Q-Learning based Metaheuristic Optimization Algorithms: A short review and perspectives

Qusay Hamad (qusay@student.usm.my)
Universiti Sains Malaysia - Engineering Campus Seri Ampangan: Universiti Sains Malaysia - Kampus Kejuruteraan Seri Ampangan https://orcid.org/0000-0002-8699-2586

Hussein Samma
Universiti Teknologi Malaysia

Shahrel Azmin Suandi
Universiti Sains Malaysia - Engineering Campus Seri Ampangan: Universiti Sains Malaysia - Kampus Kejuruteraan Seri Ampangan https://orcid.org/0000-0001-9980-7426

Research Article

Keywords: Optimization algorithms review, Swarm intelligence review, Reinforcement learning review, Q-Learning review, Particle swarm optimization review

Posted Date: January 5th, 2023

DOI: https://doi.org/10.21203/rs.3.rs-1950095/v1

Abstract

In recent years, reinforcement learning (RL) has garnered a great deal of interest from researchers because of its success in handling some complicated issues. Specifically, Q-learning as a model of RL is used a lot in various fields, and it has given an attractive result in games. In recent years, some researchers have tried to exploit the power of Q-learning to improve the results of optimization algorithms by guiding the optimization algorithm search agents based on the data saved in Q-table during the search process. The best search agent is chosen based on its accumulated performance, in other words, how well it has done overall, not how well it has done at each iteration. It is important to note that this review does not focus on reinforcement learning algorithms collaborating with metaheuristic optimization algorithms because there are so many reinforcement learning algorithms and to narrow the scope of the review, this paper will only discuss Q-learning used to enhance metaheuristic optimization algorithms. In this study will look at the huge progress made in the research community by looking at 32 different algorithms proposed on the subject from 2009 to 2022, with a focus on studies published in the last five years. As a result of the surveys conducted in this study, researchers (novices and experts) in the field of metaheuristic optimization algorithms research are expected to gain a better understanding of current research trends involving the use of Q-Learning and new motivations for outlining appropriate strategic plans for future development work as a result of the surveys conducted in this study.

Introduction

Optimization is the process of identifying the optimal solution for a given problem from among a large number of feasible alternatives. This method has been utilized in numerous fields, disciplines, and applications. Because there are numerous complex problems in the real world, the demand for more robust algorithms that can handle such problems is constantly rising. This necessitates robust algorithms that can handle the complexity of such real-world problems [1][2].

In recent years, numerous metaheuristic algorithms (MA) have been developed for a variety of real-world problems, including engineering design, clustering, image processing, data mining, etc. Most of the time, these methods are used when it won't be possible to solve a problem with enough time or accuracy using traditional methods [1]. MA is divided into four distinct classes based on its source of inspiration: swarm intelligence algorithms; physical and chemical nature algorithms; evolutionary nature algorithms; and human-based algorithms [3].

Existing MA have demonstrated their reliability in locating the best solution. However, it is not guaranteed that every metaheuristic can locate the optimal solution for all types of problems. This was made clear in the "no-free-lunch" (NFL) theory[4], which says that there is no one way of optimization that can solve all kinds of problems in the best way. In this sense, the NFL theory holds this research field open to the development of new optimization algorithms for solving complicated real-world issues [5][6][7].

Reinforcement learning happens in a dynamic setting where there is no supervisor to review the selected action at each level. The most successful strategy for maximizing predicted values or rewards at the conclusion of a learning exercise is trial and error interactions with the environment [8].

The global best search agent in the optimization method is determined by evaluating the performance of each agent at each iteration. As is well known, a spectacular performance in the moment does not guarantee future success. For instance, during the course of a game of chess, a player may gain multiple pieces over the course of several moves but ultimately lose the game. On the contrary, it is possible to purposefully sacrifice pieces in order to win the overall game. Therefore, cumulative performance may be a more accurate predictor of success. If this perspective is reasonable, then determining how to evaluate an agent's accumulated performance is an immediate issue that must be resolved. Using Q-learning to figure out each search agent's cumulative performance is one possible solution to this challenge. The researchers came up with new optimization methods based on Q-learning to get around the limitations of the original algorithms. Instead of looking at performance in the moment, they used accumulated performance to find the best search agent in the world [8].

Q-learning was first proposed by Watkins and Dayan [9] as a model of reinforcement learning (RL), a branch of machine learning concerned with how an agent might interact with its environment in different states. The agent decides which action to do based on the environment's reactions to prior encounters in order to maximize positive rewards. Because of its easy applicability, quick convergence rate, and inexpensive computing cost, the Q-learning model is widely utilized. Bayesian Q-learning [10], Double Q-learning [11], Deep Q-learning [12], and other RL models are utilized for more sophisticated parameter management.

Q-learning offers numerous benefits [13], such as:

1) No prior knowledge is necessary.

2) it does not require task completion.

3)It has a rather simple structure.

In recent years, the combination of Q-learning and optimization algorithms has garnered increasing interest. Q-learning can improve the performance of optimization algorithms because it can find the best long-term reward by trying out different ways to behave and is useful for keeping a balance between exploration and exploitation in a way that changes over time [13]. Figure 1 depicts the connection between Q-Learning and optimization algorithms.

The following is a list of the contributions made by this paper:

1. This is, as far as we are aware, the first article to summarize researches on the relationship between Q-Learning and optimization algorithms.

2. Recent advancements in the modification of metaheuristic optimization algorithms based on Q-learning, including innovations introduced in conjunction with Q-table and how Q-table controls search agents up to 2022, are described in detail and critically evaluated.

Metaheuristic Optimization Algorithms

Metaheuristics optimization algorithms are capable of avoiding assumptions and have robust global search capabilities. In addition, metaheuristics algorithms do not require a deep understanding of related fields. In recent decades, metaheuristics algorithms have been utilized to solve a multitude of real-world issues and intricate engineering designs. The metaheuristic optimization algorithms can be categorized as: evolutionary algorithms, physics-based algorithms, and swarm intelligence algorithms [14].

The majority of real-world issues may be phrased as optimization problems, which can be addressed by finding the values that maximize or minimize of the target. However, an analytic solution for such a function is seldom accessible. Also, these problems are usually of large dimension and contain several local optima, making conventional approaches susceptible to being stuck. Metaheuristic optimization algorithms have been used many times to find the best solutions in this setting [15].

In recent years [14][15][1], Many metaheuristics have been proposed, based on Darwinian processes, emergent collective behavior of animal groupings, and natural dynamic events, among others. A selection of these algorithms may be seen in Table 1.

According to recent studies [1], hybridizing two algorithms is advantageous for the design of superior algorithms because it not only maximizes the strengths of each algorithm but also minimizes its flaws. Many optimization methods have been combined in recent years to increase their overall efficiency and make them more helpful for handling real-world optimization issues. As shown in Table 1, recent optimization algorithms were published between 2019 and 2022.

Table 1
sample of resent optimization algorithm published from 2019–2022.

Reference	Year	Algorithm name	Abbreviation
Heidari et al.[16]	2019	Harris Hawks Optimization	HHO
Qiao & Yang [17]	2019	developed a chaotic-based dolphin swarm algorithm	DSA
Arora & Singh [18]	2019	butterfly optimization algorithm	BOA
Askari et al.[19]	2020	Heap-Based Optimization	HBO
Faramarzi et al.[20]	2020	Equilibrium Optimizer	EO
Kaur et al.[21]	2020	Tunicate swarm algorithm	TSA
Rahkar Farshi [22]	2021	Battle Royale Optimization	BRO
Zhang et al.[23]	2022	Mayfly Algorithm	MA
Hamad et al.[24]	2022	Q-learning embedded sine cosine algorithm	QLESCA
Farshi et al.[25]	2022	Binary Battle Royale Optimizer algorithm	BinBRO

The Concept Of Q-learning

Q-learning is a well-known artificial intelligence technique [9]. Since the date of its proposal, it has received a lot of attention from researchers for its efficiency and ease of application, according to the Web of Science database for the 11 years from 2012 to 2022. It was used in more than 5606 research studies. Figure 2 shows the number of research papers published in each of these years.

Q-learning is an algorithm for reinforcement learning that identifies the optimal selection policy. Q-learning updates the statuses of the agents as they interact with the environment. In each stage, an agent does actions for which they are rewarded or punished. The five components of Q-learning are agents, environment, actions, state, and reward, as illustrated in Fig. 3. The connection between Q-table and environment is demonstrated in Fig. 4.

Each agent has m states and n actions in each state when using Q-learning. The agent uses the Q-table to calculate the likelihood of picking different behaviors in different conditions. Table 2 shows how.

Table 2 The structure of Q table.

