Expert Systems With Applications 173 (2021) 114663

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Reinforcement learning approach for resource allocation in humanitarian logistics

Lina Yu [a], Canrong Zhang [b], Jingyan Jiang [b], Huasheng Yang [c,*], Huayan Shang [a]

[a] School of Management and Engineering, Capital University of Economics and Business, Beijing 100070, China
[b] Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen 518055, China
[c] Department of Industrial Engineering, Tsinghua University, Beijing 100084, China

ARTICLE INFO

Keywords:
Humanitarian logistics
Resource allocation
Reinforcement learning
Q-learning

ABSTRACT

When a disaster strikes, it is important to allocate limited disaster relief resources to those in need. This paper considers the allocation of resources in humanitarian logistics using three critical performance indicators: efficiency, effectiveness and equity. Three separate costs are considered to represent these metrics, namely, the accessibility-based delivery cost, the starting state-based deprivation cost, and the terminal penalty cost. A mixed-integer nonlinear programming model with multiple objectives and multiple periods is proposed. A Q-learning algorithm, a type of reinforcement learning method, is developed to address the complex optimization problem. The principles of the proposed algorithm, including the learning agent and its actions, the environment and its states, and reward functions, are presented in detail. The parameter settings of the proposed algorithm are also discussed in the experimental section. In addition, the solution quality of the proposed algorithm is compared with that of the exact dynamic programming method and a heuristic algorithm. The experimental results show that the efficiency of the algorithm is better than that of the dynamic programming method and the accuracy of the algorithm is higher than that of the heuristic algorithm. Moreover, the Q-learning algorithm provides close to or even optimal solutions to the resource allocation problem by adjusting the value of the training episode K in practical applications.

1. Introduction

Humanitarian logistics, which are of great importance, have long been an international hotspot. Saving lives and alleviating the human suffering of survivors are the most important goals of humanitarian logistics (Das & Hanaoka, 2014). The issue of resource allocation is key to humanitarian logistics and is directly related to the survival of survivors and the effectiveness of disaster relief operations. On the one hand, a large number of survivors may be waiting for resource assistance in affected areas. These people may also be in urgent need of resource assistance because of their deteriorating health conditions. On the other hand, critical resources such as water and food often become unavailable because they are either destroyed by the disaster or quickly consumed during the response process (PPérez-Rodríguez & Holguín-Veras, 2015). To make matters worse, supply of outside resources may be delayed by congestion or destroyed roads due to the poor post-disaster conditions. Therefore, in a situation of imbalance between supply and demand, it is important to allocate limited resources efficiently, effectively, and equitably to the survivors who need them most.

As (PPérez-Rodríguez & Holguín-Veras, 2015) notes, the resource allocation problem is a complex optimization problem because the allocation of supplies to demand nodes is not only the most computationally expensive task but also arguably the most difficult decision relief groups must make. This paper studies the multiperiod resource allocation problem during the critical "three-day crucial rescue period", which is the first 72 h after a disaster (Sheu, 2007; Sheu, 2014). The problem studied in this paper considers one local response center with multi-capacity rescue resources, which is responsible for allocating resources to multiple affected areas during the planning horizon. Therefore, the state scale of the resource allocation problem studied in this paper increases exponentially with the number of affected areas, the capacity of local response center, and the length of the planning period. Moreover, this paper complicates the original problem because the three performance indicators of efficiency, effectiveness and equity are

* Corresponding author.
E-mail addresses: yulina@cueb.edu.cn (L. Yu), crzhang@sz.tsinghua.edu.cn (C. Zhang), jiangjingyan@tsinghua.edu.cn (J. Jiang), yyhhss06@gmail.com (H. Yang), shanghuayan@126.com (H. Shang).

https://doi.org/10.1016/j.eswa.2021.114663
Received 25 October 2019; Received in revised form 25 January 2021; Accepted 25 January 2021
Available online 9 February 2021

considered simultaneously in the objectives. Specifically, the deprivation cost function, proposed by Holguín-Veras, Jaller, Van Wassenhove, Prez, and Wachtendorf (2012), Holguín-Veras, Prez, Jaller, Van Wassenhove, and Aros-Vera (2013) and used in this paper as an effectiveness metric, makes the problem nonlinear. As a result, the resource allocation problem studied in this paper is difficult to solve and warrants the development of an effective algorithm.

Most of the existing literature uses heuristic solution approaches to address complex optimization problems because traditional exact methods have difficulty in solving such problems. Unfortunately, existing heuristic approaches are still not sufficiently efficient to solve large-scale problems. Therefore, to solve the multiobjective, multiperiod, nonlinear resource allocation problem, this study focuses on a reinforcement learning method that can be used to model the decision making of agents who can adapt to a dynamic environment based on learning from previous knowledge (Kara & Dogan, 2018; Teo, Taniguchi, & Qureshi, 2012). Specifically, Q-learning, a typical reinforcement learning method, is proposed to solve the complex optimization problem in this paper. Q-learning is a powerful approach to complex sequential decision-making problems with large or continuous state and action spaces (Jiang, Gu, Fan, Liu, & Zhu, 2019). The resource allocation problem studied in this paper is essentially a multiperiod sequential decision problem with a large number of states and actions, which is very suitable for Q-learning. Thus, the major contribution of our research is to demonstrate that Q-learning can be used to solve the resource allocation problem in humanitarian logistics.

The novelties of this study are summarized as follows.

(1) To the best of our knowledge, this is the first study to explicitly investigate the combination of reinforcement learning and resource allocation in humanitarian logistics.
(2) A Q-learning (QL) algorithm is proposed to solve the problem, a step-by-step introduction to the algorithm is provided, and the parameter settings of the proposed algorithm are discussed in detail.
(3) A dynamic programming method and a heuristic algorithm are provided to demonstrate the validity of the QL algorithm. The experimental results show that the efficiency of the algorithm is better than that of the dynamic programming method, and the accuracy of the algorithm is higher than that of the heuristic algorithm.
(4) Suggestions are given on how to apply the QL algorithm in practical situations. In the case of $C = 1$, the proposed algorithm can obtain the optimal solution in polynomial time when the number of training episodes K is sufficiently large. In the case of $C > 1$, the QL algorithm also converges to an acceptable solution in polynomial time. Therefore, the Q-learning algorithm can be optimized by adjusting the K value and training the algorithm in advance in practical applications.

The remainder of this paper is organized as follows: Section 2 reviews previous studies related to the resource allocation problem and also summarizes recent applications of the Q-learning algorithm to the resource allocation problem. Section 3 introduces the resource allocation model to humanitarian logistics. In Section 4, the proposed Q-learning algorithm is presented step-by-step based on the basic elements. Section 5 conducts numerical experiments to compare the proposed algorithm with the dynamic programming method and a heuristic algorithm. Finally, Section 6 draws conclusions and makes some suggestions for future research.

2. Literature review

Research on the performance metrics has been reviewed thoroughly in Yu, Zhang, Yang, and Miao (2018) and Yu, Yang, Miao, and Zhang (2019). This paper focuses on the solution approaches to resource allocation problems. A summary of previous studies related to the resource allocation problem and reinforcement learning is presented in the following.

Table 1
Classification of previous work on resource allocation problems.

Reference	Problem	Model [a]	Solution method
Fiedrich et al. (2000)	Resource allocation	A dynamic optimization model	Heuristic algorithm: SA and TS
Sheu et al. (2005)	Resource allocation	A vehicle assignment model	Fuzzy clustering
Gong and Batta (2007)	Ambulance allocation and reallocation	A deterministic model	Heuristic algorithm
Zhang et al. (2012)	Resource allocation	MIP	Heuristic algorithm
Jacobson et al. (2012)	Resource allocation	A Markovian model	Heuristic algorithm
PPérez-Rodríguez and Holguín-Veras (2015)	inventory-allocation-routing	MINLP	Heuristic algorithm
Xiang and Zhuang (2016)	Resource allocation	A queueing model	Heuristic algorithm
Hu et al. (2016)	Resource allocation	A robust model	Heuristic algorithm: PSO
Zhou et al. (2017)	Resource scheduling	MILP	Heuristic algorithm
Cook and Lodree (2017)	Vehicle dispatching	A SDP model	Heuristic algorithm
Cao et al. (2018)	Relief distribution	MINLP	Heuristic algorithm: GA
Loree and Aros-Vera (2018)	Location-allocation	MINLP	Heuristic algorithm
Shavarani (2019)	Location-allocation	a facility location model	Heuristic algorithm: GA
Wex et al. (2014)	Rescue unit scheduling and assignment	MINLP	Branch-and-bound solver, Heuristic algorithm
Yu et al. (2018)	Resource allocation	MINLP	Dynamic programming, Heuristic algorithm
Yu et al. (2019)	Resource allocation	MINLP	Dynamic programming, Heuristic algorithm Approximate dynamic programming
This paper	Resource allocation	MINLP	Dynamic programming, Heuristic algorithm Q-learning

[a] MIP: Mixed-integer programming model. MINLP: Mixed-integer nonlinear programming model. SDP: Stochastic dynamic programming. SA: Simulated annealing. MILP: Mixed-integer linear programming model. TS: Tabu search. GA: Genetic algorithm. PSO: particle swarm optimization algorithm.

2.1. Resource allocation problems in humanitarian logistics

Resource allocation problems exist in all aspects of humanitarian logistics. Many scholars study resource allocation problems together with many other issues, such as facility location, inventory management, and vehicle routing. However, the resource allocation problem is a complex and challenging problem, especially when simultaneously considering three performance metrics within multiple periods. Therefore, the issue of resource allocation as a standalone problem deserves to be studied separately. As a result, this paper focuses on this problem and the related solution methods.

