Methods for Solving the Traveling Salesman Problem Based on Reinforcement Learning and Metaheuristics

Eugene Fedorov and Olga Nechyporenko

Cherkasy State Technological University, Shevchenko blvd., 460, Cherkasy, 18006, Ukraine

Abstract

To date, the search for a solution to the traveling salesman problem is relevant for general and special-purpose intelligent computer systems. Currently, there is a problem of insufficient efficiency of methods for finding a solution to the traveling salesman problem. The aim of the work is to increase the efficiency of finding a solution to the traveling salesman problem through reinforcement learning based on Q-learning and SARSA, and a metaheuristic method based on the ant colony algorithm. To achieve this goal, a method based on Q-learning for the traveling salesman problem, a method based on SARSA for the traveling salesman problem, and an Ant-Q based method for the traveling salesman problem were created in the work.

The advantages of the proposed methods include the following. Firstly, the modification of the Q-learning and SARSA methods through dynamic parameters makes it possible to increase the learning rate while maintaining the mean squared error of the method. Secondly, in the Ant-Q method with dynamic parameters, the ε-greedy Q-learning approach is used to select a new vertex, which is close to random search at initial iterations, and close to directed search at final iterations. This is ensured by the use of dynamic Q-learning parameters and makes it possible to increase the learning rate while maintaining the mean squared error of the method. Thirdly, in the Ant-Q method with dynamic parameters for calculating the change in the global pheromone level at initial iterations, the pheromone increment (current reward table for Q-learning parameters) plays the main role, which ensures the breadth of the search. In the final iterations, the previous pheromone level (global reward table for Q-learning parameters) plays the main role, which ensures the convergence of the method. This is ensured by the use of dynamic Q-learning parameters and makes it possible to increase the learning rate while maintaining the mean squared error of the method.

The numerical study made it possible to evaluate the proposed methods (for the first and second methods, the number of iterations was 300; for the third method, the number of iterations was 10; for all three methods, the mean squared error was 0.05). The proposed methods make it possible to expand the scope of reinforcement learning and metaheuristics, which is confirmed by their adaptation for the specified optimization problem, and contributes to an increase in the efficiency of general and special-purpose intelligent computer systems. The prospect of further research is the study of the proposed methods for a wide class of artificial intelligence problems.

Keywords

reinforcement learning, metaheuristics, ant colony algorithm, dynamic parameters, traveling salesman problem, pheromone level

ITTAP'2022: 2nd International Workshop on Information Technologies: Theoretical and Applied Problems, November 22–24, 2022, Ternopil, Ukraine

EMAIL: fedorovee75@ukr.net (E. Fedorov); olne@ukr.net (O. Nechyporenko)
ORCID: 0000-0003-3841-7373 (E. Fedorov); 0000-0002-3954-3796 (O. Nechyporenko)

1. Introduction

To date, the development of methods aimed at solving the traveling salesman problem, which are used in general and special-purpose intelligent computer systems, is an urgent task.

Optimization methods that find the exact solution have a high computational complexity. Optimization methods that find an approximate solution through directed search have a high probability of hitting a local extremum. Random search methods do not guarantee convergence. Due to this, there is a problem of insufficient efficiency of optimization methods, which needs to be addressed.

Metaheuristics (or modern heuristics) are used to speed up finding a solution to the traveling salesman problem and reduce the probability of hitting a local extremum [1-3]. Metaheuristics expands the possibilities of heuristics by combining heuristic methods based on a high-level strategy [4-6]. The most promising metaheuristics are agent-based metaheuristics, which show the best results when searching for a solution to the traveling salesman problem [7-8].

Another popular approach is reinforcement learning [9].

Currently, there is a trend towards the joint use of reinforcement learning and metaheuristics [10].

The aim of the work is to increase the efficiency of finding a solution to the traveling salesman problem through reinforcement learning and the metaheuristic method.

To achieve this goal, it is necessary to solve the following tasks:

1. Create a Q-learning method for the traveling salesman problem.
2. Create a SARSA-based method for the traveling salesman problem.
3. Create an Ant-Q based method for the traveling salesman problem.
4. Conduct a numerical study of the proposed optimization methods.

2. Formulation of the problem

The problem of increasing the efficiency of solving the traveling salesman problem based on reinforcement learning methods (Q-learning and SARSA) and the ant colony algorithm is presented as the problem of finding such an ordered set of operators $\{A^1, A^2\}$, the iterative application of which ensures finding a solution x^*, such that $F(x^*) \to \min$, where x – is a vertex vector (route), $F(\cdot)$ – is a target function (route length), A^1 – is an operator that selects a vertex based on the ε- greedy approach, A^2 – is an operator that updates the reward table.