State	Action			
	A_1	A_2	...	A_n
S_1	$Q(S_1, A_1)$	$Q(S_1, A_2)$	...	$Q(S_1, A_n)$
S_2	$Q(S_2, A_1)$	$Q(S_2, A_2)$	...	$Q(S_2, A_n)$
.	.	.	...	.
.	.	.	...	.
.	.	.		.
S_m	$Q(S_m, A_1)$	$Q(S_m, A_2)$	...	$Q(S_m, A_n)$

Iterative learning is what Q-learning is all about. The formula for updating the Q value is as follows:

$$Q_{t+1}(S_t, A_t) = \underbrace{(1-\alpha)\, Q(S_t, A_t)}_{\text{Old value}} + \underbrace{\alpha}_{\text{Learning rate}} \left(\underbrace{R_t}_{\text{Reward}} + \underbrace{\gamma}_{\text{Discount factor}} \underbrace{\max Q(S_{t+1}, A_t)}_{\text{Estimate of optimal future value}} \right) \quad (1)$$

Learned value

where α is the learning rate, which is in the same range as the discount factor γ, both are in ranch from 0 to 1. When γ = 0, the Q-learning algorithm considers the current reward only, and correspondingly, it looks for a long-term reward if γ = 1. $Q_{t+1}(S_t, A_t)$ is the value function that the agent has gained at time *t*. In algorithm number 1, the search steps of the Q-learning algorithm are illustrated.

	Algorithm 1: Steps in the Q-learning algorithm
1.	At the start, make an empty Q-table.
2.	While (condition = true)
3.	Choose the optimal action based on the Q-table.
4.	Perform the chosen action
5.	calculate the reward
6.	The Eq. (1) is used to update the Q-table
7.	End

To illustrate the process of updating the Q-table, a numerical illustration is provided in Fig. 5. Assuming that the current state of the agent $Q(S_t, A_t) = 5$ as depicted in Fig. 5(a), the next action could be one of four: move up, move down, move right, or move left. Each move is rewarded differently, including 25 points for going to the left, 50 points for moving up, 75 points for moving to the right, and 100 points for moving down. As a result, the best action, based on previously acquired incentives. Therefore, the new action (move down) will be performed, and a new reward will be calculated. Suppose a reward R_t is 1. Let's set the discount factor (γ) equal to 0.1 and the learning rate parameter (α) equal to 0.9 [27]. So, the Q-table will be updated as follows using Eq. (1):

$$Q_{t+1}(S_t, A_t) = 0.1*5 + 0.9 * [1 + 0.1* max(25, 50, 75, 100)] = 10.4$$

Incorporating Q-learning Into Metaheuristic Algorithms

This section discusses the efforts of researchers to improve the solutions of optimization algorithms through the use of Q-learning capabilities. Each search action is governed by the reward and penalty concept in order to keep agents moving toward the global optimal solution.

The optimization algorithm combined a hybrid PSO (HPSO) algorithm with the DE process developed by Kim and Lee [28]. Combining the proposed HPSO with Q-learning increases the search capabilities by selecting algorithm control parameters adaptively. A Q-table is utilized to match the relationship between the three-dimensional state (s_1, s_2, and s_3) and the three actions (MR, CR, BB).

Gao et al. [29] have presented a positioning system for underwater vehicles. Injection Q-learning was utilized to improve the PSO search equation. They used Q-learning, which can compare the expected value of different actions without needing a model of the environment.

Using Q-learning, Rakshit et al. [30] created a new variant of the Differential Evolution (DE) algorithm, called TDQL. The Q-table is used to control only two parameters, which are DE scaling factors (F_1 and F_2), in the proposed algorithm. The Q-table is updated based on the DE agents. For optimal performance, scaling factors for all search agents of the DE algorithm should not be equal. The search agent with the best fitness should search the local area, whereas the agent with the worst performance should participate in the global search. Thus, a good search agent should have small scaling factors, while a poor one should have relatively large scaling factors. According to the reported results, TDQL performed better than other DE variants.

Samma et al. [27] introduced a new Particle Swarm Optimization algorithm based on reinforcement learning (PSO). RLMPSO is the name of the suggested algorithm. Each search agent is exposed to five operations under the supervision of Q-learning: exploration, convergence, high-jump, low-jump, and fine-tuning. The agent follows these operations in line with the Q-learning activity. The RLMPSO is evaluated using four unimodal and multimodal benchmark problems, six composite benchmark problems, five

shifted and rotated benchmark issues, and two benchmark application problems. According to the results of the experiments, the proposed model outperforms a range of state-of-the-art PSO-based algorithms.

The QSO algorithm was created by Hsieh and Su [8], who merged Q-learning with PSO. The top global search agent is chosen in QSO based on its overall performance rather than its performance during a single test. There are N search agents in the QSO population, and each agent has an external and an interior state. While each agent's external state identifies the individual within the population and is unaffected throughout the optimization phase, each person's interior state will be altered once it takes action. The internal state of an agent reflects the agent's current position. Each agent can do two sorts of activities (imitation and disruption) in each condition. Because an agent is often motivated by the success of its neighbors, and so aspires to mimic the behavior of the best agent worldwide, any agent in a population can benefit from the discoveries made by other agents in the population during their interactions.

Watchanupaporn and Pudtuan [31] utilized Q-learning to enhance PSO for solving the multi-robot target problem. Each robot will search for the desired path by assigning rewards and penalties to specific actions. The Q-table is shared by all robots. A robot moves in the Q-table direction with the highest value.

Ma and Zhang [32] introduced a unique optimization technique known as QABC, which is built by combining Q-learning into the ABC solution search equation. The QABC's basic principle is to employ bees and spectator bees to discover the best nectar supply location using a solution search equation chosen by Q-learning. In order to update the position of the nectar source, the method determines the solution search equation that corresponds to the optimal Q-table value. This strategy improves the algorithm's exploitation potential, and the ranking-based selection probability helps to keep the population diverse.

Q-learning used to dynamically determine the optimal search operator from four different (Sine, Cosine, Levy Flight, Crossover) during runtime by Zamli et al. [33]. The new algorithm called QLSCA. Q-learning is employed to toggle between these four possibilities, selecting the ideal one based on prior awards. This method utilizes a single Q-table with a dimension of 4*4. (4 actions and 4 states). Two additional operators were added. These are the Levy Flight and crossover operators. The QLSCA outperformed five state-of-the-art algorithms, including the original SCA, the particle swarm test generator (PSTG), adaptive particle swarm optimization (APSO), the cuckoo search (CS), and particle swarm optimization (DPSO), according to experimental data.

To assist robots in determining the optimal path through an unknown environment and in learning about it. Meerza et al. [34] combined Q-learning and PSO into a single algorithm termed QL + PSO. There are only four actions in the action set: forward, backward, left, and right. The particle receives a reward based on the action. The optimal position of each particle in the swarm is determined using the Q-table; additionally, particles learn their optimal position using the Q-table until the end of the trail. Simulated results indicate that the proposed algorithm outperforms Q-learning and PSO alone.

Li et al. [13] propose a novel approach for differential evolution based on reinforcement learning and fitness ranking. DE-RLFR is the proposed name for the algorithm. Q-learning directs DE search agents towards the ideal solution. every individual in the population is regarded an agent. The order of each agent's multi-fitness function values determines the hierarchical state variable, and three typical DE mutation procedures are accessible. In a generation, those that achieve or remain in a higher hierarchical state zone are rewarded, and the agent's experience is maintained in the Q-table and updated with each iteration to assist all agents in selecting the optimal mutation strategy for the following iteration.

Liu et al. [26] proposed a Q-Learning-based PSO, dubbed QLPSO, which trains the PSO parameters using Q-Learning. At the heart of the QLPSO algorithm is a Q-table containing a state and an action. By adjusting various parameters, the action regulates the exploration and exploitation of particles. Based on their past actions (rewards and punishments), the Q-table helps particles choose the best thing to do next.

Xu and Pi [35] proposed a dynamic communication topology in PSO based on Q-learning, dubbed QLPSO. When solving more complex problems, the primary advantage of dynamic topology is that it avoids falling into the local solution. Each particle acts independently in the proposed algorithm, selecting the optimal topology under the control of Q-learning during each iteration. The performance of QLPSO is compared to static and dynamic topologies for 28 CEC 2013 benchmark functions. The presented results demonstrate that the QLPSO outperforms several state-of-the-art methods.

Chen et al. [36] introduced the reinforcement learning mechanism into genetic algorithm (GA). RLGA is the proposed algorithm. The crossover and mutation operations are carried out using Q-learning to determine the crossover fragments and mutation points to be optimized. RLGA contains two Q-tables, one dedicated to crossover and another to mutation. RLGA was compared to traditional GA and state-of-the-art methods, and the experimental results indicated that RLGA is capable of achieving higher performance.