Due to the complexity of the resource allocation problem, exact algorithms can be solved optimally only in small-scale cases. Therefore, most current studies on resource allocation problems use heuristic

algorithms, as shown in Table 1. Fiedrich, Gehbauer, and Rickers (2000) developed a dynamic combinatorial model to find the best assignment of machines and equipment to perform tasks with the goal of minimizing the total number of fatalities during the response phase. Both simulated annealing and tabu search algorithms were proposed to solve the model. Sheu et al. (2005) utilized a fuzzy clustering method to solve a disaster resource allocation problem. Gong and Batta (2007) constructed a deterministic model to study the ambulance allocation and reallocation problem. They presented two iterative procedures to optimize the makespan and the weighted total flow time. Zhang, Li, and Liu (2012) studied a multiple-resource multiple-depot relief allocation problem considering secondary disasters. The problem was solved by a linear programming-based algorithm and local search techniques with a high-possibility high-priority rule. Jacobson, Argon, and Ziya (2012) formulated a priority assignment problem to investigate the trade-off among demand urgency, rescue rewards and service times in allocating emergency resources, which was solved using sample-path methods and stochastic dynamic programming. PPérez-Rodríguez and Holguín-Veras (2015) developed an inventory-allocation-routing model for the optimal assignment of critical supplies to minimize social costs, designed suitable heuristic solution approaches, and assessed the performance of the heuristics using numerical experiments. Xiang and Zhuang (2016) studied a medical resource allocation problem considering deteriorating health conditions and proposed a novel queueing network model to maximize the service rate. A simple heuristic algorithm was used to find the optimal resource allocation plans. Hu, Liu, and Hua (2016) considered an emergency resource allocation problem that involved multiple competing affected areas and one relief resource center and formulated a biobjective robust model to maximize both efficiency and fairness. They developed a novel emergency resource allocation decision method to obtain decision makers' most preferred allocation policy. Zhou, Liu, Zhang, and Gan (2017) designed a multi-objective optimization model for multiperiod dynamic emergency resource scheduling problems. They proposed a multiobjective evolutionary algorithm based on decomposition to solve the model. Cook and Lodree (2017) investigated a two-stage relief chain consisting of a single staging area where donations arrive over time in uncertain quantities and are periodically distributed to random numbers of disaster survivors located at a point of distribution. The goal was to determine vehicle dispatching policies that minimize unsatisfied demand at the point of distribution. They produced 2 graphs to illustrate the difference between a heuristic solution's objective function value and that of the optimal solution. Cao, Li, Yang, Liu, and Qu (2018) formulated a multiobjective mixed-integer nonlinear programming model to maximize the lowest victims' perceived satisfaction and minimize the largest deviation in victims' perceived satisfaction for all demand points and subphases. They proposed a genetic algorithm to solve the relief distribution model. Loree and Aros-Vera (2018) presented a mathematical model to solve the problem of locating points of distribution to reach survivors and allocating inventory for local distribution in post-disaster humanitarian logistics. A heuristic approach based on the heuristic concentration integer method was proposed to solve the model. Shavarani (2019) studied an edge-based stochastic immobile facility location problem to find the optimum number of relief centers and refuel stations and their locations, with the objective of minimizing the total costs, travel distances and lost demands. A hybrid genetic algorithm was proposed to solve the model.

Few studies have adopted both exact algorithms for small-scale problems and heuristics for larger-scale instances, as shown in Table 1. Wex, Schryen, Feuerriegel, and Neumann (2014) addressed the rescue unit scheduling and assignment problem, which was presented as a generalization of the parallel-machine scheduling problem. They found that even small-scale instances cannot be solved optimally by the simple branch-and-bound solver in a practically reasonable time. Therefore, they proposed and computationally compared several heuristics, including a Monte Carlo-based heuristic, the joint application of 8 construction heuristics and 5 improvement heuristics, and GRASP metaheuristics. Yu et al. (2018) studied a multiperiod resource allocation problem in emergency logistics considering human suffering in terms of its economic evaluation representation. A dynamic programming model was proposed to directly solve the small-scale nonlinear problem optimally, and a heuristic delivery pattern was implemented to solve larger-scale instances. Yu et al. (2019) studied a dynamic capacitated resource allocation problem simultaneously considering efficiency, effectiveness, and equity metrics. A dynamic programming model was formulated, and an approximate dynamic programming method was proposed to overcome the computational difficulty created by the curse of dimensionality.

While most of the aforementioned studies use traditional heuristics, this paper explores the optimal solution to the problem using an exact algorithm and the problem solution using another approach from the field of machine learning. Therefore, this paper explores a new technique to address a problem that is typically handled using operations research techniques.

Table 2
Reinforcement learning applied to resource allocation problems.

Application field	Problem	Existing studies	Solution method
Communicaiton field	Channel allocation	Aihara et al. (2019)	Q-learning
	Channel allocation	Ye et al. (2019)	Deep Q-learning
	Wireless networks	Santos (2017)	Q-learning
	The 5G networks	Kim et al. (2019)	Q-learning
Computational resource allocation	Cloud computing	Du et al. (2019)	Deep reinforcement learning
	Cloud computing	Vengerov (2007)	Temporal difference learning
	Mobile edge computing	Yang et al. (2018)	Deep Q-learning
Smart City	Transportation	Jiang et al. (2019)	Q-learning
	Transportation	Jiang et al. (2018)	Q-learning
	Transportation	Ye et al. (2019)	Deep Q-learning
	Transportation	Šemrov et al. (2016)	Q-learning
	Transportation	Khadilkar (2018)	Q-learning
	Urban water resource management	Ni et al. (2013)	Q-learning
	Inventory management	Kwon et al. (2008)	Case-based reinforcement learning
	Inventory management	Jiang and Sheng (2009)	Case-based reinforcement learning
	Inventory management	Kara and Dogan (2018)	Q-learning
Emergency management	Deploy emergency infrastructure	Klaine et al. (2018)	Q-learning
	Rescue path selection	Su et al. (2011)	Q-learning
	Prediction of relief demand	Nadi and Edrissi (2016)	Q-learning
	Prediction of relief demand	Nadi and Edrisi (2017)	Q-learning

2.2. Reinforcement learning method applied to other related allocation problems

Reinforcement learning (RL) is an approach to machine intelligence that has emerged as a powerful tool for solving complex sequential decision-making problems in control theory (Gosavi, 2009). RL can be used to model the decision making of agents who can adapt to a dynamic environment (Teo et al., 2012). Harmon and Harmon (1997) stated that reinforcement learning combines principles of both dynamic programming and supervised learning and is successful in solving problems that either of these disciplines cannot solve individually.

Reinforcement learning has been applied to resource allocation problems in various fields, such as communication (channel allocation Aihara, Adachi, Takyu, Ohta, & Fujii, 2019; Ye, Li, & Juang, 2019, wireless networks Santos, 2017, 5th generation (5G) networks Kim, Kim, & Lim, 2019), computational resource allocation (cloud computing Du, Wu, & Huang, 2019; Vengerov, 2007, mobile edge computing Yang, Hu, Gursoy, Schmeink, & Mathar, 2018), and smart cities (transportation Jiang et al., 2019; Jiang, Fan, Liu, Zhu, & Gu, 2018; Ye et al., 2019; Šemrov, Marsetič, Žura, Todorovski, & Srdic, 2016; Khadilkar, 2018, urban water resource management Ni, Liu, Ren, & Yang, 2013, inventory management Kwon, Kim, Jun, & Lee, 2008; Kara & Dogan, 2018), as shown in Table 2. As an important algorithm in reinforcement learning, Q-learning has effectively been used to solve many problems. Q-learning is a typical reinforcement learning method for agents to learn how to act optimally in a controlled Markovian domain (Watkins & Dayan, 1992). It is an off-policy method where the agent learns the optimal policy without the policy being followed (Sutton & Barto, 1998).

In recent years, reinforcement learning has been increasingly used in fields closely related to humanitarian logistics, especially in transportation and inventory management. Jiang et al. (2019) proposed a coordinated optimization scheme for an urban rail transit line that combines coordinated passenger inflow control with train rescheduling strategies. They developed a novel Q-learning-based approach to this combination optimization problem. The proposed Q-learning approach provides an accurate scheme to address the problem of passenger congestion and train operation on a URT line. Jiang et al. (2018) developed a Q-learning algorithm to automatically learn when and at which stations to enforce inflow control and the optimal control rates at each control station with the aim of minimizing the safety risks at metro stations of a single urban rail transit line. Ye et al. (2019) developed a decentralized resource allocation mechanism for vehicle-to-vehicle (V2V) communications based on deep reinforcement learning, which can be applied to both unicast and broadcast scenarios. Šemrov et al. (2016) introduced a train rescheduling method based on Q-learning. The empirical results showed that Q-learning leads to rescheduling solutions that are at least equivalent and often superior to those of several basic rescheduling methods that do not rely on learning agents. Khadilkar (2018) described a Q-learning approach for scheduling trains on railway lines to define the track allocations and arrival/departure times for all trains on the line given their initial positions, priority, halt times, and traversal times while minimizing the total priority-weighted delay. Kwon et al. (2008) proposed a reinforcement learning algorithm appropriate for the nonstationary inventory control problem of a supply chain that has a large state space. Jiang and Sheng (2009) proposed a case-based reinforcement learning algorithm for dynamic inventory control in a multiagent supply-chain system. Kara and Dogan (2018) addressed the inventory management system of perishable products under random demand and deterministic lead time to minimize the total cost of a retailer using Q-learning and Sarsa algorithms.

However, few research efforts have been devoted to the integration of relief resource allocation and reinforcement learning, although reinforcement learning has been studied in humanitarian logistics, such as the deployment of emergency infrastructure (Klaine, Nadas, Souza, & Imran, 2018), the selection of rescue paths (Su, Jiang, Liang, & Zhang, 2011), and the prediction of relief demand (Nadi & Edrissi, 2016; Nadi & Edrisi, 2017). Klaine et al. (2018) proposed an intelligent solution based on Q-learning to find the best position of multiple drone small cells in an emergency scenario with the goal of maximizing the number of users covered by the relief system. Su et al. (2011) presented a path selection algorithm based on Q-learning for disaster response applications. Nadi and Edrissi (2016) proposed a stochastic programming model to evaluate real-time relief demand and network condition at the onset of a natural disaster and used reinforcement learning to optimize the proposed model with the goal of minimizing the total relief assessment time. Nadi and Edrisi (2017) proposed a Markov decision process as a multiagent assessment and response system, with reinforcement learning designed to ensure the integration of emergency response team and relief assessment team operations.

In summary, neither reinforcement learning nor Q-learning has been applied to the resource allocation problem in humanitarian logistics, which is a complex sequential decision-making problem. Due to the unique characteristics of resource allocation in humanitarian logistics, it cannot be easily compared to other related allocation problems. The disaster relief resource allocation problem has a large state space and action space and is well suited to a reinforcement learning approach, especially Q-learning. Q-learning, as a powerful method for solving complex sequential decision-making problems with large or continuous state and action spaces (Jiang et al., 2019), is suitable for solving the optimization problem in this paper. Therefore, a Q-learning algorithm that is suitable for humanitarian logistics scenarios and that can both maximize the long-term rewards and make dynamic decisions in the short term should be developed.

3. Problem description and modeling

This section introduces the characteristics of the resource allocation problem in humanitarian logistics in preparation for the subsequent application of the Q-learning algorithm.

3.1. Resource allocation problem in humanitarian logistics

Resource allocation in humanitarian logistics focuses on how to allocate the most needed relief resources to the neediest survivors in the shortest possible time. There are three basic elements involved in this problem.

(i) The first is the time element. The allocation of resources is a multiperiod issue. When a disaster occurs, the local response center should make a T time period resource allocation plan for resources as soon as possible. Consistent with Sheu (2014), this paper focuses on the "critical 72-h period" after a disaster. This paper assumes that there is only one local response center, which is responsible for allocating the limited disaster relief resources to $|N|$ affected areas during the T time periods.

(ii) The second element is the capacity of the local response center to distribute the amount of supplies. This depends on the resource reserves in the local response center and the allocation decision Y_t of the response center. Due to poor post-disaster conditions, the supply of relief resources of the local response center is assumed to be limited to C units per time period. The decision Y_t $(Y_t = (Y_{1,t}, Y_{2,t}, \ldots, Y_{|N|,t}), Y_t \leqslant C)$ of the local response center is the allocation plan for the amount of resources to be allocated to each affected area in each time period.

