameters using Equation 7 (T=1) and the third column is the cumulative sum of the second column. The first candidate action is selected if a chaotic number lies within the interval [0, 0.2542], while the second action is chosen if the chaotic number falls within the interval [0.2543, 0.3384]. This paper demonstrates that employing chaotic numbers, as opposed to random numbers, enhances the efficiency of the Softmax algorithm. In this paper, Q-learning, which is one of the famous reinforcement learning algorithms, is used.

5. Computational Results

The proposed method was implemented in Python on a Windows 8 computer with an Intel Core i5 processor and 4 GB of RAM. To evaluate its performance, we tested the method on three environments widely used in previous studies on the exploration-exploitation balance: the Multi-Armed Bandit (MAB) [3], the cliff-walking environment [26], and the taxi-domain environment [26].

The MAB is a simple yet important problem for analyzing the exploration-exploitation challenge. It involves N reward machines, each of which gives a random reward from a constant distribution when its lever is pushed. The objective is to estimate which machine gives more rewards and the specific number of iterations that result in the greatest reward possible. In this study, we considered 10 machines with 10 random means defined using a normal distribution with a mean of 0 and a variance of 1. Each lever was then selected to give a random reward to the mean of a designated machine, and a variance of 1 was given to the agent. We ran 2000 iterations and averaged the results from multiple executions.

The taxi-domain problem consists of several accessible OpenAI Gym environments [26]. The objective is for a taxi to pick up a passenger and drop them off at their destination via the shortest path possible. The environment is a 5x5 grid with 500 states and six actions: move up, move down, turn left, turn right, pick up the passenger, and drop off the passenger. The passenger gets in or out of the taxi in one of the four designated cells. Each move has a score of -1, dropping off the passenger at the destination has a score of +20, and picking up or dropping off the passenger at the wrong location has a score of -10. In the cliff-walking environment, the objective is for the agent to reach the target cell from the start cell. Each move has a score of -1, and hitting a cliff incurs a heavy penalty of -100, after which the agent returns to the start cell.

Table 2 presents the results of using different chaotic maps in the ε-greedy and Soft-Max methods for the MAB problem. In ε-greedy section, rewards were averaged from 2000 executions of 2000 steps for all values of ε between 0 and 1 with a step of 0.01. The table reports the highest reward value and its corresponding ε, as well as the highest mean of rewards received from the last 100 steps and the corresponding ε. For example, the entry for LEL [27] shows that the highest reward of 1.3846 was obtained at an ε value of 0.21, and the highest mean of rewards from the last 100 steps was obtained at an ε value of 0.21. Also, in the last two columns of Table 2, the results of using different chaotic systems in the Soft-Max method are also shown. The results are averaged over 2000 runs of 2000 steps. The initial temperature is equal to 1 and decreases at a rate of 0.9975.

In the first row of the table, the values of the traditional ε-greedy and Soft-Max methods are mentioned, which were obtained using random numbers.

The results show that most of the chaotic maps outperformed the conventional random method in terms of acquiring rewards, but the optimal ε values varied across different maps. Interestingly, in most cases, the optimal *ε* value for the chaotic maps was higher than that of the random method. This suggests that chaos performs better with a higher rate of exploration, which enhances the chances of discovering optimal solutions. Figure 2 illustrates the rewards acquired for all values of *ε* in four chaotic maps compared to the random method. The figure shows that chaotic methods consistently outperformed the conventional method at most values of *ε*. For example, the SINGER map [28] achieved significantly higher rewards in most values of *ε* compared to the conventional method.

As expected, the results for Soft-Max are better than those for ε-greedy; and the highest rewards are related to the use of chaos in the Soft-Max method. Also, the ε-greedy method has improved more than the Soft-Max method under the influence of chaos.

Tables 3 and 4 report the results of implementing the ε-greedy action selection method in the taxi-domain and cliff-walking problems.

The Q-learning method was used in these problems, where the learning rate and *γ* were both set to 0.9. The value of *ε* was first set to 1, which gradually decreased. The results of the taxi-domain problem included 500 episodes, in each of which the maximum number of steps was 99. To obtain more realistic results, 100 different runs were tested, and their means were then reported on tables. The cliff-walking problem had 400 episodes, in each of which the maximum number of steps was 99. There were also 500 different runs. In addition to the mean of the total reward and the mean of rewards from the last 100 steps, Table 4 indicates the number of failures (encounters with cliffs) in the cliff-walking problem. According to tables 3 and 4, this idea has managed to outperform the random state on many chaotic maps. In the taxi-domain problem, the mean of rewards from the last 100 episodes was nearly -5.5 in the conventional method. However, it was nearly +4 in the best chaos, i.e., LEL (in this problem).