A single optimization model called QLSA was proposed by Samma et al. [37]. The Q-learning method is combined with Simulated annealing (SA). The Q-learning method is integrated into SA to improve its performance by adaptively adjusting its parameters during run time. Q-learning is used to keep track of the optimum performance values of SA parameters. A total of seven constrained engineering design problems were employed in this study to evaluate the efficacy of the proposed QLSA method. QLSA was compared to state-of-the-art population optimization methods such as PSO, GWO, CLPSO, Harmony, and ABC for further investigation. The results reveal that QLSA beats the other algorithms tested substantially.

Zhang et al. [38] utilized Q-learning to improve the PSO search process by choosing three different actions. These are the Exploration, Exploitation, and Jump actions. Based on the selected action, the PSO velocity and position will be updated, resulting in a new search area for the PSO agent. Based on the value of the Q-table, the DQN-PSO evaluates the reward of the agent's action and selects the action that can generate the greatest reward.

Oztop et al. [39] proposed a new optimization algorithm based on General Variable Neighborhood Search (GVNS) and Q-learning. GVNS-QL is the name of the proposed algorithm. The proposed GVNS-QL

algorithm doesn't use constant parameter values. Instead, it uses Q-learning to figure out GVNS parameters.

To solve the job-shop scheduling problem, Chen et al. [40] propose a self-learning genetic algorithm (SLGA). The GA's key parameters are adjusted intelligently using reinforcement learning. In this algorithm, two reinforcement learning methods are utilized. These techniques are the SARSA and Q-Learning algorithms. In the initial and final stages of optimization, the SARSA and Q-Learning algorithms are applied as learning methods, and the conversion condition is designed. Second, the state determination method and the reward method are tailored for reinforcement learning in a GA setting.

Huynh et al. [41] used the Q-learning to improve the performance of Differential Evolution. The proposedalgorithm called qlDE. The Q-learning model is incorporated into DE as an adaptive parameter controller, adjusting the algorithm's control parameters at each search iteration to improve its behavior for diverse search domains. The performance of the proposed algorithm was improved by automatically adjusting the balance between two phases (exploration and exploitation). Five benchmark instances of truss structural weight minimization were performed in this work to validate the efficiency of the qlDE in contrast to the traditional DE and many other methods in the literature. Seyyedabbasi et al. [42] presented three optimization methods based on Q-learning to solve global optimization problems, based on three metaheuristic algorithms: I-GWO [43], Ex-GWO [43], and Whale Optimization Algorithm (WOA)[44]. The suggested algorithms are known as $RL_{I\text{-}GWO}$, $RL_{Ex\text{-}GWO}$, and RL_{WOA}. These algorithms' search agents use the Q-Table values to choose between the exploration and exploitation phases. The Q-Table is in charge of controlling two phases in order to make more effective judgments. It also aims to help search agents discover new areas of the global search universe. To determine the reward and punishment values for each activity, a control mechanism was used. Each suggested method aims to handle global optimization issues as efficiently as possible while avoiding the local optimum trap. The algorithms proposed in this work were put to the test on 30 benchmark functions from CEC 2014 and 2015, and the results were compared against three metaheuristics, as well as traditional GWO and WOA. The presented approaches have also been used to solve the challenge of inverse kinematics for robot arms. The results showed that RL_{WOA} is more effective at solving test problems.

Li et al. [45] combined the Q-learning and GA algorithms to create the QGA algorithm. The core idea of the proposed algorithm is to treat the GA's gene space as the Q-learning algorithm's action strategy space. In other words, in Q-learning, the selection action corresponds to the genetic selection operator in GA. QGA was used to schedule tasks. The experimental results demonstrated the algorithm's effectiveness.

Lu et al. [46] proposed a reinforcement learning-based PSO (RLPSO) for optimizing wastewater treatment processes (WWTP). RLPSO is composed of four fundamental components. The agent is a population particle, while the environment is the WWTP. The state represents the position of each particle in the population; the action represents the strategy for predicting the velocity of each particle, which is determined by the Q-table.

To solve problems involving dynamic optimization, Gölcük and Ozsoydan [47] proposed an algorithm recommendation architecture based on Q-learning to provide guidance on the most appropriate metaheuristic algorithm selection in changing environments. Due to the abundance of design options, choosing an appropriate metaheuristic algorithm becomes an immediate challenge. To accomplish this, the Artificial Bee Colony (ABC), Manta Ray Foraging Optimization (MRFO), Salp Swarm Algorithm (SSA), and Whale Optimization Algorithm (WOA) were used as low-level optimizers, with Q-learning selecting the optimizer based on its behavior. The results show the effectiveness of the Q-learning-based algorithm recommender in solving the dynamic multidimensional knapsack problem.

To increase the efficacy of brain storming optimization (BSO) Zhao et al. [48] proposed a new algorithm called the reinforcement learning brain storm optimization algorithm (RLBSO) by leveraging the power of reinforcement learning. The RLBSO is equipped with four mutation strategies that significantly balance the RLBSO's local and global search capabilities. The Q-learning mechanism was implemented to guide the selection of mutation strategies based on the feedback of historical data. The RLBSO was evaluated against the CEC 2017 benchmark and outperformed advanced BSO algorithm variants and state-of-the-art algorithms.

Based on DE, Hu and Gong [49] introduced a new optimization technique called RL-CORCO; they chose between two mutation strategies using Q-learning. The proposed algorithm has nine hierarchical populations; hence the Q-table is a (9 x 2) matrix with rows and columns representing actions and states, respectively. Then, based on the population's state changes after each strategy selection, the reward is obtained and the Q-table is modified.

Liao and Li [50] proposed ERLDE, a new DE variant based on Q-learning; they employed a Q-table with four actions and four states. Q-learning is used to tell the DE search agents what to do by controlling the mutation strategy and the size of the neighborhood based on the idea of rewards and punishments.

Wang et al. [51] infused Q-learning into an adaptive artificial bee colony (ABC) to determine a search operator dynamically. There are twelve states and six actions in the Q-table. The Q-learning algorithm chooses between these six actions, which represent six distinct search operators. So, different search operators can be used in different generations, and the chance of getting stuck in a locally optimal solution can be lowered.

Wu et al. [52] proposed a modification to the teaching-learning-based optimization algorithm by incorporating Q-learning into the RLTLBO algorithm. First, the teacher phase of basic TLBO is completed. The Q-learning system then works in the learner phase to enhance the search process. This facilitates the agents' comprehensive learning, thereby accelerating the convergence rate. The random opposition-based learning (ROBL) technique is added to make it easier to avoid local optima.

Hamad et al. [24] employed the Q-Learning algorithm to improve the original SCA. Q-Learning guides search agents toward a more efficient discovery and a global solution by skipping local optima. The Q-learning algorithm was introduced into SCA to control the values of two important parameters (r_1 and r_3).

These parameters are crucial for directing the movement of the search agent within the search area. The Q-Table attempts to adjust the values of these two parameters in line with the values stored in the Q-Table. This algorithm has five search agents, and each one has a Q-table, resulting in five Q-tables. The table size in QLESCA is 9 × 9 (9 actions and 9 states). The Q-table governs the process of switching between exploration and exploitation in QLESCA. This means that it can start with exploration and then move to exploitation, or the other way around.

A new PSO variant based on Q-learning for ship damage stability design has been proposed by Huang et al. [53]. The proposed algorithm is called QLPSO. Q-learning identifies the optimal PSO settings (w, c1, and c2). Agents from the PSO algorithm are utilized for Q-learning as agents. The Q-learning environment is the search space for agents. The current search operation of the agent represents the state of Q-learning. The search operation in this algorithm consists of exploration, convergence, and jump operations. The Q-learning action is the agent's next search operation. If the agent achieves a good result, it will receive a positive reward; otherwise, it will receive a penalty.

Wang et al. [54] suggested a PSO variant termed reinforcement learning level-based PSO for high-dimensional problems (RLLPSO). They apply the level-based population structure inspired by LLSO [55] to enhance the exploration capability of PSO and prevent its premature inclination. In addition, they presented a reinforcement learning technique for level number control that automatically adjusts the population's level number. RLLPSO can identify the optimal level structure to enhance the population's search potential and increase search efficiency based on the environmental reward. RLLPSO uses a level competition mechanism to speed up convergence and manage the large search space. This is done by making it more likely that top agents will be chosen as learning examples.

Table 3 summarizes the main characteristics of each Q-learning-based optimization algorithms.

Table 3
Description of Q-learning-based optimization algorithms.