(iii) The third and most critical element is the human element. The survivors in each affected area are assumed to be homogeneous, which means that the survivors have the same demand D for resources in each time period. The goal of the allocation plan is to save lives and alleviate the human suffering of all survivors in the affected areas (Das & Hanaoka, 2014). How the effectiveness of

this allocation is measured is of particular importance. Referring to Yu et al. (2018) and Yu et al. (2019), three critical performance metrics, namely, efficiency, effectiveness and equity, are considered to evaluate the performance of humanitarian logistics. The effectiveness metric, in particular, is measured based on the deprivation cost, an economic valuation of human suffering, proposed by Holguín-Veras et al. (2012) and Holguín-Veras et al. (2013). The state of the affected area is related to the deprivation cost, defined as S_t ($S_t = (S_{1,t}, S_{2,t}, \ldots, S_{|N|,t})$), which represents the duration of deprivation endured. Changes in the state of the affected area are related to the amount of goods received and the demands of the affected area. It is assumed that at the beginning of the time horizon, all affected areas are assumed not to have experienced any suffering, i.e., the states of all affected areas at the beginning of the planning horizon are set to 0. With the help of the definition of state, the allocation decision of the local response center can be driven in reverse.

The resource allocation problem which is one response center serving multiple affected areas during multiple periods has been studied in the relevant papers (Yu et al., 2018; Yu et al., 2019). However, differing from the problem studied in Yu et al., 2018, this paper expanded the capacity from one-unit to multiple units. Compared to the problem introduced in Yu et al. (2019), this paper expaned the demand of affected area in each time period from one-unit to multiple units. In other words, the demand of each affected area is not limited to one-unit and each affected area has the same demand D, where $D \geqslant 1$.

As a result, this paper investigates a multi-period multi-capacity resource allocation problem while considering three performance metrics for the post-disaster phase, namely, efficiency, effectiveness and equity. Few studies consider multiple periods and three performance metrics simultaneously. Due to the complexity of the problem, this paper introduces a reinforcement learning method to address the problem. Therefore, this paper has research value in terms of both modeling and the algorithm.

3.2. Objectives

Based on the above analysis, three performance metrics are considered in the relief resource allocation decision process, namely, efficiency, effectiveness and equity. The goal of this paper is to allocate limited resources efficiently, effectively and fairly to the survivors who need them most. Three costs are proposed to represent the metrics, i.e., the accessibility-based delivery cost, the starting state-based deprivation cost, and the terminal penalty cost.

In this paper, the accessibility-based delivery cost (AC) represented by $\Omega(c_i Y_{i,t})$ is considered to evaluate the efficiency, where c_i stands for the unit accessibility-based delivery cost, which measures the difficulty of transporting supplies. The starting state-based deprivation cost (SSDC) is represented by $\Gamma(S_{i,t})$ and represents the deprivation costs during the planning period, where $S_{i,t}$ is the starting state of affected area i in time period t. If $S_{i,t}$ is negative, the absolute value of $S_{i,t}$ represents the number of time periods for which the affected area i can satisfy demand using the resources on hand, starting from the current time period t. If $S_{i,t}$ is nonnegative, it represents the number of time periods for which the affected area i has consecutively endured $S_{i,t}$ time periods of deprivation from the depletion of the last replenishment to the current time period t. To evaluate the equity of survivors after the planning horizon, the terminal penalty cost (TPC) is used to ensure that none of the survivors will be severely disadvantaged and that all survivors have relatively reasonable and fair conditions when facing future challenges beyond the planning horizon. $\Theta(S_{i,T+1})$ is used to depict the terminal penalty cost, where $S_{i,T+1}$ is the final state of affected area i at the end of time period T. Readers interested in details about the costs should refer to Yu et al. (2018) and Yu et al. (2019). The objective of the problem is to minimize the weighted total costs, as follows.

$$\min \quad \xi_1 \times \sum_{i=1}^{|N|} \sum_{t=1}^{T} c_i Y_{i,t} + \xi_2 \times \sum_{i=1}^{|N|} \sum_{t=1}^{T} \Gamma(S_{i,t}) + \xi_3 \times \sum_{i=1}^{|N|} \Theta(S_{i,T+1}), \tag{1}$$

where $\{\xi_1, \xi_2, \xi_3\}$ are three weights used to balance the impact of the three costs in the objective function, and the weights are set to $\{\xi_1, \xi_2, \xi_3\}=\{1/3,1/3,1/3\}$ by default unless otherwise specified; these settings are based on Huang, Jiang, Yuan, and Zhao (2015).

Due to the introduction of deprivation cost, the terms of the starting state-based deprivation cost and the terminal penalty cost are nonlinear, leading to a nonlinear objective function.

3.3. Constraints

In this section, two pivotal constraints are introduced. One is the capacity constraint. The capacity of the local response center in each time period is assumed to be C units of resources. The decision of allocation $Y_{i,t}$ represents the quantity of resources sent to affected area i in time period t. Notably, the value of $Y_{i,t}$ can be $\{0,1,2,\ldots C\}$. The capacity constraint shown in Eq. (2) cannot be violated.

$$\sum_{i=1}^{|N|} Y_{i,t} \leqslant C, \qquad \forall t = 1, 2, \ldots, T. \tag{2}$$

The other constraint is the state transition constraint. If affected area i receives $Y_{i,t}$ units of resources, its state reduces by $Y_{i,t} - D_{i,t}$. If affected area i does not receive the resources, its state increases by $D_{i,t}$, where $Y_{i,t} = 0$. Therefore, the state transition equation can be summarized by Eq. (3).

$$S_{i,t+1} = S_{i,t} - Y_{i,t} + D_{i,t}, \qquad \forall i = 1, 2, \ldots, |N|, \quad t = 1, 2, \ldots, T. \tag{3}$$

In the following section, a Q-learning approach is developed to solve the nonlinear optimization model.

4. Q-learning model of resource allocation problem

This section shows how the Q-learning algorithm can be utilized to model the resource allocation problem. Q-learning is a typical reinforcement learning method. The structure of this approach generally consists of four basic elements: agent, state, action and reward. Fig. 1 depicts the agent-environment interaction in Q-learning. In each period, the learning concept in a Q-learning model occurs as follows. When the learning agent observes the state S_t of the environment in time period t, the learning agent chooses an action Y_t among all possible actions. The environment yields a numerical reward $R(S_t)$ for a chosen action. The learning agent chooses the action in each time period with either the highest reward value based on its past experience (exploitation) or a random action (exploration). The aim is to identify the optimal state-action pair that optimizes the long-run reward.

4.1. Agent and action set

In this paper, the local response center (LRC) stands for the learning agent who needs to make the allocation decision to rescue the affected areas (AAs), as shown in Fig. 1. The LRC is responsible for real-time emergency management. In each step of the Q-learning algorithm, the learning agent decides upon the action Y_t to taken based on the observation of the state S_t of the environment. Each action Y_t will result in a change in state of the affected area S_t, and each state S_t is associated with a reward $R(S_t)$. There are Y_C possible actions for the LRC to choose in

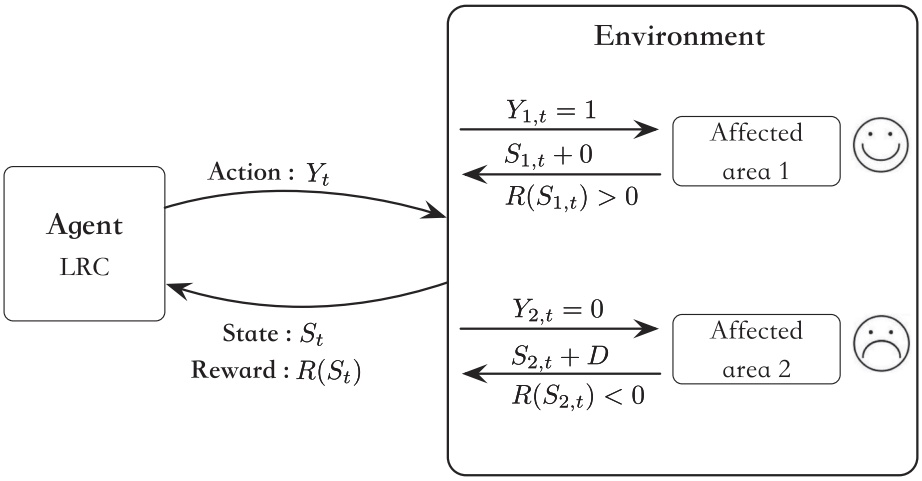


Fig. 1. Schematic diagram of Q-learning.

each step. When both C and $|N|$ are large, Y_C will be enormous because Y_C is largely dependent on these two parameters. The size of the decision space Y_C is estimated in Yu et al. (2019). The LRC can adjust the allocation decision Y_t according to the observation of the states S_t of affected areas and the rewards $R(S_t)$ fed back by the environment.

4.2. Environment and states

In this paper, the affected areas represent the environment, which interacts with the learning agent (LRC) by their changing states. It is assumed that at the beginning of the planning horizon, the states of all AAs are set to zero. When a disaster occurs, because of the shortage of relief resources, the states of all AAs are increasing. The LRC (agent) must make the allocation decision (action) to rescue the AAs (environment). This action causes the states of the AAs to change, which generates costs (reward) due to different actions. As shown in Fig. 1, if affected area 1 receives resources, its state will not increase but will remain stable, and it will feed back positive rewards. On the other hand, if affected area 2 does not receive resources, its state will increase, and the negative rewards will be fed back to the agent. The LRC receives the costs of the rescue actions, which in turn affect the next allocation decision of the LRC. The goal of the Q-learning process is to find an allocation strategy that minimizes the costs of the resource allocation plan, in other words, the strategy that maximizes the reward.

In the following, the state transition of the Q-learning algorithm is described, as shown in Fig. 2. The state of the environment is denoted by S_t ($S_t = (S_{1,t}, S_{2,t}, \ldots, S_{|N|,t})$), and there are Y_C actions that can be chosen in each time period t. If the learning agent chooses action Y_t ($Y_t = (Y_{1,t}, Y_{2,t}, \ldots, Y_{|N|,t})$) in time period t, the state of the environment in the next time period $t+1$ changes to S_{t+1}. Each state S_{t+1} in time period $t+1$ also has Y_C actions to choose, and the state changes to S_{t+2} in time period $t+2$, and so forth. Therefore, the state of the affected areas can take values in {…, −3, −2, −1, 0, 1, 2, 3,…}.