3. Literature review

Existing metaheuristics and reinforcement learning methods have one or more of the following advantages:

- computational complexity is lower than in traditional methods of exhaustive search (branch and bound, dynamic programming, etc.) [10];
- the probability of hitting a local extremum is lower than in gradient methods [10].

Existing metaheuristics and reinforcement learning methods have one or more of the following disadvantages:

- there is only an abstract description of the method or the description of the method is focused on solving only a specific problem [11];
- method convergence is not guaranteed [12];
- the influence of the iteration number on the process of finding a solution is not taken into account [13];
- there is no possibility to solve problems of conditional optimization [14];
- insufficient method accuracy [15].
- the procedure for determining parameter values is not automated [16].

This raises the problem of constructing efficient optimization methods.

One of the most common and accurate metaheuristics is the ant colony algorithm, which was proposed by Dorigo [18-20], further developed in [21-23] and implemented in software in [24].

One of the most common and accurate reinforcement learning methods are Q-learning and SARSA [9], which are based on the Bellman equation.

Currently, hybrid metaheuristics are often used to control search [25-26].

4. Q-learning method with dynamic parameters for the traveling salesman problem

The cost function (target function) is defined as

$$F(x) = d_{x_M, x_1} + \sum_{i=1}^{M-1} d_{x_i, x_{i+1}} \to \min_x, \quad (1)$$

where d_{x_M, x_1} – edge weight (x_i, x_{i+1}), $x_i, x_{i+1} \in V$,

x – vertices vector.

The method consists of the following steps:

1. Initialization.

1.1. The maximum number of iterations N, the length of the vertices vector M, the discrete set of states (vertices) $S = \{1,...,M\}$, the discrete set of actions (vertices) $A = \{1,...,M\}$, the matrix of edge weights $[d_{ij}]$, $i, j \in \overline{1,M}$, parameters $\rho^{\min}, \rho^{\max}$ (control the learning rate), $0 < \rho^{\min} < \rho^{\max} < 1$, parameters $\varepsilon^{\min}, \varepsilon^{\max}$ for the ε-greedy approach, $0 < \varepsilon^{\min} < \varepsilon^{\max} < 1$, parameters $\theta^{\min}, \theta^{\max}$ (determine the importance of the future reward), $0 < \theta^{\min} < \theta^{\max} < 1$, are specified.

1.2. The optimal vertices vector x^* is defined by randomly ordering the set A.

1.3. The reward table is initialized as

$$Q = [Q(i, j)],\ Q(i, j) = 0,\ i, j \in \overline{1,M}.$$

2. Iteration number is n=1.

3. Parameters are calculated:

$$\rho(n) = \rho^{\max} - (\rho^{\max} - \rho^{\min}) \frac{n-1}{N-1},$$

$$\varepsilon(n) = \varepsilon^{\max} - (\varepsilon^{\max} - \varepsilon^{\min}) \frac{n-1}{N-1},$$

$$\theta(n) = \theta^{\min} + (\theta^{\max} - \theta^{\min}) \frac{n-1}{N-1}.$$

4. The initial state (vertex) $s = 1$ is observed, which becomes a new vertex of the vertices vector, i.e. $x_1 = 1$.

5. The set of prohibited actions (vertices) $A^{tabu} = \{1\}$ is initialized.

6. An action (vertex) a is selected, to which it is necessary to move from vertex s, using an ε-greedy approach (if $U(0,1) < \varepsilon(n)$, then choose an action (vertex) a randomly from the set of allowed actions (vertices) A / A^{tabu}, otherwise choose an action (vertex) a as the nearest neighbor of vertex s from the set of allowed actions (vertices) A / A^{tabu}, i.e. $a = \arg\max_b Q(s, b)$, $b \in A / A^{tabu}$). The selected action (vertex) a is included in the set of forbidden actions (vertices) A^{tabu}, i.e. $A^{tabu} = A^{tabu} \bigcup \{a\}$, and becomes the new vertex of the vertices vector, i. e. $x_{|A^{tabu}|} = a$.

7. If there are no allowed actions (vertices) left, i.e. $A / A^{tabu} = \varnothing$, then go to step 12.

8. The current reward table element $R(s, a)$ is calculated as the negative weight of the edge (s, a), i.e. $R(s, a) = -d_{sa}$.