Reference	Year	Proposed Algorithm	Metaheuristic Algorithm	Characteristics
Kim & Lee [28]	2009	-	HPSO	To improve the search capabilities of HSPO algorithm, Q-learning is used to select HSPO control parameters adaptively.
Gao et al.[29]	2009	-	PSO	They update the PSO search equation by using Q-learning, so the agent will update his position based on the best value stored in the Q-table.
Rakshit et al.[30]	2013	TDQL	DE	Each DE search agent is controlled through Q-Learning by controlling the DE scaling factors.
Samma et al.[27]	2016	RLMPSO	PSO	Q-learning used to control the operations of each PSO search agent. Each agent has five operations: exploration, convergence, high-jump, low-jump, and fine-tuning.
Hsieh & Su [8]	2016	QSO	PSO	The QSO algorithm chooses the best agents in the population based on their cumulative performances. The Q-table in the QSO consists of two states (external state and internal state) and two actions (imitation and disturbance) to direct the search agent to the optimal global solution.
Watchanupaporn & Pudtuan [56]	2016	-	PSO	Q-learning is employed to improve PSO for solving the multirobot target problem.
Ma & Zhang [32]	2016	QABC	ABC	Several different search strategies that change based on Q-learning were mapped into an action that was used to change the location of the nectar source location.
Zamli et al.[33]	2018	QLSCA	SCA	During runtime, Q-learning is utilized to dynamically identify the best functioning of SCA, eliminating the chance of switching between sine and cosine equations. To switch between these two possibilities, Q-learning is employed, and it picks the best option based on previous awards.
Meerza et al.[34]	2019	QL + PSO	PSO	Each PSO particle is guided by the Q-table to find the optimal path until the end of the trial by selecting one of four actions (forward, backward, left, or right) based on previous rewards.

Reference	Year	Proposed Algorithm	Metaheuristic Algorithm	Characteristics
Li et al.[13]	2019	DE-RLFR	DE	The values of each agent's fitness ranking are used to select a state. Three standard DE mutation operations are utilized as optional agent actions. Based on the stored reward and penalty from the Q-table, each agent chooses the best mutation operation for the next round.
Liu et al.[26]	2019	QLPSO	PSO	Q-learning guides the PSO particles to choose the best action at each iteration. These actions control the exploration and exploitation behavior of these particles.
Xu & Pi [35]	2020	QLPSO	PSO	Q-Learning was incorporated into the PSO in order to determine the optimal communication topology for each particle at each iteration based on the problem that the PSO was aimed to resolve.
Chen et al.[36]	2020	RLGA	GA	Two Q-tables were used, one to determine the crossover fragments and another to determine mutation points.
Samma et al.[37]	2020	QLSA	SA	Q-learning is used to modify SA parameters adaptively during run time. The optimum performance values of SA parameters are tracked using Q-learning.
Zhang et al.[38]	2020	DQN-PSO	PSO	Q-learning is used to choose between three actions (Exploration, Exploitation, and Jump). Q-learning attempts to choose the most rewarding action. The position of the PSO agent is updated based on the selected action.
Oztop et al.[39]	2020	GVNS-QL	GVNS	The suggested approach uses Q-learning to define the GVNS algorithm's parameters.
Chen et al.[40]	2020	SLGA	GA	The SARSA algorithm and Q-learning algorithm are applied in the learning module of SLGA. SLGA is working on integrating their methods to gain the benefits of both, allowing for faster learning speed and higher solution precision.
Huynh et al.[41]	2021	qlDE	DE	The Q-learning is implemented in DE as an adaptive parameter controller, which adjusts the algorithm's control parameters at each search iteration to enhance its performance over a variety of search domains.

Reference	Year	Proposed Algorithm	Metaheuristic Algorithm	Characteristics
Seyyedabbasi et al.[42]	2021	RL_{I-GWO}	GWO	The Q-learning is integrated into three optimization algorithms (I-GWO, Ex-GWO, and WOA) in order to increase the performance of these methods. These algorithms' search agents use the Q-Table values to choose between the exploration and exploitation phases. According to the reward and penalty values for each search agent action, the Q-Table is filled with information.
Seyyedabbasi et al.[42]	2021	RL_{Ex-GWO}	GWO	
Seyyedabbasi et al.[42]	2021	RL_{WOA}	WOA	
Li et al.[45]	2021	QGA	GA	The genetic algorithm chooses the best genetic for the next generation based on the Q-table value.
Lu et al.[46]	2021	RLPSO	PSO	The Q-table used to predict each particle's velocity in the PSO.
Gölcük & Ozsoydan [47]	2021	-	ABC, MRFO, SSA, and WOA	From these four algorithms, Q-learning selects which is the best algorithm based on their previous performance.
Zhao et al.[48]	2022	RLBSO	BSO	At each iteration, Q-learning is used to select one of four mutation strategies based on the historical record of rewards and penalties.
Hu & Gong [49]	2022	RL-CORCO	DE	To pick between two mutation techniques, Q-learning is applied. There are nine states and two actions in the Q-table.
Liao & Li [50]	2022	ERLDE	DE	Q-learning is used to choose the DE algorithm's mutation strategy and neighborhood size.
Wang et al.[51]	2022	QABC	ABC	Q-learning is used to dynamically select between six different search operators of ABC.
Wu et al.[52]	2022	RLTLBO	TLBO	In the learner phase of TLBO, Q-Learning is implemented to create a system for transitioning between two different learning modes. Then, ROBL is introduced to the algorithm to improve its avoidance of local optima.
Hamad et al.[24]	2022	QLESCA	SCA	Q-learning is used to guide the SCA's search agent to the best search area. The Q-table has nine states and nine actions to control the values of two of SCA's important parameters (r_1 and r_3). By changing these parameters, QLESCA can make the search phase go from exploration to exploitation and back again.

Reference	Year	Proposed Algorithm	Metaheuristic Algorithm	Characteristics
Huang et al.[53]	2022	QLPSO	PSO	Q-learning works to determine the optimal PSO parameter settings.
Wang et al.[54]	2022	RLLPSO	PSO	In lieu of manual control, the Q-Learning approach for level number control was implemented to auto-adaptively alter the population's level structure. Due to the adaptability and flexibility provided by Q-Learning, the suggested algorithm can improve search efficiency by taking the best probable actions based on the reward earned during the learning process.

The number of metaheuristic optimization algorithms published between 2009 and 2022 that used the Q-learning algorithm to improve their performance is depicted in Fig. 6. As shown by this graph, the number of algorithms has increased over the past three years (2020, 2021, and 2022) by 6, 7, and 8 algorithms, respectively.

Figure 7 illustrates the metaheuristic optimization algorithms that were enhanced by the Q-learning algorithm. As shown in the graph, 14 metaheuristic optimization algorithms are used by researchers to enhance their performance by utilizing Q-learning. PSO ranked first with the greatest number of variants, with eleven proposed algorithms. Then DE ranked second with five variants, ABC ranked third with three variants, followed by SCA, GA, WOA, and GWO with two variants each. Only one variant exists for TLBO, SA, BSO, MRFO, SSA, GVNS, and HPSO.

Conclusions And Future Research Directions

A recent review of metaheuristic optimization algorithm enhancements based on the Q-Learning concept is presented in this paper. Starting from 2009, the optimization algorithm has been modified based on Q-Learning by researchers to enhance its effectiveness and efficiency in solving various optimization problems. The findings suggest that combining optimization techniques with Q-learning may considerably enhance population search efficiency in the solution space. Long-term returns might be utilized to gain experience by observing how different hierarchical levels in the solution space behave. With this method, optimization algorithms will be able to choose the best search strategies or control parameters in a more flexible way. Because of the global focus on Q-learning and optimization algorithms' ability to solve many engineering challenges, this trend may continue to grow rapidly in the future. This assertion is supported by our analysis, which revealed that 21 of 32 algorithms were developed between 2019 and 2022. All of this information can help new and experienced researchers choose an appropriate Q-Learning-optimization algorithm integration.

The majority of researchers utilized PSO within 11 attempts by incorporating Q-Learning, followed by DE within 5 attempts, according to our findings. Last but not the least, we think this review can have a big

impact, especially for new researchers who want to look into how Q-Learning can be used with optimization algorithms based on different approaches used by many researchers.

Declarations

Author Contributions Qusay Shihab Hamad: Conceptualization, Methodology, Analysis, Investigation, Writing original draft. **Hussein Samma**: Conceptualization, Validation, Resources, Supervision. **Shahrel Azmin Suandi**: Conceptualization, Supervision, Project administration, Funding acquisition, Writing, review and editing.

Funding This research is supported by the Malaysia Ministry of Higher Education (MOHE) Fundamental Research Grant Scheme (FRGS), no. FRGS/1/2019/ICT02/USM/03/3.

Competing Interests The authors have no relevant financial or non-financial interests to disclose.

Ethics Approval The article does not contain any studies with human participants or animals performed by any of the authors.

Consent to Participate Not applicable.

Consent to Publication Not applicable.

Data availability Not applicable.