4.3. Reward function

The proposed resource allocation problem aims to minimize the total costs, including the accessibility-based delivery cost (AC), the starting state-based deprivation cost (SSDC) and the terminal penalty cost (TPC). Based on the results of in the previous section, the AC, SSDC, and TPC are represented in Eqs. (4), (5) and (6). However, the reward in this paper is not simply equal to AC, SSDC and TPC. The reward function is complex because there are three kinds of states, each of which has a different reward function. In the following, a simple example is presented to illustrate this phenomenon.

$$AC = \Omega(Y_t) \tag{4}$$

$$SSDC = \Gamma(S_t) \tag{5}$$

$$TPC = \Theta(S_{T+1}) \tag{6}$$

The instance $\{2, 1, 2\}$ means that the local response center has a 1 unit capacity of relief resources in each time period to rescue 2 affected areas within 2 time periods. Fig. 3 shows the process of the state transition, in which the state is indicated by parentheses, such as $(0,0)$, and the action is indicated by square brackets, such as $[0,0]$. Initially, the starting state is assumed to be $(0,0)$. Three actions can be taken by the local response center, $\{[0,0],[1,0],[0,1]\}$, where $[0,0]$ indicates that none of the affected areas is served, $[1,0]$ indicates that affected area 1 is served, and $[0,1]$ indicates that affected area 2 is served. Correspondingly, the possible

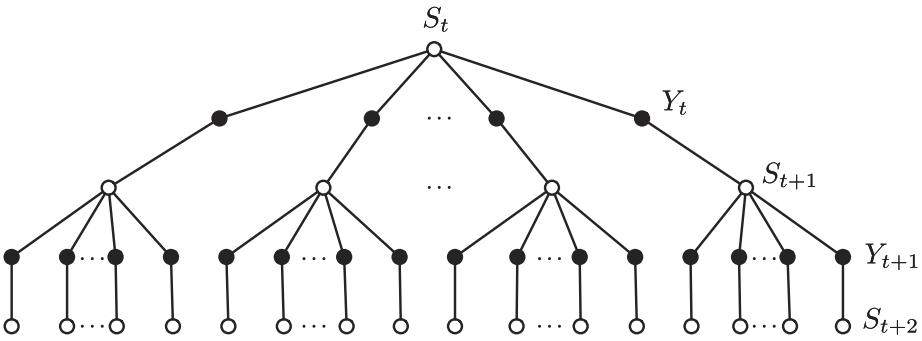


Fig. 2. The schematic diagram of state transition.

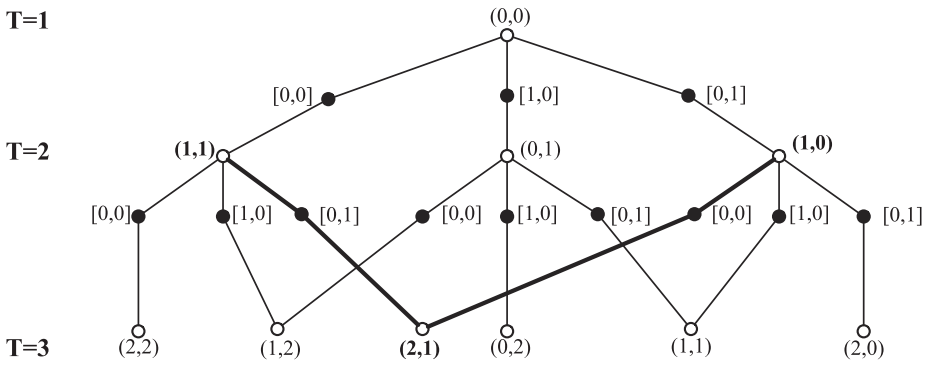


Fig. 3. Schematic diagram of state transition in the instance $\{2,1,2\}$.

states in time period 2 are $\{(1,1),(0,1),(1,0)\}$. Similarly, each state in time period 2 has three actions to choose from. Theoretically, the number of states in time period 3 should be 9, but the number of states in time period 3 is actually 6, as shown in Fig. 3, because the different states in time period 2 lead to the same state in time period 3 through different actions. Thus, the same state in time period 3 can result from different states in time period 2. For example, state $(2,1)$ in time period 3 can come either from state $(1,1)$ in time period 2 through action $[0,1]$ or from state $(1,0)$ in time period 2 through action $[0,0]$. However, the agent only cares whether the current state, such as state $(2,1)$, is good. Therefore, the rewards are related only to the state of the environment and the state that the environment is about to enter. Thus, we are not concerned with where the state originates from, only the value of the state itself, that is, which future state can be reached from the current state. Therefore, in this paper, the reward function is deliberately customized to different states.

In this paper, the states are divided into three categories: initial state, intermediate state, and final state in the planning period. The reward functions are determined as follows. First, the state of time period 1 is called the initial state of the planning period. The reward is related not only to the state of time period 1 but also to the state it is going to, i.e., the state of time period 2. Thus, the reward function for the state of time period $t=1$ is shown in Eq. (7). Second, the states from time period 2 to time period $T-1$ are called intermediate states. The reward for intermediate states corresponds to the accessibility-based delivery cost (AC) and the starting state-based deprivation cost (SSDC), as shown in Eq. (8). Finally, the reward for the final state in the final time period T corresponds to the accessibility-based delivery cost (AC) and the terminal penalty cost (TPC), as shown in Eq. (9). Notably, the goal of this study is to minimize the total costs, whereas the goal of Q-learning algorithm is to maximize the sum of rewards. Therefore, in this paper, the reward function is adjusted to be negative, such that the goal of this paper translates into maximizing the negative value of the total costs.

$$r = -\Omega(Y_1) - \Gamma(S_1) - \Gamma(S_2), \quad if \quad t=1, \tag{7}$$

$$r = -\Omega(Y_t) - \Gamma(S_{t+1}), \quad if \quad t=2,\ldots,T-1, \tag{8}$$

$$r = -\Omega(Y_T) - \Theta(S_{T+1}), \quad if \quad t=T. \tag{9}$$

4.4. Algorithm description

The basic idea of Q-learning is that the learning agent aims to find an optimal control principle that maximizes the rewards by trial-and-error iterations with its environment (Jiang et al., 2019). Q-learning requires a balance between exploitation and exploration in the strategies for selecting optimal actions and avoiding local optimum by taking different actions at random (Jiang et al., 2019). Various strategies are used to switch between the exploration and exploitation phase, and the ε-greedy policy is most commonly used (Russell & Norvig, 2003). In the ε-greedy policy, parameter ε ($0 \leqslant \varepsilon \leqslant 1$) is used to control the degree of exploration: the agent in the current training episode explores a randomly selected action with probability ε and it exploits the action with the highest utility with probability $(1-\varepsilon)$ (Šemrov et al., 2016). To balance exploration and exploitation, we used a ε-greedy policy with the exploration function Eq. (10) recommended by Šemrov et al. (2016), which ensures a high probability of exploration in the early training episodes and a high probability of exploitation in the late training episodes. The exploration function is depicted in Fig. 4:

$$\varepsilon = \frac{0.5}{1+e^{\left(\frac{10\times(k-0.4\times K)}{K}\right)}}, \tag{10}$$

where K is the total number of training episodes ($K=2000$ in Fig. 4) and and k denotes the number of the current training episode.

The Q-value is initialized to 0 at the beginning of the learning process. Then, the Q-value is updated using the Q-learning update rule given by Eq. (11). The Q-value function is used to qualify the performance and effectiveness of the specific actions and states.

$$Q(S_t^k, Y_t^k) \leftarrow Q(S_t^k, Y_t^k) + \alpha\left(R_t^k + \gamma \max_{Y_C} Q(S_{t+1}^k, Y_t^k) - Q(S_t^k, Y_t^k)\right), \tag{11}$$

where S_t^k denotes the state of the environment in time period t in training episode k, Y_t^k denotes the action chosen by the agent in time period t in training episode k, Y_C stands for the action space, α ($0 \leqslant \alpha \leqslant 1$) is the learning rate, γ ($0 \leqslant \gamma \leqslant 1$) represents the discount rate, and R_t^k expresses the reward value in time period t in training episode k.

The detailed procedure of the Q-learning algorithm is shown in

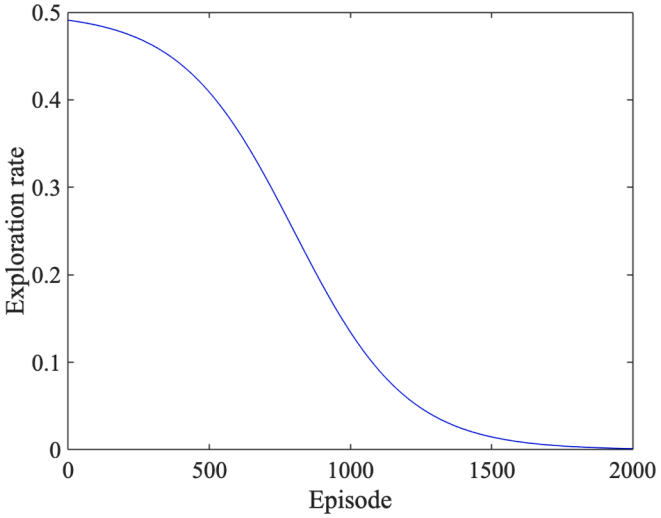


Fig. 4. The ε-greedy exploration function.

Algorithm 1:

5. Numerical experiments and comparisons of the algorithms

The Q-learning algorithm was introduced in the previous sections. This section discusses the parameter settings of the Q-learning algorithm. In addition, this section investigates the validity of the Q-learning algorithm and compares it to an exact algorithm and a heuristic algorithm. In the following, all programs are coded in MATLAB 8.6 (R2015b), and the results reported below are obtained on a PC with a 3.1 GHz CPU and 16.00 GB RAM.

5.1. Parameter settings

The convergence speed of the Q-learning algorithm is affected by three parameters, namely, the exploration rate ε, the learning rate α, and the discount factor γ. On the basis of Šemrov et al. (2016), the initial parameters are set to $\varepsilon = 0.5, \alpha = 0.8$, and $\gamma = 0.2$, and the maximum number of training episodes is set to $K = 2000$. In the following, the parameter settings are discussed by taking the instance {3,1,6} as an example, which means the local response center has 1-unit capacity of resources per period to rescue 3 affected areas within 6 time periods of the planning horizon. The deprivation parameters a and b are set to 2.04 and 0.24, respectively, referring to Pradhananga, Mutlu, Pokharel, Holguín-Veras, and Seth (2016). The accessibility cost c_i of these three affected areas and the local response center takes values in {200, 250, 300}, as in Yu et al. (2018).

Algorithm 1. Q-learning with ε-greedy policy

Generate Action set Y_C and State set S,
Initialize Q-table, e.g., $Q \leftarrow 0$,
for $k \leftarrow 1$ *to* K **do**
  Generate ε according to

$$\varepsilon = \frac{0.5}{1 + e^{\left(\frac{10\times(k-0.4\times K)}{K}\right)}},$$

  for $t \leftarrow 1$ *to* T **do**
    Choose an action according to

$$Y_t^k \leftarrow \begin{cases} Y_t^k \in \arg\max_{Y_C} Q(S_t^k, Y_t^k) & \text{with probability } 1-\varepsilon \text{ (exploit)}, \\ \text{a uniformly random action in } Y_C & \text{with probability } \varepsilon \text{ (explore)}, \end{cases}$$

    According to the current state S_t^k and current action Y_t^k, calculate the next state S_{t+1}^k and reward R_t^k,
    if $t = 1$ **then**
      $S_{t+1}^k = S_t^k - Y_t^k + D_t^k$
      $R_t^k = -\Omega(Y_t^k) - \Gamma(S_t^k) - \Gamma(S_{t+1}^k)$
    else if $t = T$ **then**
      $S_{t+1}^k = S_t^k - Y_t^k + D_t^k$
      $R_t^k = -\Omega(Y_t^k) - \Theta(S_{T+1}^k)$
    else
      $S_{t+1}^k = S_t^k - Y_t^k + D_t^k$
      $R_t^k = -\Omega(Y_t^k) - \Gamma(S_{t+1}^k)$
    Update $Q(S_t^k, Y_t^k) \leftarrow Q(S_t^k, Y_t^k) + \alpha\left(R_t^k + \gamma \max_{Y_C} Q(S_{t+1}^k, Y_t^k) - Q(S_t^k, Y_t^k)\right)$,
    Update $S_t^k \leftarrow S_{t+1}^k$,
Obtain the final Q-table Q,
Calculate the final state transition Path and Action strategy according to the final Q-table Q,
Calculate the objective value according to the final Path and Action strategy.