9. A new state (vertex) $e = a$ is observed.
10. An element of the reward table $Q(s,a)$ is calculated as

$$Q(s,a) = (1-\rho(n))Q(s,a) + \rho(n)\left(R(s,a) + \theta(n)\max_b Q(e,b)\right),\ b \in A / A^{tabu}.$$

11. Set the current state (vertex) as $s = a$. Go to step 6.
12. If the best value of the target function at the current iteration is less than the best value of the target function for all previous iterations, i.e. $F(x) < F(x^*)$, then replace the best vertices vector, i.e. $x^* = x$.
13. If not the last iteration, i.e. $n < N$, then go to step 3.

5. SARSA-based method with dynamic parameters for the traveling salesman problem

The cost function (target function) is defined as

$$F(x) = d_{x_M, x_1} + \sum_{i=1}^{M-1} d_{x_i, x_{i+1}} \to \min_x, \qquad (2)$$

where d_{x_M, x_1} – edge weight (x_i, x_{i+1}), $x_i, x_{i+1} \in V$,
x – vertices vector.

The method consists of the following steps:
1. Initialization.
1.1. The maximum number of iterations N, the length of the vertices vector M, the discrete set of states (vertices) $S = \{1,...,M\}$, the discrete set of actions (vertices) $A = \{1,...,M\}$, the matrix of edge weights $[d_{ij}]$, $i, j \in \overline{1,M}$, parameters $\rho^{\min}, \rho^{\max}$ (control the learning rate), $0 < \rho^{\min} < \rho^{\max} < 1$, parameters $\varepsilon^{\min}, \varepsilon^{\max}$ for the ε-greedy approach, $0 < \varepsilon^{\min} < \varepsilon^{\max} < 1$, parameters $\theta^{\min}, \theta^{\max}$ (determine the importance of the future reward), $0 < \theta^{\min} < \theta^{\max} < 1$, are specified.
1.2. The optimal vertices vector x^* is defined by randomly ordering the set A.
1.3. The reward table is initialized as

$$Q = [Q(i,j)],\ Q(i,j) = 0,\ i, j \in \overline{1,M}.$$

2. Iteration number is *n*=1.
3. Parameters are calculated:

$$\rho(n) = \rho^{\max} - (\rho^{\max} - \rho^{\min})\frac{n-1}{N-1},$$

$$\varepsilon(n) = \varepsilon^{\max} - (\varepsilon^{\max} - \varepsilon^{\min})\frac{n-1}{N-1},$$

$$\theta(n) = \theta^{\min} + (\theta^{\max} - \theta^{\min})\frac{n-1}{N-1}.$$

4. The initial state (vertex) $s = 1$ is observed, which becomes a new vertex of the vertices vector, i.e. $x_1 = 1$.
5. The set of prohibited actions (vertices) $A^{tabu} = \{1\}$ is initialized.
6. An action (vertex) *a* is selected, to which it is necessary to move from vertex *s*, using an ε-greedy approach (if $U(0,1) < \varepsilon(n)$, then choose an action (vertex) *a* randomly from the set of allowed actions (vertices) A / A^{tabu}, otherwise choose an action (vertex) *a* as the nearest neighbor of vertex *s* from the set of allowed actions (vertices) A / A^{tabu}, i.e. $a = \arg\max_b Q(s,b)$,

$b \in A / A^{tabu}$). The selected action (vertex) a is included in the set of forbidden actions (vertices) A^{tabu}, i.e. $A^{tabu} = A^{tabu} \bigcup \{a\}$, and becomes the new vertex of the vertices vector, i. e. $x_{|A^{tabu}|} = a$.

7. If there are no allowed actions (vertices) left, i.e. $A / A^{tabu} = \varnothing$, then go to step 13.
8. The current reward table element $R(s,a)$ is calculated as the negative weight of the edge (s,a), i.e. $R(s,a) = -d_{sa}$.
9. A new state (vertex) $e = a$ is observed.
10. An action (vertex) c is selected, to which it is necessary to move from vertex e, using an ε-greedy approach (if $U(0,1) < \varepsilon(n)$, then choose an action (vertex) c randomly from the set of allowed actions (vertices) A / A^{tabu}, otherwise choose an action (vertex) c as the nearest neighbor of vertex e from the set of allowed actions (vertices) A / A^{tabu}, i.e. $c = \arg\max_b Q(e,b)$, $b \in A / A^{tabu}$). The selected action (vertex) c is included in the set of forbidden actions (vertices) A^{tabu}, i.e. $A^{tabu} = A^{tabu} \bigcup \{c\}$, and becomes the new vertex of the vertices vector, i. e. $x_{|A^{tabu}|} = c$.
11. An element of the reward table $Q(s,a)$ is calculated as

$$Q(s,a) = (1-\rho(n))Q(s,a) + \rho(n)\big(R(s,a) + \theta(n)Q(e,c)\big).$$

12. Set the current state (vertex) as $s = a$. Set the current state (vertex) as $a = c$. Go to step 7.
13. If the best value of the target function at the current iteration is less than the best value of the target function for all previous iterations, i.e. $F(x) < F(x^*)$, then replace the best vertex vector, i.e. $x^* = x$.
14. If not the last iteration, i.e. $n < N$, then go to step 3.