Table 2. The results of using the chaotic action selection method in the MAB problem: $R_{avg} = \frac{r_{t=1}+\dots+r_{t=2000}}{2000}$ **and** $R_{100} = \frac{r_{t=1901}+\dots+r_{t=2000}}{100}$ **.**

Chaotic map	Maximum R_{avg} in ε-greedy	Maximum R_{100} in ε-greedy	R_{avg} in Soft-Max	R_{100} in Soft-Max
conventional random method	1.3607 (ε=0.06)	1.4315 (ε=0.04)	1.4469	1.5172
Logistic [14]	1.3739 (ε=0.12)	1.4516 (ε=0.06)	1.3857	1.5458
Sine [26]	1.3738 (ε=0.06)	1.4600 (ε=0.06)	1.3864	1.5388
Tinkerbell [27]	**1.3915 (ε=0.17)**	1.4628 (ε=0.1)	1.4376	1.5203
Zaslavskii [28]	1.3866 (ε=0.17)	1.4579 (ε=0.13)	**1.4555**	1.5385
Zhang [29]	1.3679 (ε=0.17)	1.4498 (ε=0.08)	1.3054	1.4475
HyperChaos [30]	1.3605 (ε=0.27)	1.4525 (ε=0.23)	1.3762	1.4347
SPL [15]	1.3638 (ε=0.16)	1.4706 (ε=0.16)	1.3495	1.4733
Chen [13]	1.3684 (ε=0.18)	1.4557 (ε=0.16)	1.4220	1.4904
Cubic [28]	1.3735 (ε=0.06)	1.4669 (ε=0.06)	1.3944	1.5400
ICMIC [31]	1.3620 (ε=0.01)	1.4298 (ε=0.01)	1.3599	1.5353
Lorenz [12]	1.3692 (ε=0.21)	1.4552 (ε=0.21)	1.4102	1.4798
Piecewise [11]	1.3599 (ε=0.05)	1.4413 (ε=0.03)	1.4284	1.5402
Kent [27]	1.3721 (ε=0.04)	1.4533 (ε=0.04)	1.4271	1.5387
Singer [26]	1.3845 (ε=0.12)	**1.4753 (ε=0.08)**	1.4093	1.5395
Singh [32]	1.3654 (ε=0.23)	1.4508 (ε=0.19)	1.4179	1.4867
Nguyen [33]	1.3749 (ε=0.19)	1.4635 (ε=0.19)	1.4180	1.4855
Circle [34]	1.3693 (ε=0.04)	1.4553 (ε=0.04)	1.4255	1.5403
Modified Logistic [35]	1.3911 (ε=0.08)	1.4689 (ε=0.08)	1.3784	1.5353
Bernoulli Shift [11]	1.3653 (ε=0.04)	1.4475 (ε=0.04)	1.4155	1.5263
LEL [25]	1.3846 (ε=0.21)	1.4660 (ε=0.15)	1.3893	**1.5482**
Tent [26]	1.3737 (ε=0.03)	1.4700 (ε=0.03)	1.4165	1.5312
Iterative [26]	1.3718 (ε=0.04)	1.4458 (ε=0.01)	0.9551	1.1226
Leibovitch [11]	1.3688 (ε=0.06)	1.4398 (ε=0.02)	1.4072	1.5286
Arnold Cat [36]	1.3668 (ε=0.17)	1.4624 (ε=0.11)	1.4441	1.5254
SEL [25]	1.3673 (ε=0.14)	1.4514 (ε=0.12)	1.3812	1.5262

In the cliff-walking problem, the mean of rewards from the last 100 episodes was nearly -62.3 in the random state. However, it was nearly -16.9 in the best chaos, i.e., HyperChaos. Evidently, the difference was 45 between the received rewards.

For instance, Figures 3 and 4 demonstrate the comparative charts of some chaotic maps in these two problems. In these figures, the average award obtained in consecutive episodes is shown.