References

1. J. Liu, X. Liu, Y. Wu, Z. Yang, and J. Xu, “Dynamic multi-swarm differential learning harris hawks optimizer and its application to optimal dispatch problem of cascade hydropower stations,” *Knowledge-Based Syst.*, vol. 242, p. 108281, Apr. 2022, doi: 10.1016/j.knosys.2022.108281.
2. M. Braik, A. Hammouri, J. Atwan, M. A. Al-Betar, and M. A. Awadallah, “White Shark Optimizer: A novel bio-inspired meta-heuristic algorithm for global optimization problems,” *Knowledge-Based Syst.*, vol. 243, p. 108457, May 2022, doi: 10.1016/j.knosys.2022.108457.
3. F. A. Hashim and A. G. Hussien, “Snake Optimizer: A novel meta-heuristic optimization algorithm,” *Knowledge-Based Syst.*, vol. 242, p. 108320, Apr. 2022, doi: 10.1016/j.knosys.2022.108320.
4. D. H. Wolpert and W. G. Macready, “No free lunch theorems for optimization,” *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, 1997, doi: 10.1109/4235.585893.
5. M. Braik, A. Hammouri, J. Atwan, M. A. Al-Betar, and M. A. Awadallah, “White Shark Optimizer: A novel bio-inspired meta-heuristic algorithm for global optimization problems,” *Knowledge-Based Syst.*, vol. 243, p. 108457, May 2022, doi: 10.1016/j.knosys.2022.108457.
6. I. Ahmadianfar, A. A. Heidari, S. Noshadian, H. Chen, and A. H. Gandomi, “INFO: An efficient optimization algorithm based on weighted mean of vectors,” *Expert Syst. Appl.*, vol. 195, p. 116516, Jun. 2022, doi: 10.1016/j.eswa.2022.116516.

7. F. A. Hashim and A. G. Hussien, “Snake Optimizer: A novel meta-heuristic optimization algorithm,” *Knowledge-Based Syst.*, vol. 242, p. 108320, Apr. 2022, doi: 10.1016/j.knosys.2022.108320.
8. Y.-Z. Hsieh and M.-C. Su, “A Q-learning-based swarm optimization algorithm for economic dispatch problem,” *Neural Comput. Appl.*, vol. 27, no. 8, pp. 2333–2350, Nov. 2016, doi: 10.1007/s00521-015-2070-1.
9. C. J. C. H. Watkins and P. Dayan, “Q-learning,” *Mach. Learn.*, vol. 8, no. 3–4, pp. 279–292, May 1992, doi: 10.1007/BF00992698.
10. R. Dearden, N. Friedman, and S. Russell, “Bayesian Q-learning,” *Proc. Natl. Conf. Artif. Intell.*, pp. 761–768, 1998.
11. H. Hasselt, “Double Q-learning,” *Adv. Neural Inf. Process. Syst.*, vol. 23, pp. 2613–2621, 2010.
12. T. Hester *et al.*, “Deep q-learning from demonstrations,” 2018.
13. Z. Li, L. Shi, C. Yue, Z. Shang, and B. Qu, “Differential evolution based on reinforcement learning with fitness ranking for solving multimodal multiobjective problems,” *Swarm Evol. Comput.*, vol. 49, pp. 234–244, Sep. 2019, doi: 10.1016/j.swevo.2019.06.010.
14. X. Zhao, Y. Fang, S. Ma, and Z. Liu, “Multi-swarm improved moth–flame optimization algorithm with chaotic grouping and Gaussian mutation for solving engineering optimization problems,” *Expert Syst. Appl.*, p. 117562, May 2022, doi: 10.1016/j.eswa.2022.117562.
15. M. Castelli, L. Manzoni, L. Mariot, M. S. Nobile, and A. Tangherloni, “Salp Swarm Optimization: A critical review,” *Expert Syst. Appl.*, vol. 189, p. 116029, Mar. 2022, doi: 10.1016/j.eswa.2021.116029.
16. A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, “Harris hawks optimization: Algorithm and applications,” *Futur. Gener. Comput. Syst.*, vol. 97, pp. 849–872, Aug. 2019, doi: 10.1016/j.future.2019.02.028.
17. W. Qiao and Z. Yang, “Modified Dolphin Swarm Algorithm Based on Chaotic Maps for Solving High-Dimensional Function Optimization Problems,” *IEEE Access*, vol. 7, pp. 110472–110486, 2019, doi: 10.1109/ACCESS.2019.2931910.
18. S. Arora and S. Singh, “Butterfly optimization algorithm: a novel approach for global optimization,” *Soft Comput.*, vol. 23, no. 3, pp. 715–734, Feb. 2019, doi: 10.1007/s00500-018-3102-4.
19. Q. Askari, M. Saeed, and I. Younas, “Heap-based optimizer inspired by corporate rank hierarchy for global optimization,” *Expert Syst. Appl.*, vol. 161, p. 113702, Dec. 2020, doi: 10.1016/j.eswa.2020.113702.
20. A. Faramarzi, M. Heidarinejad, B. Stephens, and S. Mirjalili, “Equilibrium optimizer: A novel optimization algorithm,” *Knowledge-Based Syst.*, vol. 191, p. 105190, Mar. 2020, doi: 10.1016/j.knosys.2019.105190.
21. S. Kaur, L. K. Awasthi, A. L. Sangal, and G. Dhiman, “Tunicate Swarm Algorithm: A new bio-inspired based metaheuristic paradigm for global optimization,” *Eng. Appl. Artif. Intell.*, vol. 90, p. 103541, Apr. 2020, doi: 10.1016/j.engappai.2020.103541.

22. T. Rahkar Farshi, “Battle royale optimization algorithm,” *Neural Comput. Appl.*, vol. 33, no. 4, pp. 1139–1157, Feb. 2021, doi: 10.1007/s00521-020-05004-4.
23. T. Zhang, Y. Zhou, G. Zhou, W. Deng, and Q. Luo, “Bioinspired Bare Bones Mayfly Algorithm for Large-Scale Spherical Minimum Spanning Tree,” *Front. Bioeng. Biotechnol.*, vol. 10, Mar. 2022, doi: 10.3389/fbioe.2022.830037.
24. Q. S. Hamad, H. Samma, S. A. Suandi, and J. Mohamad-Saleh, “Q-learning embedded sine cosine algorithm (QLESCA),” *Expert Syst. Appl.*, vol. 193, p. 116417, May 2022, doi: 10.1016/j.eswa.2021.116417.
25. T. A. (Rahkar Farshi), S. Agahian, and R. Dehkharghani, “BinBRO: Binary Battle Royale Optimizer algorithm,” *Expert Syst. Appl.*, vol. 195, p. 116599, Jun. 2022, doi: 10.1016/j.eswa.2022.116599.
26. Y. Liu, H. Lu, S. Cheng, and Y. Shi, “An Adaptive Online Parameter Control Algorithm for Particle Swarm Optimization Based on Reinforcement Learning,” in *2019 IEEE Congress on Evolutionary Computation (CEC)*, Jun. 2019, pp. 815–822, doi: 10.1109/CEC.2019.8790035.
27. H. Samma, C. P. Lim, and J. Mohamad Saleh, “A new Reinforcement Learning-based Memetic Particle Swarm Optimizer,” *Appl. Soft Comput. J.*, vol. 43, pp. 276–297, 2016, doi: 10.1016/j.asoc.2016.01.006.
28. P. Kim and J. Lee, “An integrated method of particle swarm optimization and differential evolution,” *J. Mech. Sci. Technol.*, vol. 23, no. 2, pp. 426–434, Feb. 2009, doi: 10.1007/s12206-008-0917-4.
29. Y. Gao, J. Ye, Y. Chen, and F. Liang, “Q-learning based on particle swarm optimization for positioning system of underwater vehicles,” in *2009 IEEE International Conference on Intelligent Computing and Intelligent Systems*, Nov. 2009, pp. 68–71, doi: 10.1109/ICICISYS.2009.5358098.
30. P. Rakshit *et al.*, “Realization of an Adaptive Memetic Algorithm Using Differential Evolution and Q-Learning: A Case Study in Multirobot Path Planning,” *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 43, no. 4, pp. 814–831, Jul. 2013, doi: 10.1109/TSMCA.2012.2226024.
31. O. Watchanupaporn and P. Pudtuan, “Multi-robot target reaching using modified Q-learning and PSO,” in *2016 2nd International Conference on Control, Automation and Robotics (ICCAR)*, Apr. 2016, pp. 66–69, doi: 10.1109/ICCAR.2016.7486700.
32. P. Ma and H.-L. Zhang, “Improved Artificial Bee Colony Algorithm Based on Reinforcement Learning,” 2016, pp. 721–732.
33. K. Z. Zamli, F. Din, B. S. Ahmed, and M. Bures, “A hybrid Q-learning sine-cosine-based strategy for addressing the combinatorial test suite minimization problem,” *PLoS One*, vol. 13, no. 5, p. e0195675, May 2018, doi: 10.1371/journal.pone.0195675.
34. S. I. A. Meerza, M. Islam, and M. M. Uzzal, “Q-Learning Based Particle Swarm Optimization Algorithm for Optimal Path Planning of Swarm of Mobile Robots,” in *2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT)*, May 2019, pp. 1–5, doi: 10.1109/ICASERT.2019.8934450.
35. Y. Xu and D. Pi, “A reinforcement learning-based communication topology in particle swarm optimization,” *Neural Comput. Appl.*, vol. 32, no. 14, pp. 10007–10032, Jul. 2020, doi:

10.1007/s00521-019-04527-9.