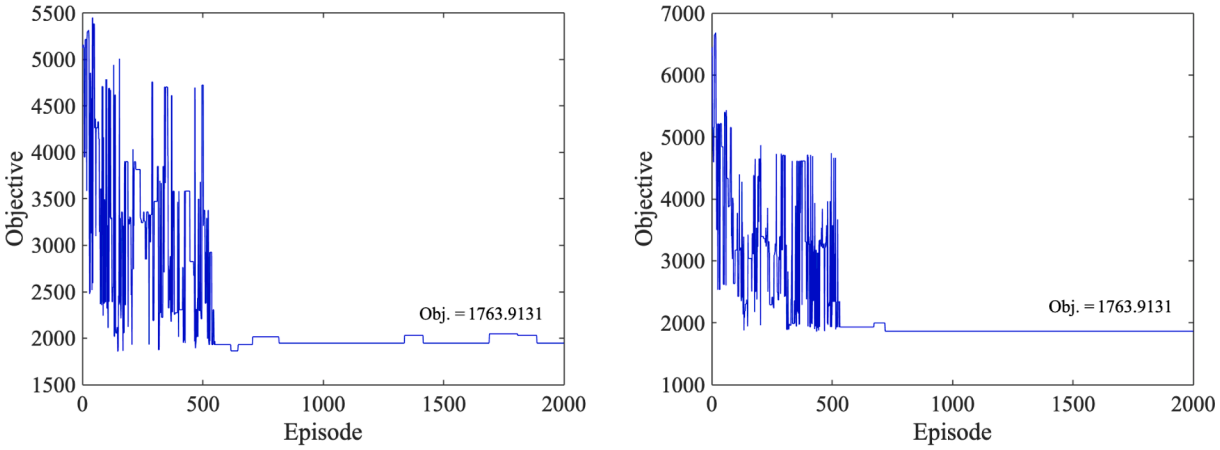
5.1.1. Exploration rate

According to the previous section, the ε-greedy policy is proposed to balance the exploration and exploitation, that is, randomly selecting an action with a probability of ε or executing the action with the highest Q value with a probability of $1-\varepsilon$. There are two ways to implement the ε-greedy policy. One is to set the exploration probability to a constant value, as in most studies, for example, $\varepsilon = 0.5$. Another is to use Eq. (10) proposed by Šemrov et al. (2016), where the exploration rate starts with a high probability and gradually decreases to zero. In the following, these two policies are compared based on the instance {3,1,6}. The results are shown in Fig. 5.

As shown in from Fig. 5, the objective values of both policies fluctuate when the training episode $k < 500$. Although the $\varepsilon = 0.5$ policy shown in Fig. 5 (a) can converge to the optimal objective value around $k = 550$, the objective value is unstable and fluctuates around the optimal value of 1763.9131. By contrast, the Eq. (10) policy shown in Fig. 5 (b) can converge to the optimal objective and remain stable, even if it requires $k = 700$ episodes. Therefore, the Eq. (10) policy outperforms the $\varepsilon = 0.5$ policy in terms of convergence to the optimal value.

5.1.2. Learning rate

The learning rate indicates the efficiency with which the agent utilizes new information. The closer the learning rate α is to 1, the more the Q-learning algorithm considers the rewards. Thus, when the learning rate α approaches 1, the agent will make a large update in the next step. In the following, this paper gradually increases the value of α from 0 to 1. To abate the impact of randomness, 10 groups of experiments are performed for each value of α. The results of the 10 groups are shown in Table 3, and the results for each group are averaged, as shown in the last



(a) $\varepsilon = 0.5$ (b) ε = Eq. (10)

Fig. 5. The convergence speed of QL with different ε-greedy policies.

column in the table. Furthermore, to present the results more clearly, the average values are depicted in Fig. 6.

As shown in Table 3, the Q-learning algorithm converges to the optimal value when α equal {0.2, 0.4, 0.6, 0.8, 1}. Fig. 6 shows that the larger α is, the more easily the algorithm reaches the optimal value. However, a larger α slows the convergence because the update step is too small. Therefore, $\alpha = 0.8$ is chosen, referring to Šemrov et al. (2016), in the following experiments.

5.1.3. Discount factor

The discount factor γ determines the present value of future rewards (Šemrov et al., 2016). When the discount factor γ is close to zero, the proposed algorithm tends to maximize immediate rewards, and when γ approaches 1, the proposed algorithm is more inclined to consider future rewards. Similar to the previous section, 10 groups of experiments are performed for each value of γ, where $\gamma = \{0, 0.2, 0.4, 0.6, 0.8, 1\}$. The results of the 10 groups are shown in Table 4, and the results for each group are averaged, as shown in the last column in the table. Furthermore, to present the results more clearly, the average values are depicted in Fig. 7.

As shown in Table 4, the proposed algorithm can converge to the optimal value when $\gamma = \{0.6, 0.8, 1\}$. Fig. 7 shows that the larger γ is, the more easily the algorithm reaches the optimal value. To balance future rewards and immediate rewards, $\gamma = 0.6$ is chosen for the following experiments.

5.2. Comparisons to other methods

To investigate the quality of solutions produced by the proposed algorithm, this paper further compares the Q-learning algorithm (QL) with an exact method, dynamic programming (DP), and a heuristic algorithm. The dynamic programming method proposed by Yu et al. (2018) is employed to handle the nonlinear model directly. On the other hand, although it is interesting to compare QL with DP, it is more meaningful to compare QL with heuristic approaches. The most common method used in disasters is the principle of proximity relief. The logic adopted by the greedy algorithm (GE) proposed by Yu et al. (2019) is to select the affected areas most in need of assistance to distribute materials in each period, which is in line with the principle of proximity assistance. Therefore, GE is regarded as a simple myopic approach in this paper. As this paper is most concerned with the effectiveness of QL, the specific algorithm steps pertaining to the dynamic programming method and the heuristic method will not be repeated. Interested readers can refer to the literature for more details.

To maintain consistency, the instances used in this section also draw on Yu et al. (2018) and Yu et al. (2019). The scale of the instances is enlarged by increasing either or both of $|N|$ and T. The results of the experiments are restricted to small-scale instances. Instances with larger state sizes are not illustrated in this paper due to the curse of dimensionality of DP, which cannot be solved in a reasonable amount of time.

The objective values and computational time are shown in Table 5. Although the computational time of GE outperforms that of the other two methods, the objective value of GE is much worse than that of DP, and GE does not yield the optimal solutions, as shown in the "Gaps" column in Table 5, By contrast, the objective values of the QL algorithm are the same as those of the DP algorithm, and the computational time of QL is better than that of DP for some large cases. Thus, QL can achieve optimal solutions, like DP, in a reasonable amount of time. The last four columns of Table 5 are the parameter settings that enable the QL algorithm to reach the optimal solutions. These settings confirm the discussion of ε, α and γ in the previous sections. The difference in parameter settings between the instances is the maximum number of training episodes that ensures convergence. The number of convergence episodes required to perform QL on different instances is different. The larger the state size of the instance is, the greater the number of episodes required to obtain the optimal solution. Furthermore, the computational time grows as the maximum number of convergence episodes increases.

The above table shows that both DP and QL, but not GE, can achieve

Table 3
The results obtained by QL as α varies.

α	1	2	3	4	5	6	7	8	9	10	Avg.[a]
0	6366.42	6366.42	6366.42	6366.42	6366.42	6366.42	6366.42	6366.42	6366.42	6366.42	6366.42
0.2	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91
0.4	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91
0.6	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91
0.8	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91
1	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91

[a] Avg.: the average of 10 experimental results.

Table 4
The results obtained by QL as γ varies.

γ	1	2	3	4	5	6	7	8	9	10	Avg.[a]
0	3450.78	3450.78	3450.78	3450.78	3450.78	3450.78	3450.78	3450.78	3450.78	3450.78	3450.78
0.2	2733.78	2733.78	2733.78	2733.78	2733.78	2733.78	2733.78	2733.78	2733.78	2733.78	2733.78
0.4	2733.78	2733.78	2733.78	2733.78	2733.78	2733.78	2733.78	2733.78	2733.78	2733.78	2733.78
0.6	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91
0.8	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91
1	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91	1763.91

[a] Avg.: the average of 10 experimental results.

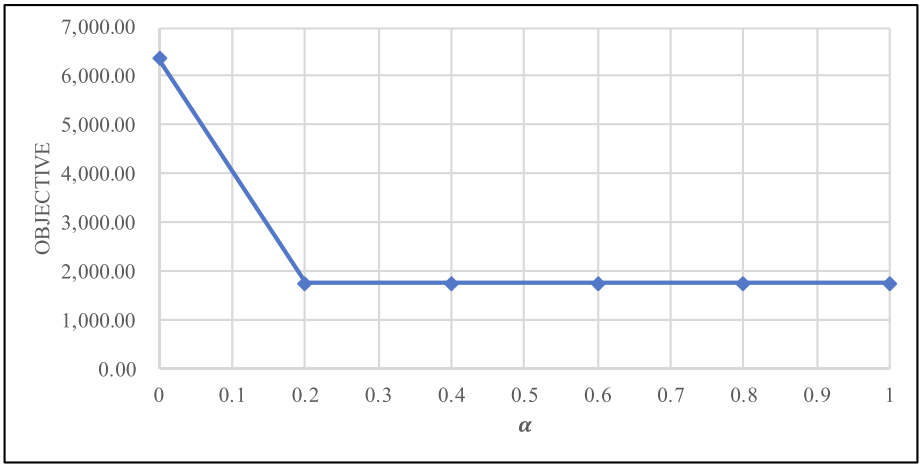


Fig. 6. The objective values obtained by QL as α increases.

the optimal solutions but with different trends in the growth of computational time. Therefore, to compare the computational time of DP, GE and QL as the instance scale increases, the following sections discuss the efficiency of these methods when changing $|N|$, T and even C. Although GE has an advantage in terms of computational time, the objective value is far from the optimal value. Therefore, in the following, the experiments focus on comparing the computational time of DP and QL.

5.2.1. Impact of $|N|$

To investigate the impact of $|N|$, three groups of instances are discussed, i.e., {$|N|$, 1, 6}, {$|N|$, 1, 9}, and {$|N|$, 1, 12}, where $|N|$ takes values in the range of 3 to 8. The computational time of these three groups is depicted in Fig. 8. In contrast to GE's stable and fast calculation time, as $|N|$ grows, the computational time of DP and QL increases. In particular, the computational time required by DP increases slowly in the first group of instances ($T = 6$), and the computational time required by QL is longer than that of DP when $T = 6$, as shown in Fig. 8 (a). In the second group of instances ($T = 9$), the computational time required by DP starts to increase as $|N|$ grows. Notably, the computational time of instance {8, 1, 9} required by DP is much longer than that of QL, as shown in Fig. 8 (b). In the third group of instances ($T = 12$), most of the instances solved by DP have a longer computational time than those solved by QL, as shown in Fig. 8 (c). Therefore, the computational time of QL is better than that of DP when $|N|$ increases, even if the solution efficiency of QL is worse than that of DP when $|N|$ is small.