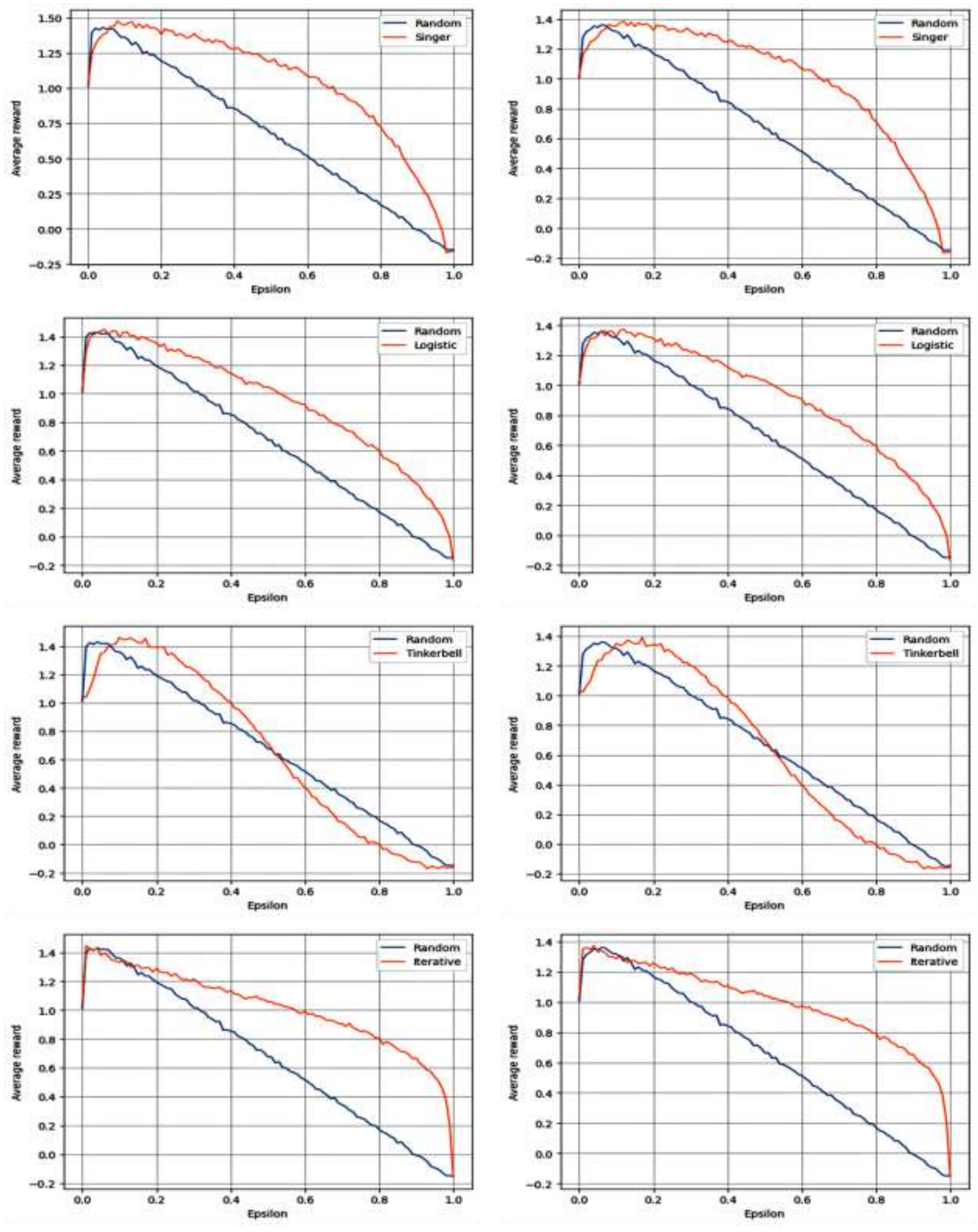

Figure 2. The results of using the ε-greedy method in four chaotic maps: Left and Right respectively: R_{100} and R_{avg} for different values of ε.

According to the results and tests, the recently proposed chaotic maps such as LEL, modified logistic, and Zhang yielded much higher efficiencies; hence, they are recommended for similar problems. These maps are usually the integrated or developed versions of previous maps.

In addition, the chaotic systems that generate more than one number in each iteration can also yield better outputs (e.g., Nguyen, Tinkerbell, and Singh), something which can be due to averaging and using their generated numbers.

Table 3. Comparative results of chaotic and random methods in the taxi-domain environment:
$R_{avg} = \frac{r_{e=1}+...+r_{e=500}}{500}$ **and** $R_{100} = \frac{r_{e=401}+...+r_{e=500}}{100}$.

Chaotic map	R_{avg}	R_{100}
conventional random method	-153.4948	-5.5191
Logistic	-103.8618	1.4886
Sine	-112.1734	0.2252
Tinkerbell	-147.9634	1.8507
Zaslavskii	-138.6673	2.0033
Zhang	**-85.0984**	3.7985
HyperChaos	-150.1541	2.8764
SPL	-131.0355	3.8396
Chen	-156.9166	2.1884
Cubic	-109.3988	0.6909
ICMIC	-135.9574	-11.6701
Lorenz	-150.2218	2.6355
Piecewise	-152.6548	-4.9568
Kent	-152.8995	-5.4057
Singer	-89.4994	3.3733
Singh	-150.4401	2.8554
Nguyen	-150.4119	2.8875
Circle	-166.8422	-5.4243
Modified Logistic	-103.8775	1.3778
Bernoulli Shift	-152.578	-5.0860
LEL	-99.0054	**3.9206**
Tent	-152.2878	-4.9191
Iterative	-134.0724	-28.1704
Leibovitch	-169.2107	-8.4864
Arnold Cat	-150.7027	1.5231
SEL	-107.0654	3.52

6. Discussion

The results of this research show that replacing random numbers with chaotic numbers significantly improves the performance of the learning algorithm in many chaotic systems; This is due to the special characteristics of chaos, especially its ergodicity.

According to the successful experience of chaos in many applications, in general, replacing random numbers with chaotic numbers is desirable in most cases, but one of the major weaknesses of chaos is the diversity and extent of these systems, as well as the somewhat different behavior of chaotic systems in Different environments [11]. For example, in the paper on the use of chaos in cellular automata [11], Tent's chaotic system has a good performance, but this is not the case in the tests of this research.