36. Q. Chen, M. Huang, Q. Xu, H. Wang, and J. Wang, “Reinforcement Learning-Based Genetic Algorithm in Optimizing Multidimensional Data Discretization Scheme,” *Math. Probl. Eng.*, vol. 2020, pp. 1–13, Mar. 2020, doi: 10.1155/2020/1698323.
37. H. Samma, J. Mohamad-Saleh, S. A. Suandi, and B. Lahasan, “Q-learning-based simulated annealing algorithm for constrained engineering design problems,” *Neural Comput. Appl.*, vol. 32, no. 9, pp. 5147–5161, May 2020, doi: 10.1007/s00521-019-04008-z.
38. P. Zhang, H. Li, Q. P. Ha, Z.-Y. Yin, and R.-P. Chen, “Reinforcement learning based optimizer for improvement of predicting tunneling-induced ground responses,” *Adv. Eng. Informatics*, vol. 45, p. 101097, Aug. 2020, doi: 10.1016/j.aei.2020.101097.
39. H. Oztop, M. F. Tasgetiren, L. Kandiller, and Q.-K. Pan, “A Novel General Variable Neighborhood Search through Q-Learning for No-Idle Flowshop Scheduling,” in *2020 IEEE Congress on Evolutionary Computation (CEC)*, Jul. 2020, pp. 1–8, doi: 10.1109/CEC48606.2020.9185556.
40. R. Chen, B. Yang, S. Li, and S. Wang, “A self-learning genetic algorithm based on reinforcement learning for flexible job-shop scheduling problem,” *Comput. Ind. Eng.*, vol. 149, p. 106778, Nov. 2020, doi: 10.1016/j.cie.2020.106778.
41. T. N. Huynh, D. T. T. Do, and J. Lee, “Q-Learning-based parameter control in differential evolution for structural optimization,” *Appl. Soft Comput.*, vol. 107, p. 107464, Aug. 2021, doi: 10.1016/j.asoc.2021.107464.
42. A. Seyyedabbasi, R. Aliyev, F. Kiani, M. U. Gulle, H. Basyildiz, and M. A. Shah, “Hybrid algorithms based on combining reinforcement learning and metaheuristic methods to solve global optimization problems,” *Knowledge-Based Syst.*, vol. 223, p. 107044, Jul. 2021, doi: 10.1016/j.knosys.2021.107044.
43. A. Seyyedabbasi and F. Kiani, “I-GWO and Ex-GWO: improved algorithms of the Grey Wolf Optimizer to solve global optimization problems,” *Eng. Comput.*, vol. 37, no. 1, pp. 509–532, Jan. 2021, doi: 10.1007/s00366-019-00837-7.
44. S. Mirjalili and A. Lewis, “The Whale Optimization Algorithm,” *Adv. Eng. Softw.*, vol. 95, pp. 51–67, May 2016, doi: 10.1016/j.advengsoft.2016.01.008.
45. Z. Li, X. Wei, X. Jiang, and Y. Pang, “A Kind of Reinforcement Learning to Improve Genetic Algorithm for Multiagent Task Scheduling,” *Math. Probl. Eng.*, vol. 2021, pp. 1–12, Jan. 2021, doi: 10.1155/2021/1796296.
46. L. Lu, H. Zheng, J. Jie, M. Zhang, and R. Dai, “Reinforcement learning-based particle swarm optimization for sewage treatment control,” *Complex Intell. Syst.*, vol. 7, no. 5, pp. 2199–2210, Oct. 2021, doi: 10.1007/s40747-021-00395-w.
47. İ. Gölcük and F. B. Ozsoydan, “Q-learning and hyper-heuristic based algorithm recommendation for changing environments,” *Eng. Appl. Artif. Intell.*, vol. 102, no. November 2020, p. 104284, Jun. 2021, doi: 10.1016/j.engappai.2021.104284.

48. F. Zhao, X. Hu, L. Wang, J. Zhao, J. Tang, and Jonrinaldi, “A reinforcement learning brain storm optimization algorithm (BSO) with learning mechanism,” *Knowledge-Based Syst.*, vol. 235, p. 107645, Jan. 2022, doi: 10.1016/j.knosys.2021.107645.
49. Z. Hu and W. Gong, “Constrained evolutionary optimization based on reinforcement learning using the objective function and constraints,” *Knowledge-Based Syst.*, vol. 237, p. 107731, Feb. 2022, doi: 10.1016/j.knosys.2021.107731.
50. Z. Liao and S. Li, “Solving Nonlinear Equations Systems with an Enhanced Reinforcement Learning Based Differential Evolution,” *Complex Syst. Model. Simul.*, vol. 2, no. 1, pp. 78–95, Mar. 2022, doi: 10.23919/CSMS.2022.0003.
51. J. Wang, D. Lei, and J. Cai, “An adaptive artificial bee colony with reinforcement learning for distributed three-stage assembly scheduling with maintenance,” *Appl. Soft Comput.*, vol. 117, p. 108371, Mar. 2022, doi: 10.1016/j.asoc.2021.108371.
52. D. Wu, S. Wang, Q. Liu, L. Abualigah, and H. Jia, “An Improved Teaching-Learning-Based Optimization Algorithm with Reinforcement Learning Strategy for Solving Optimization Problems,” *Comput. Intell. Neurosci.*, vol. 2022, pp. 1–24, Mar. 2022, doi: 10.1155/2022/1535957.
53. X. Huang, G. Yang, C. Yang, Q. Sheng, and C. Pan, “A Collaborative Optimization Algorithm for Ship Damage Stability Design,” *J. Phys. Conf. Ser.*, vol. 2203, no. 1, p. 012071, Feb. 2022, doi: 10.1088/1742-6596/2203/1/012071.
54. F. Wang, X. Wang, and S. Sun, “A reinforcement learning level-based particle swarm optimization algorithm for large-scale optimization,” *Inf. Sci. (Ny).*, vol. 602, pp. 298–312, Jul. 2022, doi: 10.1016/j.ins.2022.04.053.
55. Q. Yang, W.-N. Chen, J. Da Deng, Y. Li, T. Gu, and J. Zhang, “A Level-Based Learning Swarm Optimizer for Large-Scale Optimization,” *IEEE Trans. Evol. Comput.*, vol. 22, no. 4, pp. 578–594, Aug. 2018, doi: 10.1109/TEVC.2017.2743016.
56. O. Watchanupaporn and P. Pudtuan, “Multi-robot target reaching using modified Q-learning and PSO,” in *2016 2nd International Conference on Control, Automation and Robotics (ICCAR)*, Apr. 2016, pp. 66–69, doi: 10.1109/ICCAR.2016.7486700.