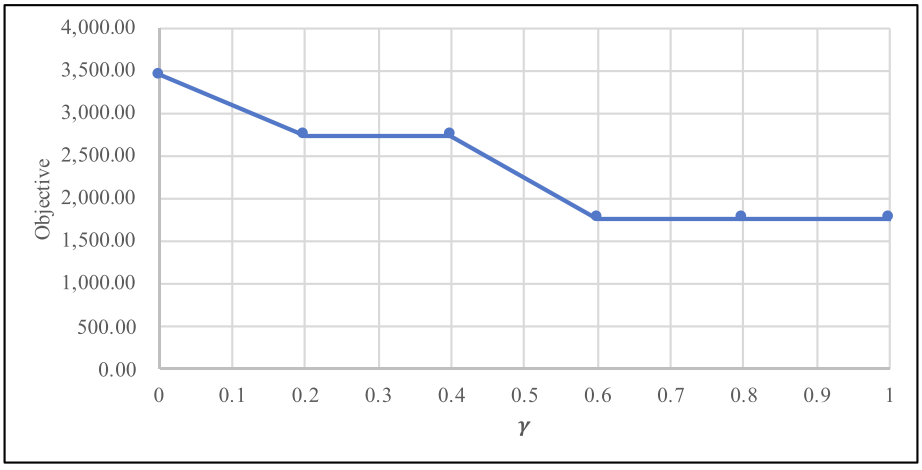
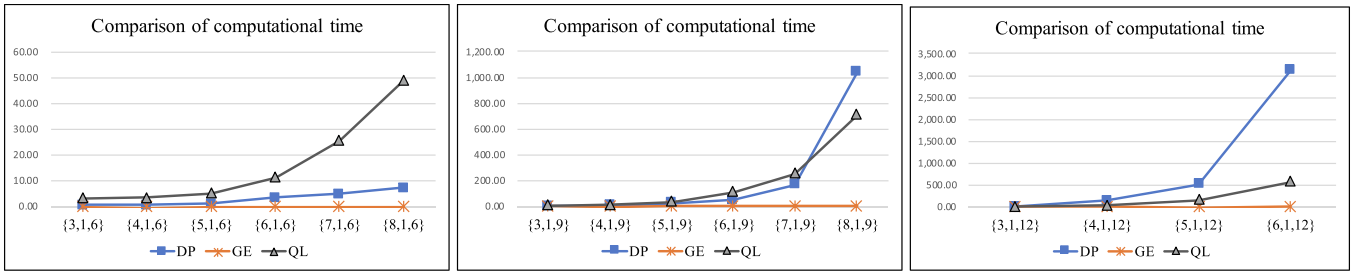


Fig. 7. The objective values obtained by QL as γ increases.



(a) $T = 6$ (b) $T = 9$ (c) $T = 12$

Fig. 8. The computational time of DP and QL as $|N|$ increases.

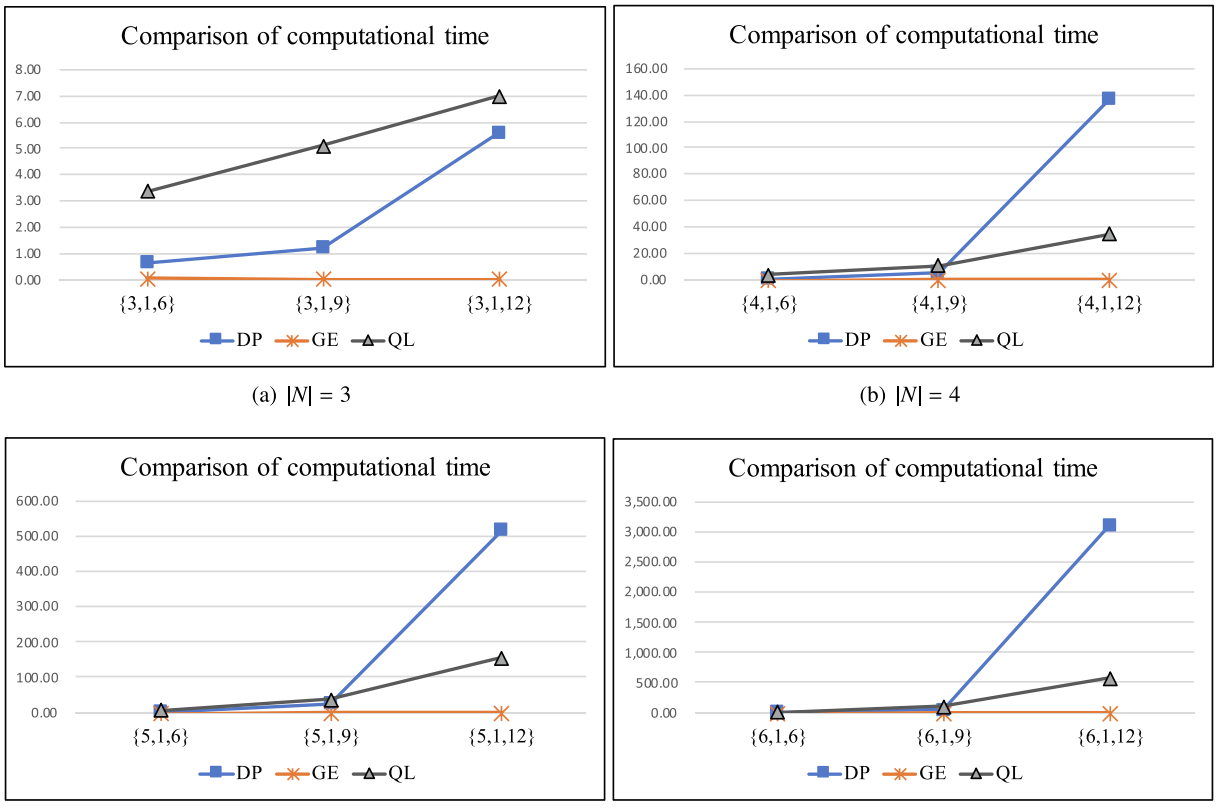
5.2.2. Impact of T

To investigate the impact of T, four groups of instances are considered, i.e., {3, 1, T}, {4, 1, T}, {5, 1, T}, and {6, 1, T}, where T takes values in 6, 9 and 12. The computational time of these groups is depicted in Fig. 9. The computational times of both DP and QL increase as T increases. In the first group of instances ($|N| = 3$), as shown in Fig. 9 (a), although the computational time required by QL is longer than that of DP, the computational time of DP increases faster than that of QL when protracting the horizon from $T = 9$ to $T = 12$. In the second, third and fourth groups of instances ($|N| = 4, |N| = 5, |N| = 6$), the computational time of DP increases rapidly as T increases, as shown in Fig. 9 (b) (c) (d). Specifically, the computational time required by DP is much longer than that of QL when $T = 12$. Therefore, the computational time of DP is better than that of QL when T is small, but the solution efficiency of QL is better than that of DP as T increases.

5.2.3. Impact of state size

The above analysis indicates that the computational time of both DP and QL increases regardless of whether $|N|$ grows or T grows. In other words, as the state size of the instance increases, the computational time of both DP and QL increases. To further verify this conclusion, the instances are sorted by their state sizes according to Table 5, and their computational times are depicted in Fig. 10.

The computational time of DP increases sharply with the growth in state size. On the other hand, the computational time of QL is insensitive to the state size. The reason for this phenomenon is that the computational time of QL is related only to the maximum number of training episodes, that is, the value of K. In particular, the proposed algorithm can solve large-scale instances that cannot be solved by the dynamic programming method. Therefore, when the DP suffers from the curse of dimensionality and cannot obtain the optimal objective, the QL can yield the optimal objective within reasonable computational time by



(a) $|N| = 3$ (b) $|N| = 4$

(c) $|N| = 5$ (d) $|N| = 6$

Fig. 9. The computational time of DP and QL when T grows.

Table 5
Comparisons of objective values and computational time ($C = 1$).

Ins.[a]	State size	Solution of DP		Solution of GE		Solution of QL		Gaps (%)		Parameters of QL			
		Obj.	Time (s)	Obj.	Time (s)	Obj.	Time (s)	GE-DP[b]	QL-DP	ε	α	γ	K
{3, 1, 6}	210	1763.91	0.66	3450.78	0.04	1763.91	3.37	95.63	0.00	0.6	0.8	Eq. (10)	2000
{3, 1, 9}	715	9458.97	1.23	23145.01	0.03	9458.97	5.10	144.69	0.00	0.6	0.8	Eq. (10)	2000
{3, 1, 12}	1820	60491.39	5.63	156023.28	0.03	60491.39	7.02	157.93	0.00	0.6	0.8	Eq. (10)	2000
{4, 1, 6}	462	3139.01	0.62	5572.92	0.01	3139.01	3.47	77.54	0.00	0.6	0.8	Eq. (10)	2000
{4, 1, 9}	2002	23068.61	5.57	47404.19	0.01	23068.61	10.29	105.49	0.00	0.6	0.8	Eq. (10)	4000
{4, 1, 12}	6188	191722.23	137.18	397671.46	0.03	191722.23	34.63	107.42	0.00	0.6	0.8	Eq. (10)	10000
{5, 1, 6}	924	4620.64	1.23	7695.06	0.02	4620.64	5.18	66.54	0.00	0.6	0.8	Eq. (10)	3000
{5, 1, 9}	5005	43281.44	24.15	73000.79	0.02	43281.44	36.11	68.67	0.00	0.6	0.8	Eq. (10)	14000
{5, 1, 12}	18,564	420379.01	518.81	756670.49	0.02	420379.01	155.93	80.00	0.00	0.6	0.8	Eq. (10)	40000
{6, 1, 6}	1716	6326.79	3.63	9817.20	0.02	6326.79	11.28	55.17	0.00	0.6	0.8	Eq. (10)	6300
{6, 1, 9}	11,440	67057.97	49.30	102036.57	0.02	67057.97	108.93	52.16	0.00	0.6	0.8	Eq. (10)	37,000
{6, 1, 12}	50,388	693754.08	3117.26	1179714.77	0.03	693754.08	573.35	70.05	0.00	0.6	0.8	Eq. (10)	136,000
{7, 1, 6}	3003	8448.93	4.98	11939.34	0.02	8448.93	25.63	41.31	0.00	0.6	0.8	Eq. (10)	12500
{7, 1, 9}	24,310	91367.16	169.42	139884.06	0.02	91367.16	253.78	53.10	0.00	0.6	0.8	Eq. (10)	92,000
{8, 1, 6}	5005	10571.07	7.28	14061.48	0.02	10571.07	48.92	33.02	0.00	0.6	0.8	Eq. (10)	23,000
{8, 1, 9}	48,620	117013.75	1039.09	177731.55	0.02	117013.75	709.28	51.89	0.00	0.6	0.8	Eq. (10)	220,000

[a] Ins.: Instances $\{|N|, C, T\}$, with $|N|$ affected areas, C capacities, and T planning periods.