6. Ant-Q method with dynamic parameters for the traveling salesman problem

The cost function (target function) is defined as

$$F(x) = d_{x_M, x_1} + \sum_{i=1}^{M-1} d_{x_i, x_{i+1}} \to \min_x, \qquad (3)$$

where d_{x_M, x_1} – edge weight (x_i, x_{i+1}), $x_i, x_{i+1} \in V$,
x – vertices vector.

The method consists of the following steps:
1. Initialization.
1.1. The maximum number of iterations N, the length of the vertices vector M, population size K, the discrete set of states (vertices) $S = \{1,...,M\}$, the discrete set of actions (vertices) $A = \{1,...,M\}$, the matrix of edge weights $[d_{ij}]$, $i, j \in \overline{1,M}$, parameters $\rho^{\min}, \rho^{\max}$ (control the learning rate), $0 < \rho^{\min} < \rho^{\max} < 1$, parameters $\varepsilon^{\min}, \varepsilon^{\max}$ for the modified ε-greedy approach, $0 < \varepsilon^{\min} < \varepsilon^{\max} < 1$, parameters $\theta^{\min}, \theta^{\max}$ (determine the importance of the future reward), $0 < \theta^{\min} < \theta^{\max} < 1$, are specified.
1.2. The optimal vertices vector x^* is defined by randomly ordering the set A.
1.3. The reward table is initialized (pheromone level) as

$$Q = [Q(i,j)],\ Q(i,j) = 0,\ i, j \in \overline{1,M}.$$

2. Iteration number is *n*=1.
3. Parameters are calculated:

$$\rho(n) = \rho^{\max} - (\rho^{\max} - \rho^{\min})\frac{n-1}{N-1},$$

$$\varepsilon(n) = \varepsilon^{\max} - (\varepsilon^{\max} - \varepsilon^{\min})\frac{n-1}{N-1},$$

$$\theta(n) = \theta^{\min} + (\theta^{\max} - \theta^{\min})\frac{n-1}{N-1}.$$

4. The initial state (vertex) $s_k = 1$ is observed, which becomes a new vertex of the vertices vector, i.e. $x_{k1} = 1$, $k \in \overline{1,K}$.

5. The set of prohibited actions (vertices) $A_k^{tabu} = \{1\}$, $k \in \overline{1,K}$ is initialized.

6. Ant number is $k = 1$.

7. Calculate the transition probabilities of the k th ant from vertex s_k to other vertices

$$p_{s_k b} = \begin{cases} \dfrac{(1-\rho(n))Q(s_k,b) + \rho(n)(1/d_{s_k b})}{\sum\limits_{l \in A \setminus A_k^{tabu}} (1-\rho(n))Q(s_k,b) + \rho(n)(1/d_{s_k b})}, & b \in A \setminus A_k^{tabu} \\ 0, & b \notin A \setminus A_k^{tabu} \end{cases}, b \in \overline{1,M}.$$

8. An action (vertex) a_k is selected, to which it is necessary to move from vertex s_k, using the modified ε-greedy approach (if $U(0,1) < \varepsilon$, then choose an action a_k, satisfying the inequality $\sum\limits_{b=1}^{a_k - 1} p_{s_k b} < U(0,1) \le \sum\limits_{b=1}^{a_k} p_{s_k b}$, from the set of allowed actions (vertices) A / A^{tabu}, otherwise choose an action (vertex) a as the nearest neighbor of vertex s from the set of allowed actions (vertices) A / A^{tabu}, i.e. $a_л = \arg\max\limits_b \{(1-\rho(n))Q(s_k,b) + \rho(n)(1/d_{s_k b})\}$, $b \in A \setminus A_k^{tabu}$). The selected action (vertex) is included in the set of forbidden actions (vertices), i.e. $A_k^{tabu} = A_k^{tabu} \bigcup \{a_k\}$, and becomes the new vertex of the vertices vector, i. e. $x_{k,|A_k^{tabu}|} = a_k$.