This work does not propose a new method nor claims superiority over existing well-established approaches. In reinforcement learning, common action selection techniques such as ε-greedy and Softmax rely on random numbers for decision-making. To improve their efficiency, chaos-based mechanisms were introduced, which demonstrated promising results. Extensive comparisons were carried out across multiple environments to evaluate their performance. Beyond ε-greedy and Softmax, which are the focus of this study, random and greedy action selection methods were also examined. However, under the specified conditions in this paper, these methods failed to achieve convergence, leaving the problem unresolved.

Table 4. Comparative results of chaotic and random methods in the cliff-walking environment:
$R_{avg} = \frac{r_{e=1}+...+r_{e=400}}{400}$ **and** $R_{100} = \frac{r_{e=301}+...+r_{e=400}}{100}$.

Chaotic map	Average number of failures	R_{avg}	R_{100}
conventional random method	309.316	-89.7255	-62.3534
Logistic	239.064	-73.4141	-42.7098
Sine	258.076	-77.9455	-48.1701
Tinkerbell	254.162	-75.7387	-32.5329
Zaslavskii	247.93	-74.4726	-33.6996
Zhang	**182.522**	**-59.7008**	-29.7584
HyperChaos	215.1	-65.7251	**-16.9275**
SPL	273.214	-81.4399	-33.3061
Chen	242.728	-72.5083	-24.2114
Cubic	250.414	-76.1308	-46.0402
ICMIC	316.036	-92.1133	-73.6471
Lorenz	231.458	-69.8922	-22.3126
Piecewise	309.734	-89.8002	-62.7174
Kent	309.824	-89.8037	-62.6652
Singer	190.892	-61.6546	-31.6921
Singh	231.806	-69.9747	-22.4358
Nguyen	232.06	-70.0131	-22.3273
Circle	325.596	-93.5226	-65.0926
Modified Logistic	238.512	-73.2755	-42.4887
Bernoulli Shift	309.204	-89.7047	-62.2438
LEL	210.214	-66.3543	-30.0920
Tent	309.158	-89.7160	-62.2757
Iterative	323.816	-94.4511	-83.7766
Leibovitch	331.028	-94.8541	-69.9228
Arnold Cat	257.56	-76.5054	-34.1446
SEL	230.328	-71.1815	-34.4375

7. Conclusion

In this paper, the method of using chaos in choosing random actions in reinforcement learning is presented, and a comprehensive review on the effectiveness of different types of chaos functions (maps) in action selection has been done. The paper replaces the random numbers with numbers generated by different chaotic systems in the ε-greedy and Soft-Max action selection methods. The study compares the results of different chaotic methods with the conventional method in the MAB, taxi-domain, and cliff-walking environments. The results show that using many chaotic methods leads to greater rewards and fewer steps towards a target. The study also finds that chaotic methods with greater rewards have larger values of ε than the conventional method. Additionally, chaotic methods obtain greater rewards even at large values of ε compared to the conventional method. Chaotic methods outperformed the conventional method in terms of receivable rewards in the taxi-domain and cliff-walking environments. The study indicates the superiority of more recent chaotic systems (e.g., LEL and modified logistic) and multidimensional systems (e.g., HyperChaos and Singh) than other systems.As the next step, the use of chaotic systems in the action selection part of deep reinforcement learning algorithms is considered.