[b] The gaps in the objective values between GE and DP, for example, $\frac{Obj(GE) - Obj(DP)}{Obj(DP)} \times 100\%$.

selecting an appropriate K value. As long as the value of K is sufficiently large, the algorithm can be optimized.

Although a larger K consumes more computational time, the Q-table of the algorithm during the training episode can be stored. The stored trained Q-table can be downloaded for the next training episode to drastically reduce the computational time of the Q-learning algorithm. Therefore, in practical applications, K can be set sufficiently large in advance, thereby achieving the purpose of the optimization algorithm. Based on these results, it can be concluded that QL is superior to DP in real-life applications, especially in the case of large-scale instances.

5.2.4. Impact of C

In the above sections, the numerical experiments consider the case of a 1-unit capacity of relief resources and demonstrate that QL is applicable and effective for the resource allocation problem with a capacity of 1. In addition, this paper studies the applicability of QL in multiunit capacity cases.

To investigate the impact of C, three groups of instances are discussed, i.e., {3, C, 6}, {4, C, 6} and {5, C, 6}, where C takes values in 1, 2 and 3. DP, GE and QL are employed to solve these three groups of instances, and the results are shown in Table 6. Even if C increases, GE still does not perform well in all groups of instances. On the other side, although the gaps between QL and DP for $C > 1$ cases are much larger than those of $C = 1$ cases, the performance of QL is much better than that of GE. As shown in the fifth-last column of Table 6, in all instances, the gaps between the objective values of QL and DP decrease as the number of training episodes K increases. However, as K increases to a certain level, the QL algorithm converges to a solution. In all cases where $C > 1$, when K is set to a sufficiently large value of 1,000,000, QL still fails to jump out of the local optimal solution. Fortunately, although this solution is not the global optimal solution, it is still a good solution. The maximum gap between it and the optimal solution is smaller than the gap between the best solution obtained with the GE algorithm and the optimal solution. In future studies, the search strategy of QL can be improved to jump out of the partial solution dilemma. Moreover, although the computational times of QL are longer than those of DP in some instances, QL can obtain a sufficiently good solution in polynomial time, whereas DP cannot be solved in polynomial time. Therefore, in practice, the aforementioned method can be used to train in advance with a larger K to save computational time.

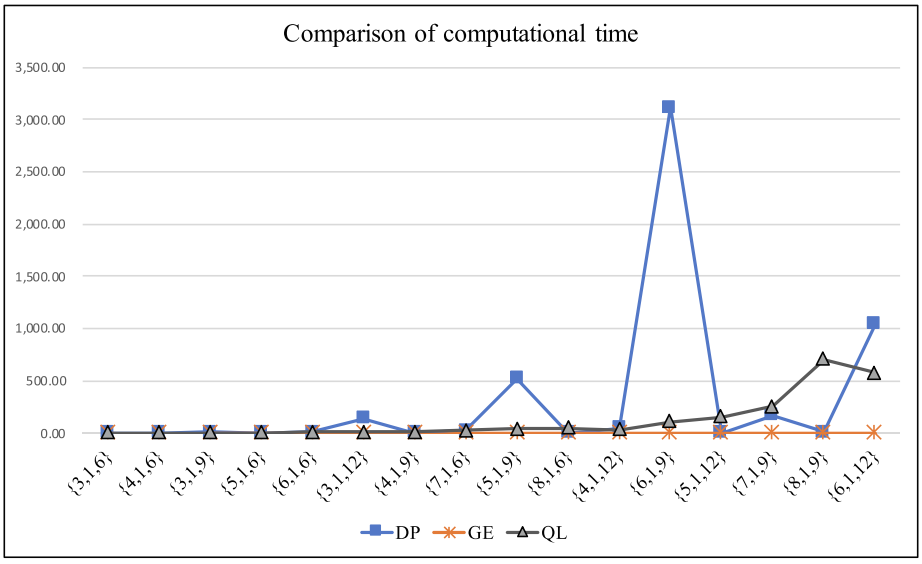


Fig. 10. The computational time of DP and QL as the state size increases.

Table 6
Comparisons of objective values and computational time ($C \geqslant 1$).

Ins.[a]	State size	Solution of DP		Solution of GE		Solution of QL		Gaps (%)		Parameters of QL			
		Obj.	Time (s)	Obj.	Time (s)	Obj.	Time (s)	GE-DP[b]	QL-DP	ε	α	γ	K
{3, 1, 6}	210	1763.91	0.66	3450.78	0.06	1763.91	3.37	95.63	0.00	0.6	0.8	Eq. (10)	2000
{3, 2, 6}	1036	1194.65	1.86	2224.29	0.04	1472.62	5.63	86.19	23.27	0.6	0.8	Eq. (10)	3000
{3, 2, 6}	1036	1194.65	1.86	2224.29	0.04	1472.62	11.12	86.19	23.27	0.6	0.8	Eq. (10)	5000
{3, 2, 6}	1036	1194.65	1.86	2224.29	0.04	1472.62	18.25	86.19	23.27	0.6	0.8	Eq. (10)	10,000
{3, 2, 6}	1036	1194.65	1.86	2224.29	0.04	1472.62	182.40	86.19	23.27	0.6	0.8	Eq. (10)	100,000
{3, 2, 6}	1036	1194.65	1.86	2224.29	0.04	1472.62	1579.82	86.19	23.27	0.6	0.8	Eq. (10)	1,000,000
{4, 1, 6}	462	3139.01	0.62	5572.92	0.03	3139.01	3.47	77.54	0.00	0.6	0.8	Eq. (10)	2000
{4, 2, 6}	3612	1878.18	6.53	3672.58	0.04	2418.96	23.65	95.54	28.79	0.6	0.8	Eq. (10)	10,000
{4, 2, 6}	3612	1878.18	6.53	3672.58	0.04	2163.33	21.86	95.54	15.18	0.6	0.8	Eq. (10)	11,000
{4, 2, 6}	3612	1878.18	6.53	3672.58	0.04	2163.33	31.07	95.54	15.18	0.6	0.8	Eq. (10)	15,000
{4, 2, 6}	3612	1878.18	6.53	3672.58	0.04	2163.33	40.04	95.54	15.18	0.6	0.8	Eq. (10)	20,000
{4, 2, 6}	3612	1878.18	6.53	3672.58	0.04	2163.33	1830.76	95.54	15.18	0.6	0.8	Eq. (10)	1,000,000
{4, 3, 6}	13,972	1656.36	126.96	2861.80	0.03	2246.71	158.19	72.78	35.64	0.6	0.8	Eq. (10)	30,000
{4, 3, 6}	13,972	1656.36	126.96	2861.80	0.03	2177.53	182.24	72.78	31.46	0.6	0.8	Eq. (10)	40,000
{4, 3, 6}	13,972	1656.36	126.96	2861.80	0.03	2177.53	186.05	72.78	31.46	0.6	0.8	Eq. (10)	50,000
{4, 3, 6}	13,972	1656.36	126.96	2861.80	0.03	2177.53	255.86	72.78	31.46	0.6	0.8	Eq. (10)	100,000
{4, 3, 6}	13,972	1656.36	126.96	2861.80	0.03	2177.53	1709.13	72.78	31.46	0.6	0.8	Eq. (10)	1,000,000
{5, 1, 6}	924	4620.64	1.23	7695.06	0.02	4620.64	5.18	66.54	0.00	0.6	0.8	Eq. (10)	3000
{5, 2, 6}	11,088	2729.80	38.95	5345.40	0.03	3524.43	82.49	95.82	29.11	0.6	0.8	Eq. (10)	30,000
{5, 2, 6}	11,088	2729.80	38.95	5345.40	0.03	3256.05	121.27	95.82	19.28	0.6	0.8	Eq. (10)	50,000
{5, 2, 6}	11,088	2729.80	38.95	5345.40	0.03	3256.05	175.68	95.82	19.28	0.6	0.8	Eq. (10)	80,000
{5, 2, 6}	11,088	2729.80	38.95	5345.40	0.03	3256.05	193.45	95.82	19.28	0.6	0.8	Eq. (10)	100,000
{5, 2, 6}	11,088	2729.80	38.95	5345.40	0.03	3256.05	1862.79	95.82	19.28	0.6	0.8	Eq. (10)	1,000,000
{5, 3, 6}	57,862	2288.32	13011.05	4168.91	0.03	4325.86	6672.90	82.18	89.04	0.6	0.8	Eq. (10)	150,000
{5, 3, 6}	57,862	2288.32	13011.05	4168.91	0.03	4318.15	7024.00	82.18	88.70	0.6	0.8	Eq. (10)	200,000
{5, 3, 6}	57,862	2288.32	13011.05	4168.91	0.03	2884.90	7218.10	82.18	26.07	0.6	0.8	Eq. (10)	300,000
{5, 3, 6}	57,862	2288.32	13011.05	4168.91	0.03	2884.90	7157.26	82.18	26.07	0.6	0.8	Eq. (10)	400,000
{5, 3, 6}	57,862	2288.32	13011.05	4168.91	0.03	2884.90	8713.24	82.18	26.07	0.6	0.8	Eq. (10)	1,000,000

[a] Ins.: Instances $\{|N|, C, T\}$, with $|N|$ affected areas, C capacities, and T planning periods.

[b] the gaps in the objective values between GE and DP, for example, $\frac{Obj(GE) - Obj(DP)}{Obj(DP)} \times 100\%$.

6. Conclusion and future work

In this paper, a model of resource allocation in humanitarian logistics is constructed with the objective of minimizing the accessibility-based delivery cost, the starting state-based deprivation cost, and the terminal penalty cost. A novel method based on the Q-learning algorithm, a reinforcement learning method, is developed to ensure the efficiency, effectiveness and equity of the allocation operations. A detailed description of the basic principles and fundamental components of the Q-learning algorithm is given, including the environment and its states, action space, and reward functions. The parameter settings of the proposed algorithm are also discussed in the experimental section. Numerical experiments are proposed to demonstrate the validity and efficiency of the proposed algorithm by comparing the results with an exact method, i.e., the dynamic programming algorithm, and a greedy heuristic algorithm. The results show that the Q-learning algorithm outperforms the dynamic programming algorithm in terms of computational time and far outperforms the greedy algorithm in terms of objective values. The proposed algorithm can obtain the optimal solution in polynomial time when the number of training episodes K is sufficiently large in the case of $C = 1$. The QL algorithm also converges to an acceptable solution in polynomial time in the case of $C > 1$. Therefore, the proposed Q-learning algorithm is well suited for the multiperiod, multiobjective and nonlinear resource allocation problem. In particular, in practical applications, the Q-learning algorithm can be optimized by adjusting the K value and training the algorithm in advance.