9. A new state (vertex) $e_k = a_k$ is observed.

10. An element of the reward table (pheromone level) $Q(s_k, a_k)$ is calculated as

$$Q(s_k,a_k) = (1-\rho(n))Q(s_k,a_k) + \rho(n)\theta(n)\max_b Q(e_k,b),\ b \in A / A_k^{tabu}.$$

11. Set the current state (vertex) as $s_k = a_k$.

12. If the current ant is not the last one, i.e. $k < K$, then increase the ant number, i.e. $k = k+1$, go to step 7.

13. If there are still allowed actions (vertices) left, i.e. $A / A_K^{tabu} \neq \varnothing$, then go to step 6.

14. If the best value of the target function at the current iteration is less than the best value of the target function for all previous iterations, i.e. $\min\limits_{k \in \overline{1,K}} F(x_k) < F(x^*)$, then replace the best vertices vector, i.e. $x^* = \arg\min\limits_{k \in \overline{1,K}} F(x_k)$.

15. The table of current rewards $R(s,a)$ is calculated as

$$R(s,a) = \sum_{k=1}^{K} \frac{\chi_{Z_{sa}}(x_k)}{F(x_k)},\ s \in \overline{1,M-1},\ a \in \overline{i+1,M},$$

$$\chi_{Z_{sa}}(x_k) = \begin{cases} 1, & x_k \in Z_{sa} \\ 0, & x_k \notin Z_{sa} \end{cases},$$

where Z_{sa} – the set of vertices vectors that contain edges (s,a) or (a,s).

16. The reward table (pheromone level) is calculated as

$$Q(s,a) = (1-\rho(n))Q(s,a) + \rho(n)R(s,a),\ s \in \overline{1,M-1},\ a \in \overline{i+1,M}.$$

17. If the current iteration is not the last one, i.e. $n < N$, then increase the iteration number, i.e. $n = n+1$, go to step 3, otherwise stop.

Comment. $U(0,1)$ – is a function that returns a uniformly distributed random number in the range $[0,1]$.

7. Experiments and results

The numerical study of the proposed optimization methods was carried out using the Python package.

For Q-learning methods, SARSA, Ant-Q with dynamic parameters, parameters values $\rho^{\min} = 0.1, \rho^{\max} = 0.9$ (control learning rate, evaporation rate coefficients), parameters values $\varepsilon^{\min} = 0.1, \varepsilon^{\max} = 0.9$ for ε-greedy approach, parameters values $\theta^{\min} = 0.1, \theta^{\max} = 0.9$ (determine importance of the future reward).

For the Ant-Q method with dynamic parameters, the ant population size is $K = 20$.

For the problem "about the shortest path in the world of tiles", the search for a solution was carried out on the standard database https://digitalcommons.du.edu/gridmaps2D, which is described in [25] (traditionally used to test methods for solving problems of finding routes in the world of tiles).

The dependence of parameter $\theta(n)$ is defined as $\theta(n) = \theta^{\min} + (\theta^{\max} - \theta^{\min})\dfrac{n-1}{N-1}$ and is shown in Figure 1.