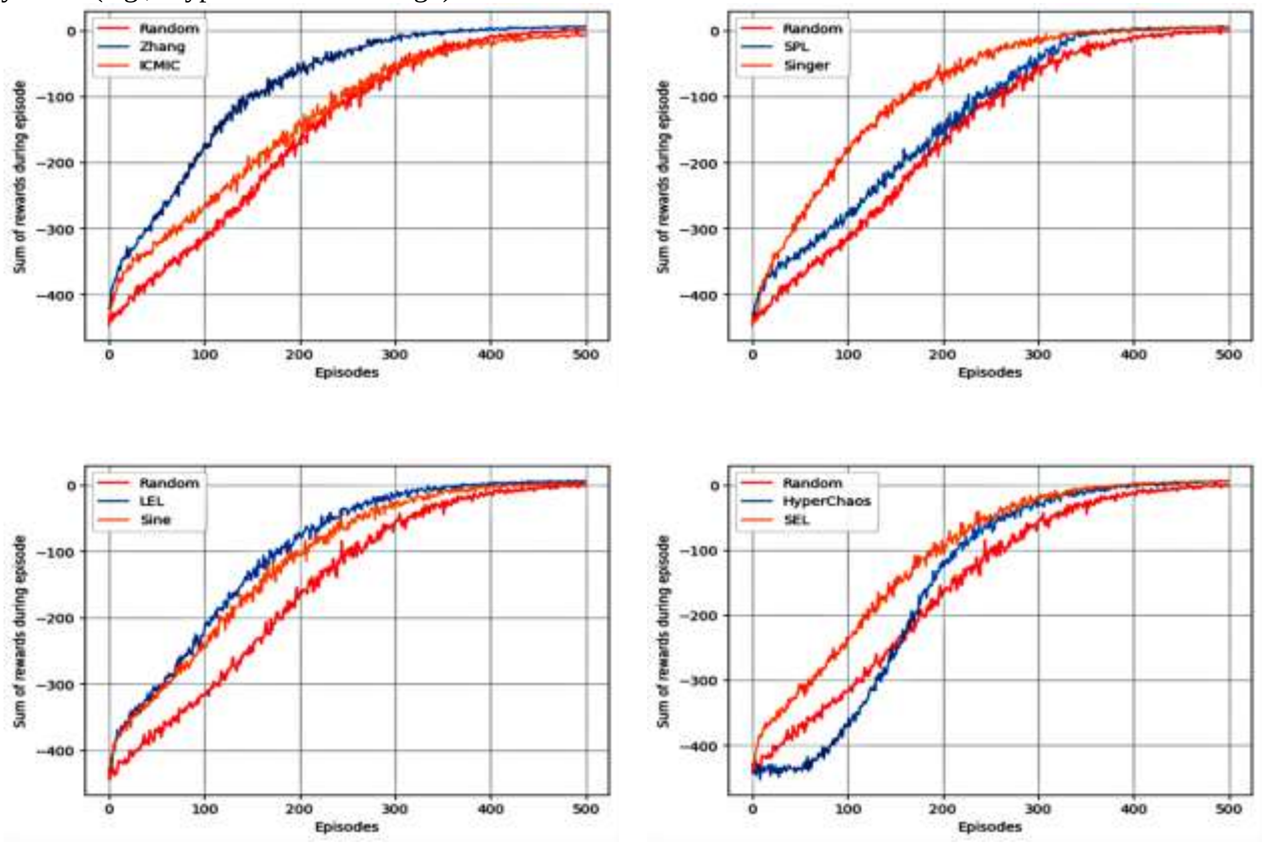

Figure 3. Comparing the rewards received from some chaotic maps with those of the random method in the taxi-domain environment.

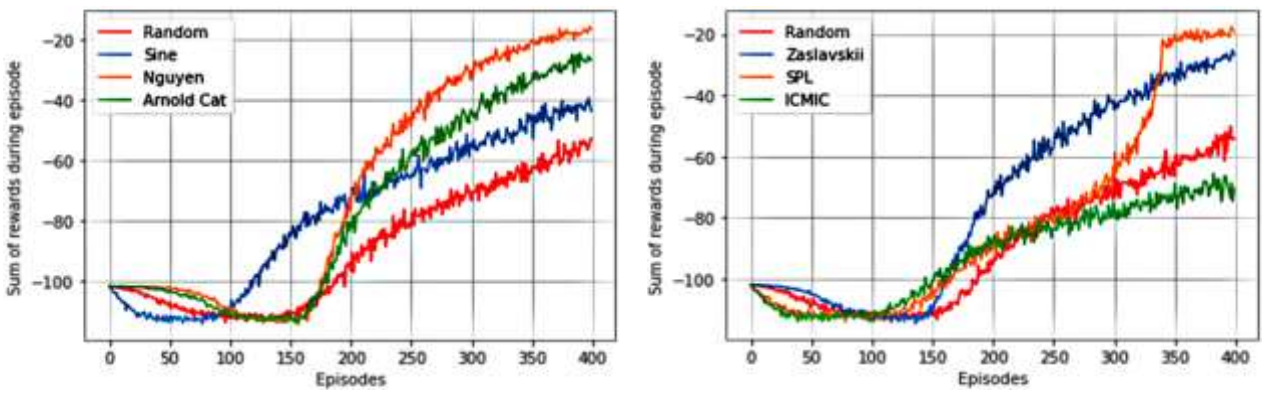

Figure 4. Comparing the rewards received from some chaotic maps with those of the random method in the cliff-walking environment.

References

[1] V. Derhami, F. Alamian Harandi and M. B. Dowlatshahi, *Reinforcement Learning*. Yazd, Iran, Yazd University Press, 2017.

[2] F. Alamiyan-Harandi, V. Derhami and F. Jamshidi, “A new framework for mobile robot trajectory tracking using depth data and learning algorithms”, *Journal of Intelligent & Fuzzy Systems*, vol. 34, no. 6, pp. 3969-3982, 2018.

[3] RS. Sutton and AG. Barto, Reinforcement *learning: An introduction*. 2nd Ed, London, The MIT Press, 2018.

[4] BH. Abed-alguni, “Action-selection method for reinforcement learning based on cuckoo search algorithm”, Arabian *Journal for Science and Engineering*, vol. 43, no. 12, pp. 6771-6785, 2018.

[5] K. Morihiro, T. Isokawa, N. Matsui and H. Nishimura, “Effects of chaotic exploration on reinforcement learning in target capturing task”,

International Journal of Knowledge-based and Intelligent Engineering Systems, vol. 12, no. 5-6, pp. 369-377, 2008.

[6] K. Morihiro, T. Isokawa, N. Matsui and H. Nishimura, "Reinforcement learning by chaotic exploration generator in target capturing task", *proc. International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, Springer Berlin Heidelberg, 2005, pp. 1248-1254.