Figures

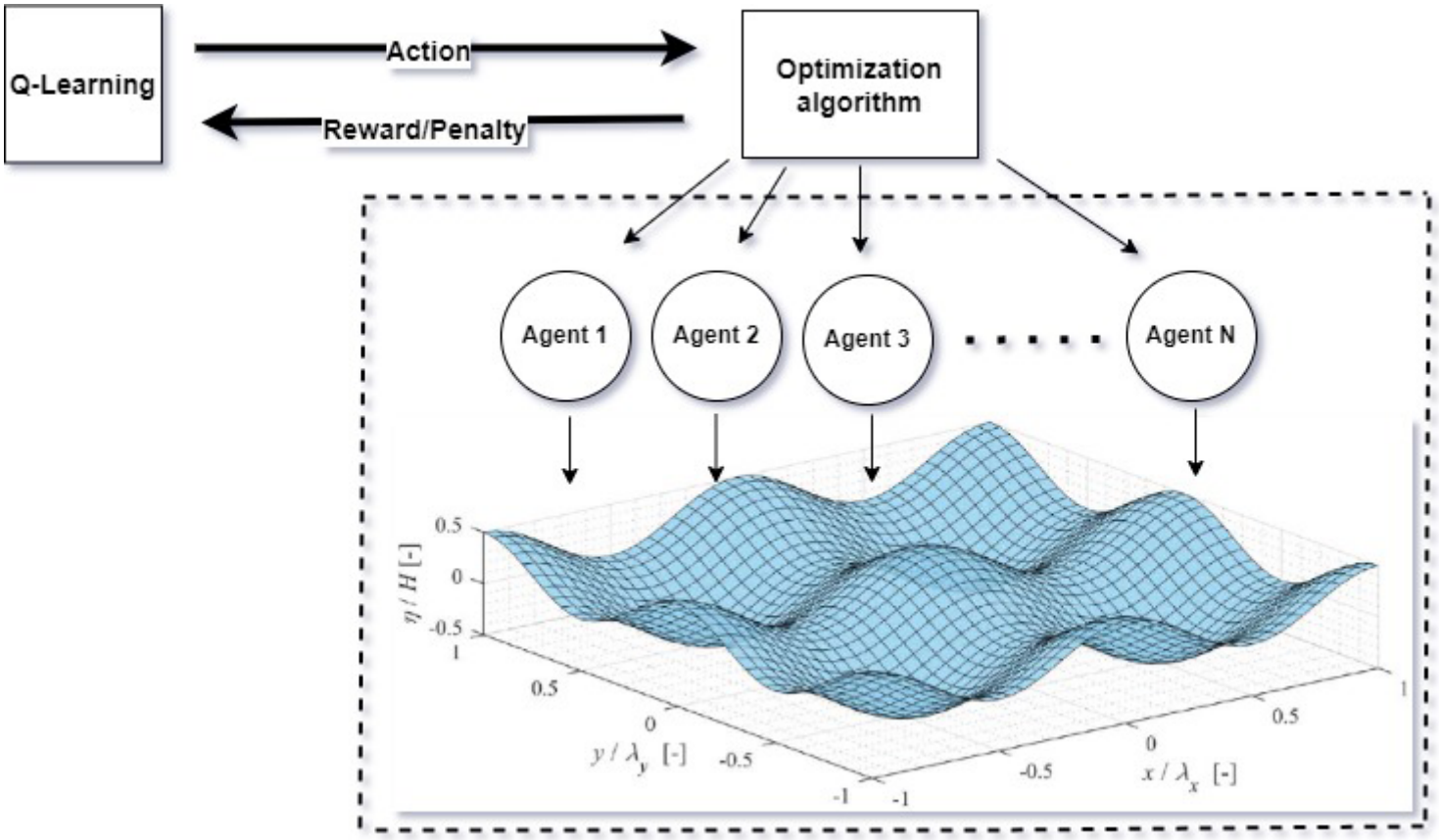

Figure 1

Illustration the way of connecting Q-Learning with optimization algorithms

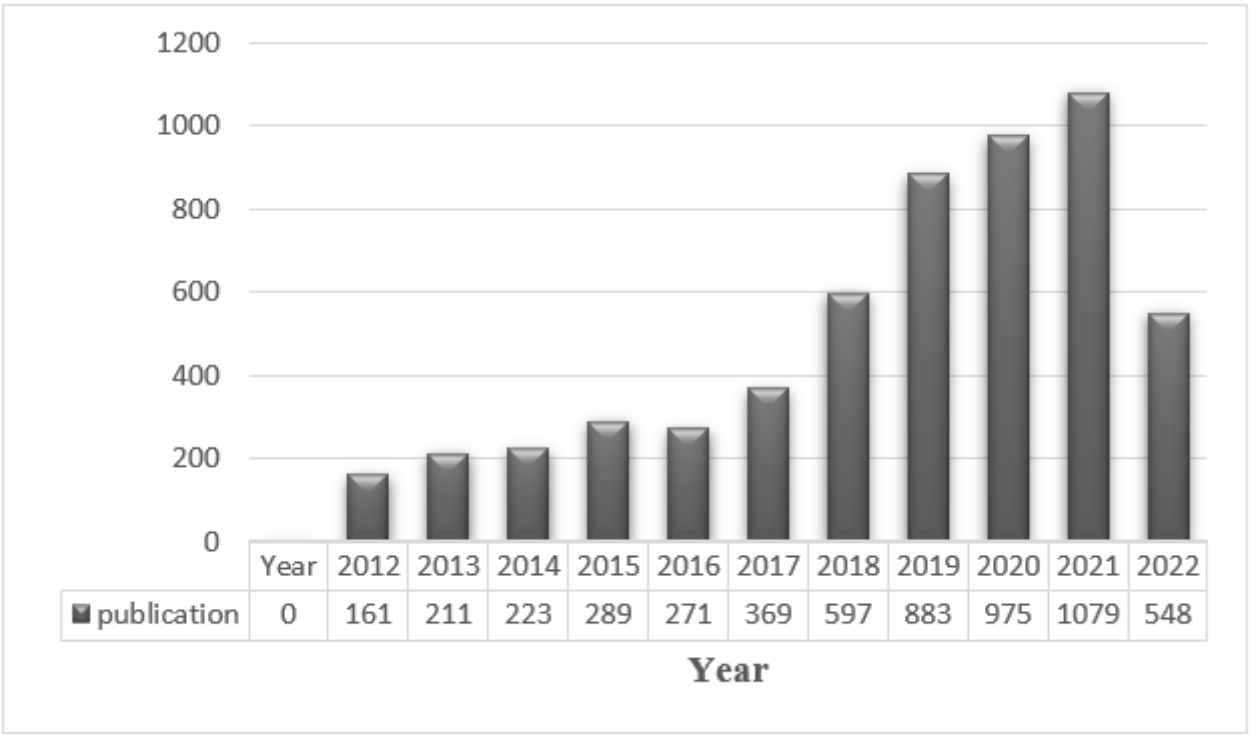

Figure 2

Annual research publications for Q-learning in Web of Science.

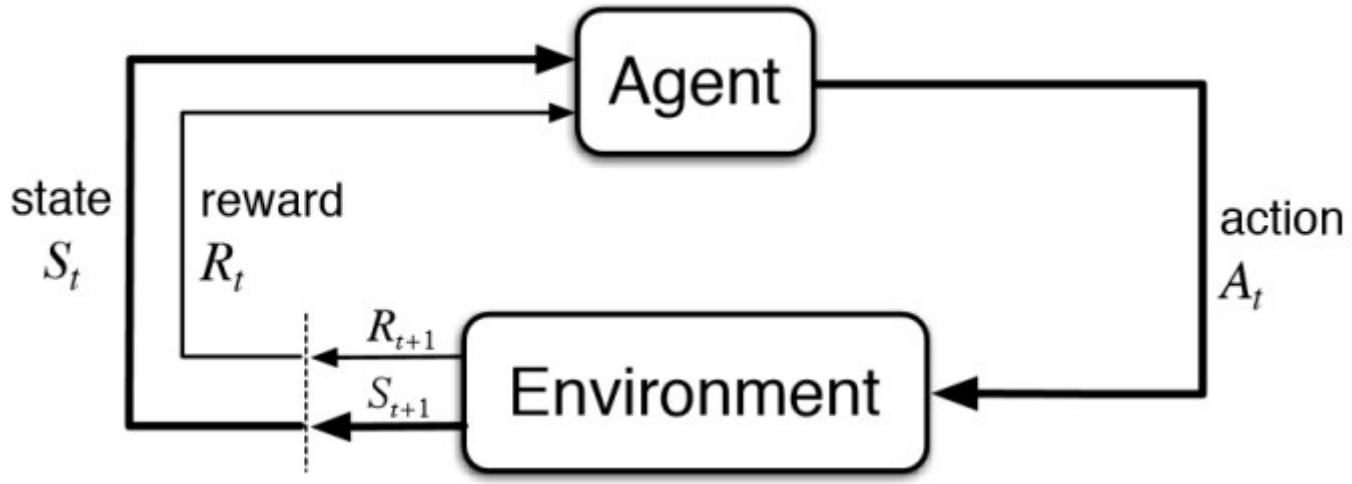

Figure 3

Illustration of Reinforcement learning process

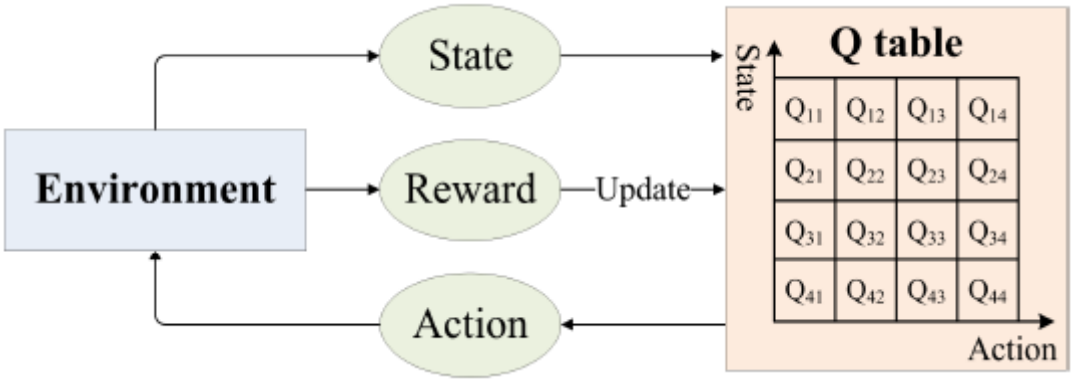

Figure 4

The model of Q-leaning [26].