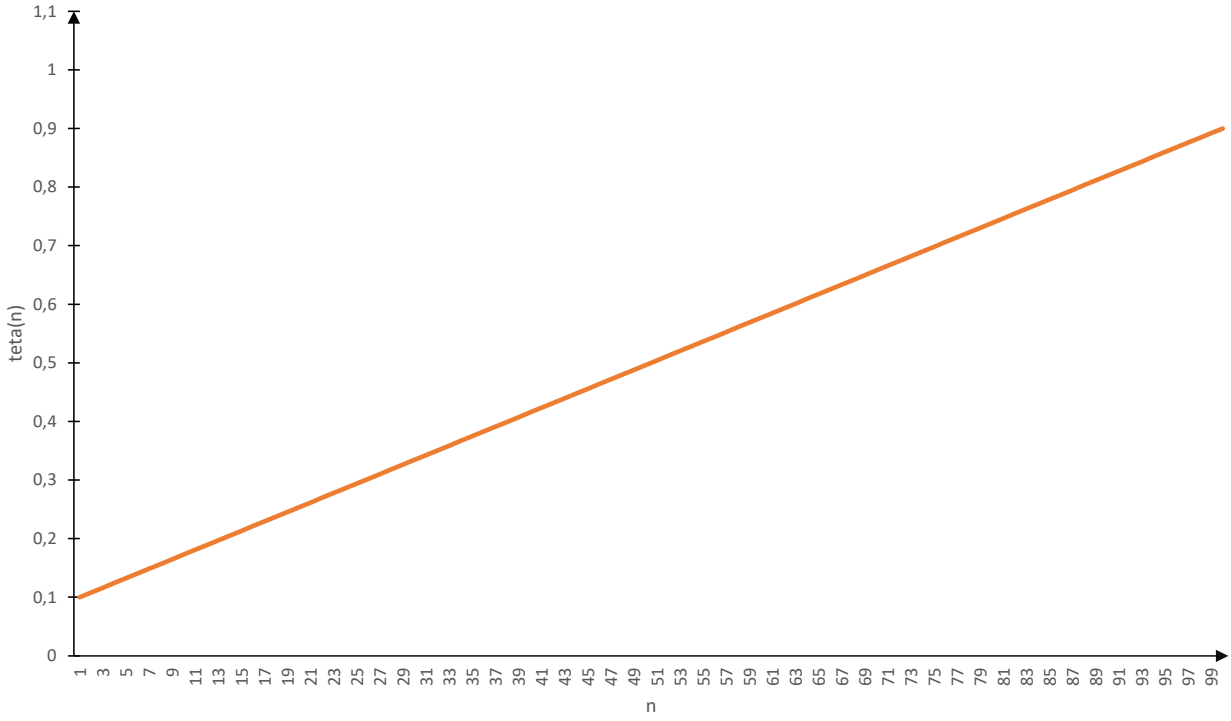

Figure 1: Dependence of parameter $\theta(n)$ on iteration number *n*

The dependence (Figure 1) of parameter $\theta(n)$ on the iteration number *n* shows that its share increases with the iteration number.

The dependence of parameters $\rho(n)$ and $\varepsilon(n)$ is determined as $\rho(n) = \rho^{\max} - (\rho^{\max} - \rho^{\min})\dfrac{n-1}{N-1}$ and $\varepsilon(n) = \varepsilon^{\max} - (\varepsilon^{\max} - \varepsilon^{\min})\dfrac{n-1}{N-1}$ and is shown in Figure 2.

The dependence (Figure 2) of parameters $\rho(n)$ and $\varepsilon(n)$ on the iteration number *n* shows that their share decreases with an increase in the iteration number.

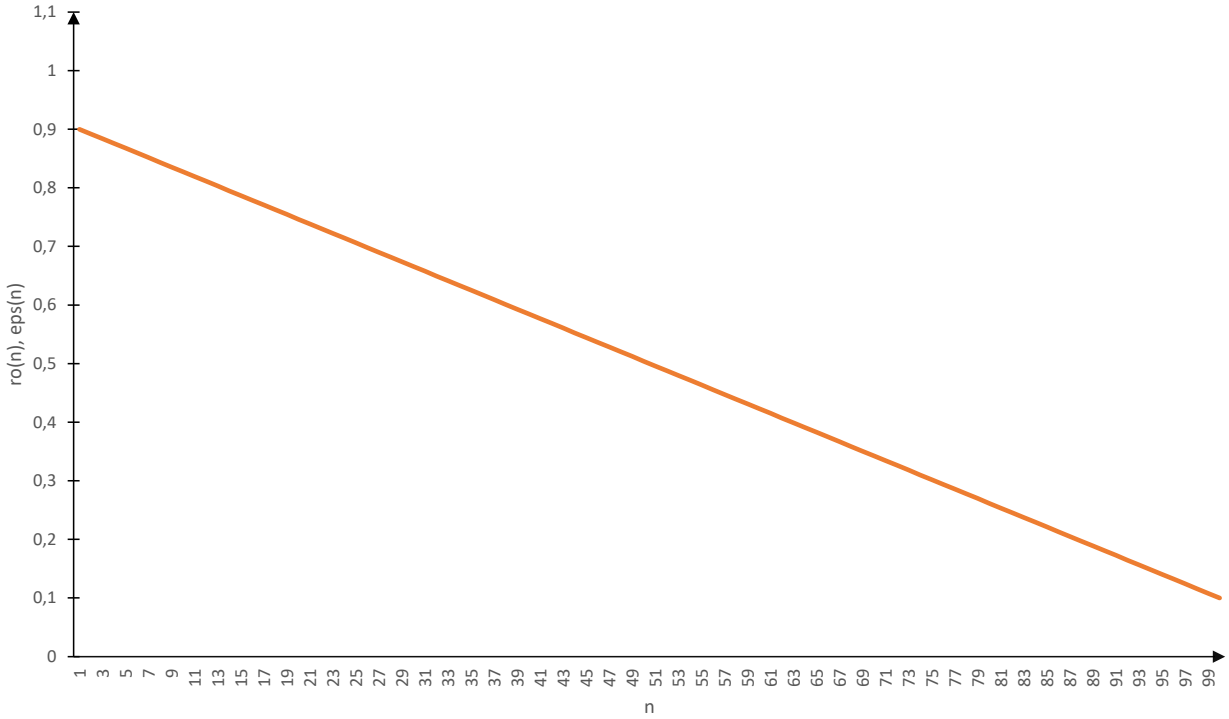

Figure 2: Dependence of parameters $\rho(n)$ and $\varepsilon(n)$ on iteration number *n*

The results of comparison of the proposed methods with the methods based on the ant colony algorithm without simulated annealing and random pheromone level, described in [16–22], are presented in Table 1.