[7] K. Morihiro, N. Matsui and H. Nishimura, "Effects of chaotic exploration on reinforcement maze learning", *Proc. International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, Springer Berlin Heidelberg, 2004, pp. 833-839.

[8] K. Morihiro, N. Matsui and H. Nishimura, "Chaotic exploration effects on reinforcement learning in shortcut maze task", *International Journal of Bifurcation and Chaos*, vol. 16, no. 10, pp. 3015-3022, 2006.

[9] AB. Potapov and MK. Ali, "Learning, exploration and chaotic policies", *International Journal of Modern Physics C,* vol. 11, no.07, pp. 1455-1464, 2000.

[10] E. Pei, J. Jiang, L. Liu, Y. Li and Z. Zhang, "A chaotic Q-learning-based licensed assisted access scheme over the unlicensed spectrum", *IEEE Transactions on Vehicular Technology,* vol. 68, no. 10, pp. 9951-9962, 2019.

[11] B. Zarei and MR. Meybodi, "Improving learning ability of learning automata using chaos theory", *The Journal of Supercomputing,* vol. 77, no. 1, pp. 652-678, 2021.

[12] EN. Lorenz, "Deterministic nonperiodic flow", *Journal of atmospheric sciences,* vol. 20, no. 2, pp. 130-141, 1963.

[13] G. Chen and T. Ueta, "Yet another chaotic attractor", *International Journal of Bifurcation and chaos,* vol. 9, no. 07, pp. 1465-1466, 1999.

[14] H. Khodadadi and V. Derhami, "Improving Speed and Efficiency of Dynamic Programming Methods through Chaos", *Journal of AI and Data Mining,* vol. 9, no. 4, pp. 487-496, 2021.

[15] M. Mollaeefar, A. Sharif and M. Nazari, "A novel encryption scheme for colored image based on high level chaotic maps", *Multimedia Tools and Applications,* vol. 76, pp. 607-629, 2017.

[16] RY. Chen, J. Schulman, P. Abbeel and S. Sidor, "UCB and infogain exploration via q-ensembles", arXiv:1706.01502, 2017.

[17] M. Tokic, "Adaptive ε-greedy exploration in reinforcement learning based on value differences", *proc. Annual Conference on Artificial Intelligence, Springer Berlin Heidelberg,* 2010, pp. 203-210.

[18] M. Tokic and G. Palm, "Value-difference based exploration: adaptive control between epsilon-greedy and softmax". *proc. Annual conference on artificial intelligence*, Berlin, 2011, pp. 335-346.

[19] V. Derhami, V. Johari Majd, MN. Ahmadabadi, "Exploration and exploitation balance management in fuzzy reinforcement learning", *Fuzzy sets and systems,* vol. 161, no. 4, pp. 578-595, 2010.

[20] YL. He, XL. Zhang, W. Ao and JZ. Huang, "Determining the optimal temperature parameter for Softmax function in reinforcement learning", *Applied Soft Computing,* vol. 70, pp.80-85, 2018.

[21] M. Guo, Y. Liu and J. Malec, "A new Q-learning algorithm based on the metropolis criterion", IEEE *Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics),* vol. 34, no. 5, pp. 2140-2143, 2004.

[22] C. Chen, D. Dong, HX. Li, J. Chu and TJ. Tarn, "Fidelity-based probabilistic Q-learning for control of quantum systems", *IEEE transactions on neural networks and learning systems,* vol. 25, no. 5, pp. 920-933, 2013.

[23] RA. Bianchi, CH. Ribeiro CH and AHR. Costa, "Heuristically Accelerated Reinforcement Learning: Theoretical and Experimental Results", *proc. 20th European Conference on Artificial Intelligence (ECAI),* IOS Press, 2012, pp. 169-174.

[24] A. Ecoffet, J. Huizinga, J. Lehman, K.O. Stanley and J. Clune, "First return, then explore", *Nature,* vol. 590, no.7847, pp. 580-586, 2021.

[25] T. Lin and A. Jabri, "MIMEx: intrinsic rewards from masked input modeling", arXiv preprint arXiv:2305.08932, 2023.

[26] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang and W. Zaremba, "OpenAI Gym", ArXiv:1606.01540, 2016.

[27] Z. Hua and Y. Zhou, "Exponential chaotic model for generating robust chaos", *IEEE transactions on systems, man, and cybernetics,* vol. 51, no. 6, pp. 3713-3724, 2019.

[28] AH. Gandomi and XS. Yang," Chaotic bat algorithm", *Journal of computational science,* vol. 5, no. 2, pp. 224-232, 2014.

[29] Jr I. Fister, M. Perc, SM. Kamal and I. Fister, "A review of chaos-based firefly algorithms: perspectives and research challenges", *Applied Mathematics and Computation,* vol. 252, pp. 155-165, 2015.

[30] H. Lu, X. Wang, Z. Fei and M. Qiu, "The effects of using chaotic map on improving the performance of multi objective evolutionary algorithms", *Mathematical Problems in Engineering,* no. 1, Article ID 924652, 2014.