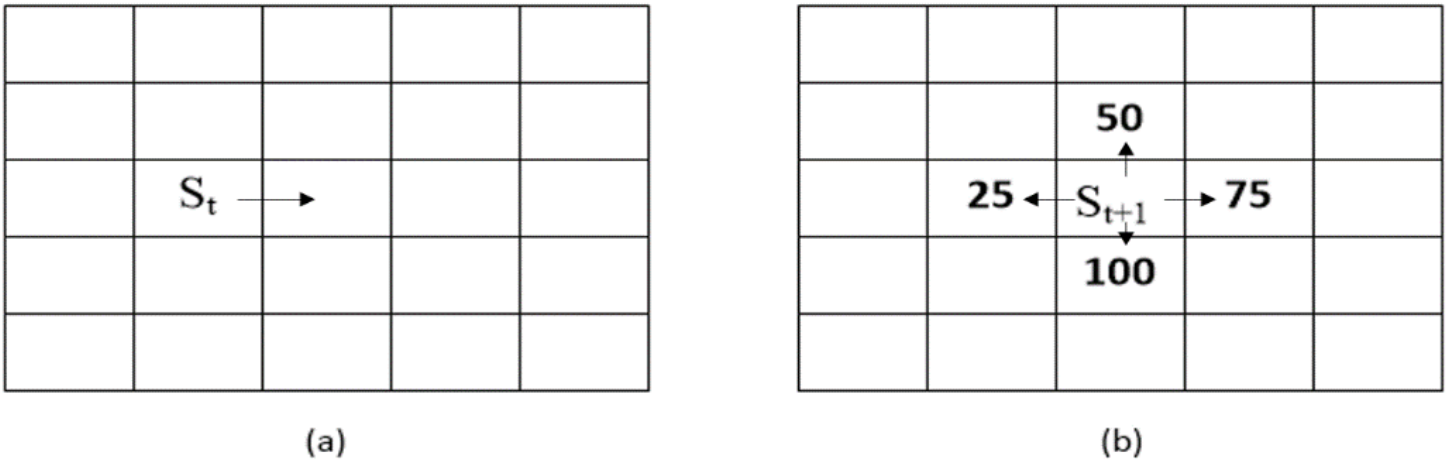

Figure 5

A numerical depiction of a) the present state and b) the upcoming state

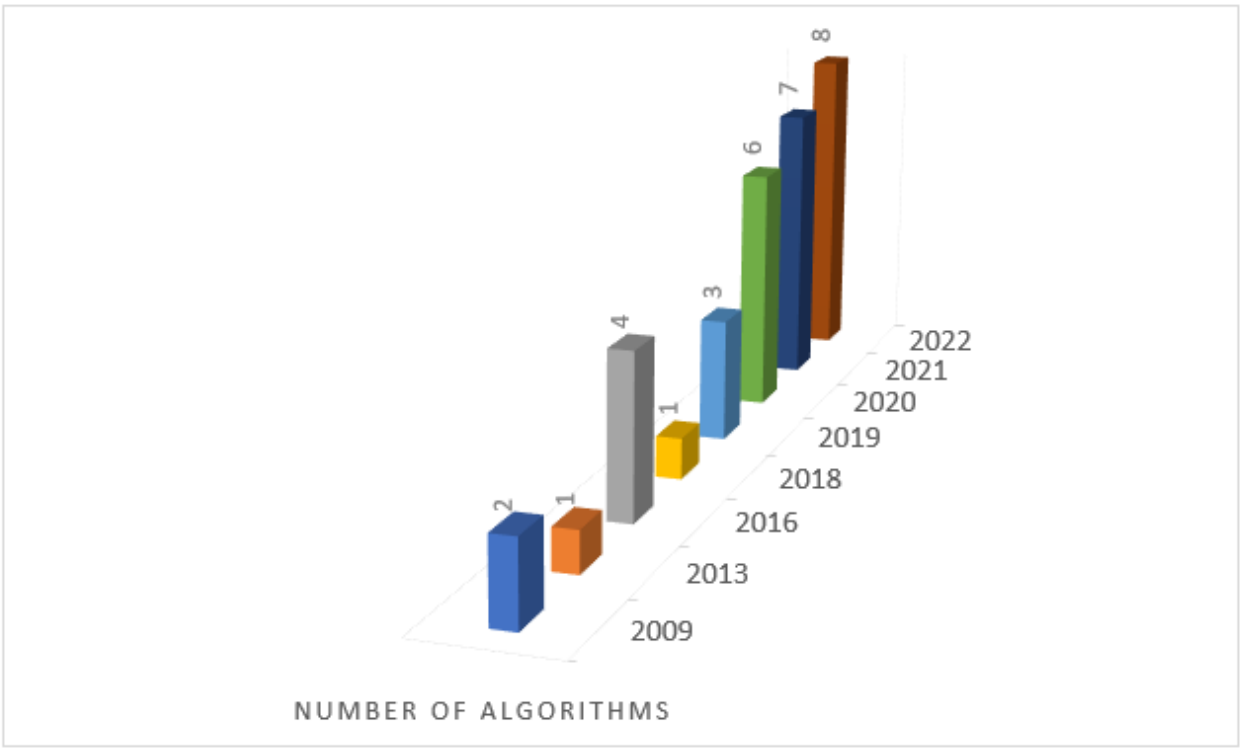

Figure 6

Annual number of Q-learning-based metaheuristic optimization techniques.

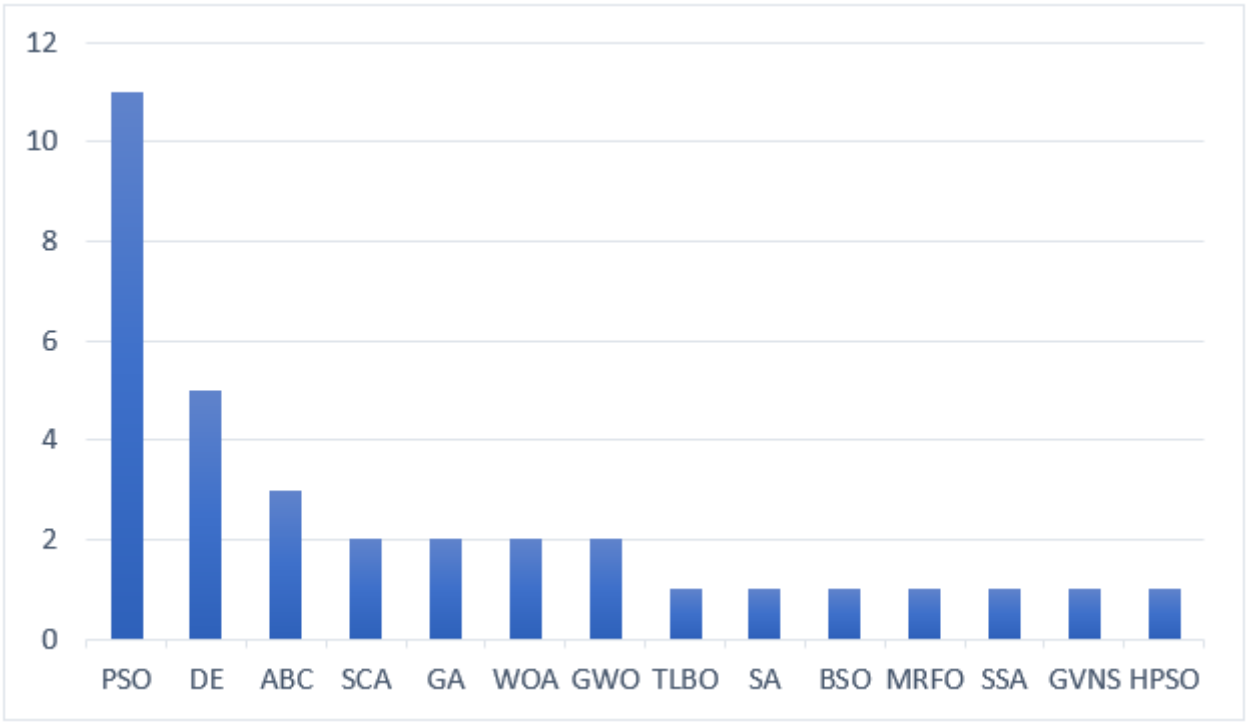

Figure 7

Number of Q-learning-based metaheuristic optimization algorithms