The results of comparison of the proposed Q-learning method with dynamic parameters and with the traditional Q-learning method based on the mean squared error criterion and the number of iterations for solving the traveling salesman problem are presented in Table 1. Similar results were obtained for the proposed SARSA method with dynamic parameters and with the traditional SARSA method.

Table 1

Comparison of the proposed optimization method with the traditional Q-learning method based on the mean squared error criterion and the number of iterations for solving the «traveling salesman» problem

Method mean squared error		Number of iterations	
proposed	existing	proposed	existing
0.05	0.05	300	2000

The results of comparing the proposed Ant-Q method with dynamic parameters and with the traditional Ant-Q method based on the mean squared error criterion and the number of iterations for solving the traveling salesman problem are presented in Table 2.

Table 2

Comparison of the proposed optimization method with the traditional Ant-Q method based on the mean squared error criterion and the number of iterations for solving the «traveling salesman» problem

Method mean squared error		Number of iterations	
proposed	existing	proposed	existing
0.05	0.05	10	200

8. Discussion

Advantages of the proposed methods:
1. Modification of Q-learning and SARSA methods using dynamic parameters allows to increase the learning rate while maintaining the mean squared error of the method. The learning rate of the modified Q-learning and SARSA methods using dynamic parameters increased by about 7 times compared to the known methods.
2. In the Ant-Q method with dynamic parameters, the ε-greedy Q-learning approach is used to select a new vertex, which is close to random search at initial iterations, and close to directed search at final iterations. This is achieved by using dynamic Q-learning parameters and makes it possible to increase the learning rate while maintaining the mean squared error of the method (Table 2). The learning rate of the modified Ant-Q methods using dynamic parameters increased by 10 times, compared to the known methods.
3. In the Ant-Q method with dynamic parameters to calculate the change in the global pheromone level in the initial iterations, the pheromone increment (current reward table for Q-learning parameters) plays the main role, which ensures the breadth of the search, and in the final iterations the previous pheromone level plays the main role (global reward table for Q-learning parameters), which ensures the convergence of the method. This is achieved by using dynamic Q-learning parameters and makes it possible to increase the learning rate while maintaining the mean squared error of the method (Table 2).

9. Conclusions

1. The paper proposes a modification of the Q-learning and SARSA methods by using dynamic parameters in the reward table update rule, which makes it possible to increase the learning rate.
2. The paper proposes a modification of the Ant-Q method by using dynamic parameters in the reward table update rule and the ε-greedy vertex selection approach, which makes it possible to increase the learning rate.
3. The proposed optimization methods, due to the study of the entire search space at the initial iterations and the search directionality at the final iterations, make it possible to ensure high accuracy in solving the "traveling salesman" problem.
Practical value. The proposed methods allow expanding the scope of reinforcement learning and metaheuristics, which is confirmed by their adaptation for the specified optimization problem, and improves the efficiency of general and special-purpose intelligent computer systems.
The prospect of further research is to investigate the proposed methods for a wide class of artificial intelligence problems.

10.References

[1] X. Yu, M. Gen, Introduction to evolutionary algorithms, London: Springer-Verlag, 2010. doi: 10.1007/978-1-84996-129-5.
[2] A. P. Engelbrecht, Computational Intelligence: an introduction, Chichester, West Sussex, Wiley & Sons, 2007. doi: 10.1002/9780470512517.
[3] El-G. Talbi, Metaheuristics: from design to implementation, Hoboken, New Jersey: Wiley & Sons, 2009. doi: 10.1002/9780470496916.
[4] X.-S. Yang, Nature-inspired Algorithms and Applied Optimization, Charm: Springer, 2018. doi: 10.1007/978-3-642-29694-9.
[5] A. Nakib, El-G. Talbi, Metaheuristics for Medicine and Biology, Berlin: Springer-Verlag, 2017. doi: 10.1007/978-3-662-54428-0.
[6] F. Glover, G. A. Kochenberger, Handbook of metaheuristics, Dordrecht: Kluwer Academic Publishers, 2003. doi: 10.1007/B101874.