[31] X. Zhang and Y. Cao, "A novel chaotic map and an improved chaos-based image encryption scheme", *The Scientific World Journal,* no. 1, Article ID 713541, 2014.

[32] C. Zhu, "A novel image encryption scheme based on improved hyperchaotic sequences", *Optics communications,* vol. 285, no. 1, pp. 29-37, 2012.

[33] A. Rezaee Jordehi, "A chaotic artificial immune system optimization algorithm for solving global continuous optimization problems", *Neural Computing and Applications*, vol. 26, pp. 827-833, 2015.

[34] PP. Singh, "A chaotic system with large Lyapunov exponent: Nonlinear observer design and circuit implementation", *In 2020 3rd international conference on energy, power and environment: Towards clean energy technologies*, 2021, pp. 1-6.

[35] N. Nguyen, L. Pham-Nguyen, MB. Nguyen and G. Kaddoum, "A low power circuit design for chaos-key based data encryption", *IEEE Access,* vol. 8, pp. 104432-104444, 2020.

[36] KZ. Zamli, F. Din, HS. Alhadawi, "Exploring a Q-learning-based chaotic naked mole rat algorithm for S-box construction and optimization", *Neural Computing and Applications,* vol. 35, no. 14, pp. 10449-10471, 2023.

[37] L. Moysis, A. Tutueva, C. Volos, D. Butusov, JM. Munoz-Pacheco, H. Nistazakis, "A two-parameter modified logistic map and its application to random bit generation", *Symmetry*, vol. 12, no. 5: 829, 2020.

[38] L. Skanderova, I. Zelinka, "Arnold cat map and sinai as chaotic numbers generators in evolutionary algorithms", In *AETA 2013: Recent Advances in Electrical Engineering and Related Sciences*, 2014, pp. 381-389.

8. Appendix

Here the chaos maps worked in this paper are listed along with the desired parameters.

Chaotic System	Equation	Designated Parameters
Logistic[14]	$x_{n+1} = \lambda x_n(1 - x_n)$	$\lambda = 4.0$
Sine[28]	$x_{n+1} = \frac{a}{4}\sin(\pi x_n)$	$a = 4.0$
Tinkerbell [29]	$\begin{cases} x_{n+1} = x_n^2 y_n^2 + ax_n + by_n \\ y_{n+1} = 2x_n y_n + cx_n + dy_n \end{cases}$	$a = 0.9\,, b = 0.6013, c = 2.0$ and $d = 0.5$
Zaslavskii [30]	$\begin{cases} x_{n+1} = (x_n + v + ay_{n+1}) \bmod 1 \\ y_{n+1} = \cos(2\pi x_n) + e^{-r} y_n \end{cases}$	$v = 400\,, r = 3$ and $a = 12.6695$
Zhang [31]	$x_{n+1} = 2\mu\|x_n\|(1 - 2\|x_n\|)$	$\mu = 2.4140$
HyperChaos [33]	$\begin{cases} \dot{x} = a(y - x) + yz \\ \dot{y} = cx - y - xz + w \\ \dot{z} = xy - bz \\ \dot{w} = dw - xz \end{cases}$	$a = 35\,, b = 3, c = 28.0$
SPL [15]	$x_{n+1} = \sin(r \times \arcsin(\sqrt{\|x_n\|}))^2 + (1 - r)2(\|x_n\|)(1 - 2\|x_n\|)$	$r = 3.465$
Chen [13]	$\begin{cases} \dot{x} = a(y - x) \\ \dot{y} = (c - a)x + cy - xz \\ \dot{z} = xy - bz \end{cases}$	$a = 35\,, b = \frac{8}{3}, c = 55$ and $d = 1.3$
Cubic [30]	$x_{n+1} = \rho x_n(1 - x_n^2)$	$\rho = 2.59$
ICMIC [33]	$x_{n+1} = \sin(\frac{a}{x_n})$	$a = 7.0$
Lorenz [12]	$\begin{cases} \dot{x} = a(y - x) \\ \dot{y} = bx - y - xz \\ \dot{z} = xy - cz \end{cases}$	$a = 10.0\,, b = \frac{8}{3}, c = 28.0$
Piecewise [11]	$x_{n+1} = \begin{cases} \frac{x_n}{p} & 0 \le x_n < p \\ \frac{x_n - p}{0.5 - p} & p \le x_n < 0.5 \\ \frac{1 - p - x_n}{0.5 - p} & 0.5 \le x_n < 1 - p \\ \frac{1 - x_n}{p} & 1 - p < x_n \le 1 \end{cases}$	$p = 0.4$
Kent [29]	$x_{n+1} = \begin{cases} \frac{x_n}{m} & 0 < x_n \le m \\ \frac{1 - x_n}{1 - m} & m < x_n \le 1 \end{cases}$	$m = 0.4$
Singer [28]	$x_{n+1} = \mu(7.86x_n - 23.31x_n^2 + 28.75x_n^3 - 13.3x_n^4)$	$\mu = 1.05$
Singh [34]	$\begin{cases} \dot{x} = ax - y^2 \\ \dot{y} = b(z - y) \\ \dot{z} = cy + xy - z \end{cases}$	$a = -2.667\,, b = 10, c = 27.3$
Nguyen [35]	$\begin{cases} \dot{x} = -(y - z) \\ \dot{y} = -z \\ \dot{z} = -a(x - y + z) + b \times \tanh(cx - d) \end{cases}$	$a = 0.3\,, b = 0.1, c = 80$ and $d = 0.7$
Circle [36]	$x_{n+1} = \{x_n + b - (a - 2\pi)\sin(2\pi x_n)\} \bmod 1$	$b = 0.2\,, a = 0.5$
Modified Logistic [37]	$x_{n+1} = r(x_n \bmod 1)(1 - (x_n \bmod 1))(2\beta - \frac{x_n^2}{\beta})$	$\beta = 40$ and $r = 4$
Bernoulli Shift [11]	$x_{n+1} = \begin{cases} \frac{x_n}{p} & 0 \le x_n \le p \\ \frac{x_n - p}{1 - p} & p < x_n \le 1 \end{cases}$	$p = 0.9$