There are several limitations to our approach that should be addressed in future work. First, it would be more realistic to consider uncertain demand in practical applications. This paper focuses on the validity of the Q-learning algorithm for the resource allocation problem. For simplicity, this paper assumes that the demands of the affected areas are known. Future work should expand to the scenario of random demand, and it is expected that Q-learning will have advantages in stochastic problems. Second, the Q-learning algorithm used in this paper is a classic algorithm. This paper is only the first step in applying Q-learning to disaster relief resource allocation. In future research, QL algorithms with specific strategies that allow the QL to jump out of local solutions when $C > 1$ should be investigated. Moreover, other reinforcement learning algorithms can be introduced to the resource allocation problem. Third, the proposed algorithm can be compared with approximate dynamic programming in large-scale instances. In the field of operations research and control theory, the study of reinforcement learning is also referred to as approximate dynamic programming (Bertsekas, 2012). Therefore, it would be interesting to compare the Q-learning algorithm with other approximate dynamic programming algorithms, such as the rollout algorithm.

Funding

This work is supported by the National Natural Science Foundation of China under Grant Nos. 71901154, 71872092 and 71971144, and Beijing Municipal Natural Science Foundation (8192006).

CRediT authorship contribution statement

Lina Yu: Conceptualization, Methodology, Data curation, Writing - original draft. **Canrong Zhang:** Conceptualization. **Jingyan Jiang:** Methodology. **Huasheng Yang:** Conceptualization, Project administration, Supervision, Writing - review & editing. **Huayan Shang:** Data curation.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The authors would like to thank the anonymous referees for their valuable comments and suggestions which improve the quality of this paper.

References

Aihara, N., Adachi, K., Takyu, O., Ohta, M., & Fujii, T. (2019). Q-learning aided resource allocation and environment recognition in LoRaWAN with CSMA/CA. *IEEE Access, 7*, 152126–152137.

Bertsekas, D. P. (2012). *Dynamic programming and optimal control: Approximate dynamic programming* (4th Ed.). Nashua, NH: Athena Scientific.

Cao, C., Li, C., Yang, Q., Liu, Y., & Qu, T. (2018). A novel multi-objective programming model of relief distribution for sustainable disaster supply chain in large-scale natural disasters. *Journal of Cleaner Production, 174*, 1422–1435.

Cook, R. A., & Lodree, E. J. (2017). Dispatching policies for last-mile distribution with stochastic supply and demand. *Transportation Research Part E: Logistics and Transportation Review, 106*, 353–371.

Das, R., & Hanaoka, S. (2014). An agent-based model for resource allocation during relief distribution. *Journal of Humanitarian Logistics and Supply Chain Management, 4*(2), 265–285.

Du, B., Wu, C. & Huang, Z. (2019). Learning resource allocation and pricing for cloud profit maximization. In Proceedings of the AAAI conference on artificial intelligence (Vol. 33, pp. 7570–7577).

Fiedrich, F., Gehbauer, F., & Rickers, U. (2000). Optimized resource allocation for emergency response after earthquake disasters. *Safety Science, 35*(1), 41–57.

Gong, Q., & Batta, R. (2007). Allocation and reallocation of ambulances to casualty clusters in a disaster relief operation. *IIE Transactions, 39*(1), 27–39.

Gosavi, A. (2009). Reinforcement learning: A tutorial survey and recent advances. *INFORMS Journal on Computing, 21*(2), 178–192.

Harmon, M. E. & Harmon, S. S. (1997). Reinforcement learning: A tutorial (No. WL-TR-97-1028). WRIGHT LAB WRIGHT-PATTERSON AFB OH.

Holguín-Veras, J., Jaller, M., Van Wassenhove, L. N., Prez, N., & Wachtendorf, T. (2012). On the unique features of post-disaster humanitarian logistics. *Journal of Operations Management, 30*(7), 494–506.

Holguín-Veras, J., Prez, N., Jaller, M., Van Wassenhove, L. N., & Aros-Vera, F. (2013). On the appropriate objective function for post-disaster humanitarian logistics models. *Journal of Operations Management, 31*(5), 262–280.

Hu, C. L., Liu, X., & Hua, Y. K. (2016). A bi-objective robust model for emergency resource allocation under uncertainty. *International Journal of Production Research, 54*(24), 7421–7438.

Huang, K., Jiang, Y., Yuan, Y., & Zhao, L. (2015). Modeling multiple humanitarian objectives in emergency response to large-scale disasters. *Transportation Research Part E: Logistics and Transportation Review, 75*, 1–17.

Jacobson, E. U., Argon, N. T., & Ziya, S. (2012). Priority assignment in emergency response. *Operations Research, 60*(4), 813–832.

Jiang, C., & Sheng, Z. (2009). Case-based reinforcement learning for dynamic inventory control in a multi-agent supply-chain system. *Expert Systems with Applications, 36*(3), 6520–6526.

Jiang, Z., Fan, W., Liu, W., Zhu, B., & Gu, J. (2018). Reinforcement learning approach for coordinated passenger inflow control of urban rail transit in peak hours. *Transportation Research Part C: Emerging Technologies, 88*, 1–16.

Jiang, Z., Gu, J., Fan, W., Liu, W., & Zhu, B. (2019). Q-learning approach to coordinated optimization of passenger inflow control with train skip-stopping on a urban rail transit line. *Computers & Industrial Engineering, 127*, 1131–1142.

Kara, A., & Dogan, I. (2018). Reinforcement learning approaches for specifying ordering policies of perishable inventory systems. *Expert Systems with Applications, 91*, 150–158.

Khadilkar, H. (2018). A scalable reinforcement learning algorithm for scheduling railway lines. *IEEE Transactions on Intelligent Transportation Systems, 20*(2), 727–736.

Kim, Y., Kim, S., & Lim, H. (2019). Reinforcement learning based resource management for network slicing. *Applied Sciences, 9*(11), 2361.

Klaine, P. V., Nadas, J. P., Souza, R. D., & Imran, M. A. (2018). Distributed drone base station positioning for emergency cellular networks using reinforcement learning. *Cognitive Computation, 10*(5), 790–804.

Kwon, I. H., Kim, C. O., Jun, J., & Lee, J. H. (2008). Case-based myopic reinforcement learning for satisfying target service level in supply chain. *Expert Systems with Applications, 35*(1–2), 389–397.

Loree, N., & Aros-Vera, F. (2018). Points of distribution location and inventory management model for Post-Disaster Humanitarian Logistics. *Transportation Research Part E: Logistics and Transportation Review, 116*, 1–24.

Nadi, A., & Edrisi, A. (2017). Adaptive multi-agent relief assessment and emergency response. *International Journal of Disaster Risk Reduction, 24*, 12–23.

Nadi, A., & Edrissi, A. (2016). A reinforcement learning approach for evaluation of real-time disaster relief demand and network condition. *World Academy of Science, Engineering and Technology, International Journal of Environmental, Chemical, Ecological, Geological and Geophysical Engineering, 11*(1), 5–10.

Ni, J., Liu, M., Ren, L., & Yang, S. X. (2013). A multiagent Q-learning-based optimal allocation approach for urban water resource management system. *IEEE Trans. Autom. Sci. Eng., 11*(1), 204–214.

Pérez-Rodríguez, N., & Holguín-Veras, J. (2015). Inventory-allocation distribution models for postdisaster humanitarian logistics with explicit consideration of deprivation costs. *Transp. Sci., 50*(4), 1261–1285.

Pradhananga, R., Mutlu, F., Pokharel, S., Holguín-Veras, J., & Seth, D. (2016). An integrated resource allocation and distribution model for pre-disaster planning. *Computers & Industrial Engineering, 91*, 229–238.

Russell, S. J., & Norvig, P. (2003). *Artificial intelligence, a modern approach* (3rd Ed.). Pearson Education.

Santos, E. C. (2017). A simple reinforcement learning mechanism for resource allocation in lte-a networks with markov decision process and q-learning. arXiv preprint arXiv: 1709.09312.

Šemrov, D., Marsetič, R., Žura, M., Todorovski, L., & Srdic, A. (2016). Reinforcement learning approach for train rescheduling on a single-track railway. *Transportation Research Part B: Methodological, 86*, 250–267.

Shavarani, S. M. (2019). Multi-level facility location-allocation problem for post-disaster humanitarian relief distribution: A case study. *Journal of Humanitarian Logistics and Supply Chain Management, 9*(1), 70–81.

Sheu, J. B., Chen, Y. H. and Lan, L. W. (2005). A novel model for quick response to disaster relief distribution. In Proceedings of the Eastern Asia Society for transportation studies (Vol. 5, No. 1, pp. 2454–2462).

Sheu, J. B. (2007). An emergency logistics distribution approach for quick response to urgent relief demand in disasters. *Transportation Research Part E: Logistics and Transportation Review, 43*(6), 687–709.

Sheu, J. B. (2014). Post-disaster relief-service centralized logistics distribution with survivor resilience maximization. *Transportation Research Part B: Methodological, 68*, 288–314.

Sutton, R. S., & Barto, A. G. (1998). *Introduction to reinforcement learning* (Vol. 135). Cambridge: MIT press.

Su, Z. P., Jiang, J. G., Liang, C. Y., & Zhang, G. F. (2011). Path selection in disaster response management based on Q-learning. *International Journal of Automation and Computing, 8*(1), 100–106.

Teo, J. S., Taniguchi, E., & Qureshi, A. G. (2012). Evaluation of distance-based and cordon-based urban freight road pricing in e-commerce environment with multiagent model. *Transportation Research Record, 2269*(1), 127–134.

Vengerov, D. (2007). A reinforcement learning approach to dynamic resource allocation. *Engineering Applications of Artificial Intelligence, 20*(3), 383–390.

Watkins, C. J., & Dayan, P. (1992). Q-learning. *Machine Learning, 8*(3–4), 279–292.

Wex, F., Schryen, G., Feuerriegel, S., & Neumann, D. (2014). Emergency response in natural disaster management: Allocation and scheduling of rescue units. *European Journal of Operational Research, 235*(3), 697–708.

Xiang, Y., & Zhuang, J. (2016). A medical resource allocation model for serving emergency victims with deteriorating health conditions. *Annals of Operations Research, 236*(1), 177–196.

Yang, T., Hu, Y., Gursoy, M. C., Schmeink, A. & Mathar, R. (2018). Deep reinforcement learning based resource allocation in low latency edge computing networks. In 2018 15th international symposium on wireless communication systems (ISWCS) (pp. 1–5). IEEE.

Ye, H., Li, G. Y., & Juang, B. H. F. (2019). Deep reinforcement learning based resource allocation for V2V communications. *IEEE Transactions on Vehicular Technology, 68*(4), 3163–3173.

Yu, L., Zhang, C., Yang, H., & Miao, L. (2018). Novel methods for resource allocation in humanitarian logistics considering human suffering. *Computers & Industrial Engineering, 119*, 1–20.

Yu, L., Yang, H., Miao, L., & Zhang, C. (2019). Rollout algorithms for resource allocation in humanitarian logistics. *IISE Transactions, 51*(8), 887–909.

Zhang, J. H., Li, J., & Liu, Z. P. (2012). Multiple-resource and multiple-depot emergency response problem considering secondary disasters. *Expert Systems with Applications, 39*(12), 11066–11071.

Zhou, Y., Liu, J., Zhang, Y., & Gan, X. (2017). A multi-objective evolutionary algorithm for multi-period dynamic emergency resource scheduling problems. *Transportation Research Part E: Logistics and Transportation Review, 99*, 77–95.