[7] S. Subbotin, A. Oliinyk, V. Levashenko, E. Zaitseva, Diagnostic rule mining based on artificial immune system for a case of uneven distribution of classes in sample, Communications, volume 3 (2016) 3-11.
[8] X.-S. Yang, Optimization Techniques and Applications with Examples, Hoboken, New Jersey: Wiley & Sons, 2018. doi: 10.1002/9781119490616.
[9] A. L. C. Ottoni, E. G. Nepomuceno, M. S. de Oliveira, D. C. R. de Oliveira, Reinforcement learning for the traveling salesman problem with refueling, in: Complex & Intelligent Systems, vol. 8, 2021, pp. 1-15. doi.org/10.1007/s40747-021-00444-4.
[10] M. Dorigo, L. Gambardella, Ant-Q: A reinforcement learning approach to the traveling salesman problem, in: Proceedings of ML-95, Twelfth Internet Conference on Machine Learning, 1995, pp. 252-260.
[11] C. Blum, G. R. Raidl, Hybrid Metaheuristics. Powerful Tools for Optimization, Charm: Springer, 2016. doi: 10.1007/978-3-319-30883-8.
[12] R. Martí, P. M. Pardalos, M. G. C. Resende, Handbook of Heuristics, Charm: Springer, 2018. doi: 10.1007/978-3-319-07124-4.
[13] O. Bozorg Haddad, M. Solgi, H. Loaiciga, Meta-heuristic and Evolutionary Algorithms for Engineering Optimization, Hoboken, New Jersey: Wiley & Sons, 2017. doi: 10.1002/9781119387053.
[14] M. Gendreau, J.-Y. Potvin, Gendreau M. Handbook of Metaheuristics, New York: Springer, 2010. doi: 10.1007/978-1-4419-1665-5.
[15] B. Chopard, M. Tomassini, An Introduction to Metaheuristics for Optimization, New York: Springer, 2018. doi: 10.1007/978-3-319-93073-2.
[16] J. Radosavljević, Metaheuristic Optimization in Power Engineering, New York: Institution of Engineering and Technology, 2018. doi: 10.1049/PBPO131E.
[17] K. F. Doerner, M. Gendreau, P. Greistorfer, W. Gutjahr, R. F. Hartl, M. Reimann, Metaheuristics. Progress in Complex Systems Optimization, New York: Springer, 2007. doi: 10.1007/978-0-387-71921-4.
[18] M. Dorigo, V. Maniezzo, A. Colorni, Ant system: optimization by a colony of cooperating agents, in: IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, vol. 26, issue 1, 1996, pp. 29–41. doi: 10.1109/3477.484436
[19] A. Colorni, M. Dorigo, V. Maniezzo, M. Trubian, Ant system for job-shop scheduling, in: Belgian Journal of Operations Research, Statistics and Computer Science, vol. 34, issue 1, 1994, pp. 39–53.
[20] L. M. Gambardella, É. D. Taillard, M. Dorigo, Ant colonies for the quadratic assignment problem, in: Journal of the Operational Research Society, vol. 50, issue 2, 1999, pp. 167–176. doi: 10.1057/palgrave.jors.2600676
[21] H. Min, P. Dazhi, Y. Song, An improved hybrid ant colony algorithm and its application in solving TSP, in: Proceedings of the 7th IEEE Joint International Information Technology and Artificial Intelligence Conference (ITAIC '14), 2014, pp. 423–427.
[22] K.-L. Du, M. N. S. Swamy, Search and Optimization by Metaheuristics. Techniques and Algorithms Inspired by Nature, Charm: Springer, 2016. doi: 10.1007/978-3-319-41192-7.
[23] E. Alba, A. Nakib, P. Siarry, Metaheuristics for Dynamic Optimization, Berlin: Springer-Verlag, 2013. doi: 10.1007/978-3-642-30665-5
[24] J. Brownlee, Clever algorithms: nature-inspired programming recipes, Melbourne: Brownlee, 2011.
[25] O. O. Grygor, E. E. Fedorov, T. Yu. Utkina, A. G. Lukashenko, K. S. Rudakov, D. A. Harder, V. M. Lukashenko, Optimization method based on the synthesis of clonal selection and annealing simulation algorithms, Radio Electronics, Computer Science, Control (2019) 90-99. doi: 10.15588/1607-3274-2019-2-10.
[26] E. Fedorov, V. Lukashenko, T. Utkina, A. Lukashenko, K. Rudakov, Method for parametric identification of Gaussian mixture model based on clonal selection algorithm, in: CEUR Workshop Proceedings, vol. 2353, 2019. pp. 41-55.
[27] N. R. Sturtevant, Benchmarks for Grid-Based Pathfinding, in: IEEE Transactions on Computational Intelligence and AI in Games, vol. 4, issue 2, 2012, pp. 144–148. doi:10.1109/TCIAIG.2012.2197681.