Iterative [28]	$x_{n+1} = \sin(\frac{a\pi}{x_n})$	$a = 0.4$
Leibovitch [11]	$x_{n+1} = \begin{cases} ax_n & 0 < x_n \le p_1 \\ \frac{p_2 - x_n}{p_2 - p_1} & p_1 < x_n \le p_2 \\ 1 - b(1 - x_n) & p_2 < x_n \le 1 \end{cases}$ $a = \frac{p_2}{p_1}\left(1 - (p_2 - p_1)\right)$ $b = \frac{1}{p_2 - 1}(p_2 - 1 - p_1(p_2 - p_1))$	$p_1 = 0.4$ and $p_2 = 0.8$
LEL [27]	$x_{n+1} = (4ax_n(1 - x_n))^{\ln(4bx_n(1-x_n)+c)}$	$a = 1$, $b = 1$ and $c = 2.0$
SEL [27]	$x_{n+1} = (a\sin(\pi x_n))^{\ln(4bx_n(1-x_n)+c)}$	$a = 1$, $b = 1$ and $c = 2.0$
Tent [28]	$x_{n+1} = \begin{cases} \frac{x_n}{0.7} & x_n < 0.7 \\ \frac{10}{3}(1 - x_n) & x_n \ge 0.7 \end{cases}$	-
Arnold Cat [38]	$\begin{cases} x_{n+1} = (x_n + y_n) \bmod 1 \\ y_{n+1} = (x_n + ky_n) \bmod 1 \end{cases}$	$k = 2$

استفاده از نظریه آشوب برای ایجاد تعادل بین کاوش و بهره‌گیری در یادگیری تقویتی

حبیب خدادادی[۱] و ولی درهمی[۲،*]

[۱] گروه مهندسی کامپیوتر، واحد میناب، دانشگاه آزاد اسلامی، میناب، ایران.

[۲] دانشکده مهندسی کامپیوتر، دانشگاه یزد، یزد، ایران.

ارسال ۲۰۲۴/۱۲/۲۱؛ بازنگری ۲۰۲۵/۰۲/۱۰؛ پذیرش ۲۰۲۵/۰۲/۱۸

چکیده:

تعادل بین کاوش و بهره‌گیری یکی از مهم‌ترین مسائل در یادگیری تقویتی است و به این منظور به جای انتخاب عمل حریصانه از روش‌های انتخاب عمل دیگری همچون شبه‌حریصانه و بیشین نرم استفاده می‌شود. در این روش‌ها به کمک تولید اعداد تصادفی، عملی انتخاب می‌شود که بتواند این تعادل را برقرار کند؛ به همین دلیل می‌توان انتظار داشت که اعمال با ارزش بیشتری شناسایی شود و محیط بهتر شناخته شود. آشوب با داشتن ویژگی‌هایی همچون حساسیت زیاد به شرایط اولیه، غیر تناوبی، غیر قابل پیش‌بینی، ارگودیسیته و رفتار شبه تصادفی، دارای کاربردهای فراوانی در بسیاری از علوم است. در این مقاله، جهت ایجاد تعادل بهتر بین کاوش و بهره‌گیری، به جای تولید سنتی اعداد تصادفی یکنواخت، از اعداد تولیدی توسط سیستم‌های آشوبناک مختلف جهت استفاده در روش‌های انتخاب عمل شبه‌حریصانه و بیشینه نرم بهره‌گیری می‌شود و نگاشت‌های بهتر در حالات و تعداد اعمال مختلف شناسایی می‌شوند. آزمایش‌های انجام شده در چند محیط مورد بررسی شامل راهزن چند دست، تاکسی و صخره، نشان-دهنده افزایش سرعت و کسب جایزه بیشتر در خیلی از نگاشت‌های آشوبی است.

کلمات کلیدی: انتخاب عمل، تئوری آشوب، کاوش و بهره‌گیری، یادگیری تقویتی.