doi:10.3772/j.issn.1006-6748.2021.01.003

A geospatial service composition approach based on MCTS with temporal-difference learning①

Zhuang Can (庄 灿)*, Guo Mingqiang*, Xie Zhong②*,**
(*School of Geography and Information Engineering, China University of Geosciences, Wuhan 430074, P. R. China)
(**National Engineering Research Center for GIS, Wuhan 430074, P. R. China)

Abstract

With the complexity of the composition process and the rapid growth of candidate services, realizing optimal or near-optimal service composition is an urgent problem. Currently, the static service composition chain is rigid and cannot be easily adapted to the dynamic Web environment. To address these challenges, the geographic information service composition (GISC) problem as a sequential decision-making task is modeled. In addition, the Markov decision process (MDP), as a universal model for the planning problem of agents, is used to describe the GISC problem. Then, to achieve self-adaptivity and optimization in a dynamic environment, a novel approach that integrates Monte Carlo tree search (MCTS) and a temporal-difference (TD) learning algorithm is proposed. The concrete services of abstract services are determined with optimal policies and adaptive capability at runtime, based on the environment and the status of component services. The simulation experiment is performed to demonstrate the effectiveness and efficiency through learning quality and performance.

Key words: geospatial service composition, reinforcement learning (RL), Markov decision process (MDP), Monte Carlo tree search (MCTS), temporal-difference (TD) learning

0 Introduction

With the rapid development of cloud computing and Web technologies, the Internet has become an open distributed computing platform. Especially in the era of cloud computing, numerous enterprises and institutions publish their products in the form of Web services on the Internet for people to access, and applications are no longer built from scratch but from a composition of available services[1]. In the geospatial application domain, more and more geographic information (GI) services are being used by the public. Because of the complexity of geospatial problems, a single GI service can not effectively meet user needs and its capability is limited[2]. Geographic information service composition (GISC) has become a widely used pattern to reuse resources and build complex applications because of its interoperability, reusability, and flexibility[3-5]. With the complexity of the composition process and the rapid growth of candidate services, achieving optimal or near-optimal service composition is a non-trivial task.

GISC has received considerable attention in recent years. Most works can be classified into two categories: static methods based on workflow techniques and adaptive and automatic methods based on artificial intelligence (AI) planning techniques.

For static methods, a complete service chain is constructed before the execution of service composition[6-8]. The construction of service composition starts with a process model formed by abstract services in workflow-based techniques. The abstract workflow is instantiated by a set of concrete services with the optimal quality of service (QoS). A number of investigations have been devoted to building geoprocessing workflows. Kepler[9] has been used to implement distributed geospatial data processing workflows. GeoPWTManager[8] leverages Web service and workflow technologies to design and execute tasks. GeoJModelBuilder[10] is an open-source geoprocessing workflow tool. Other works mostly used BPEL-based business workflow technology to orchestrate geospatial services[11]. However, these only simplify the workflow

① Supported by the National Natural Science Foundation of China (No. 41971356, 41671400, 41701446), National Key Research and Development Program of China (No. 2017YFB0503600, 2018YFB0505500), and Hubei Province Natural Science Foundation of China (No. 2017CFB277).
② To whom correspondence should be addressed. E-mail: xiezhong@cug.edu.cn
Received on Mar. 24, 2020

composition process syntactically and cannot perform automatic service composition.

Recently, more attempts are undertaken towards achieving automatic geospatial service composition. There have been many approaches to perform automatic service composition, including rule-based[4], semantic-based[12-13], and integer programming[14-15] approaches. Moreover, some approximate optimization algorithms, such as genetic algorithms[16], particle swarm optimization[17], and evolutionary algorithms[18], have been adopted to achieve optimization of service composition[19]. For example, Ref. [20] transformed the optimization of service composition into a fuzzy linear programming problem by considering users' preferences, to better find the optimal composition to meet users' needs. In general, in most of the approaches, QoS values are assumed to be fixed and determined. Consequently, its adaptivity to the dynamic environments is poor.

However, the Web environment is highly dynamic, and a static service chain is rigid and not self-adaptive. When one of the services is unavailable, the composition has to be adjusted manually[1]. In recent years, adaptive service composition has attracted wide attention, and there have been many approaches to address the adaptability issue, most of which perform service composition optimization at runtime under a dynamic environment, such as graph-based approaches[21], reinforcement learning (RL) algorithms[22]. Ref. [23] proposed an algorithm for service composition using multi-objective reinforcement learning. Ref. [22] employed the Q-learning algorithm to solve constraint-satisfied service composition modeled as a Markov decision process (MDP). Ref. [24] combined the Gaussian process with an RL algorithm for adaptive service composition. In addition, along with the rapid increase of available Web services, some studies have focused on solving the problem of large-scale service composition[24-25]. Because of the complexity and high dynamicity of GISC, there remains a huge need for further research exploration.

In this study, a model based on MDP, called GISC-MDP, is adopted to address the GISC problem in uncertain and dynamic environments and model it as a sequential decision-making task. To achieve good adaptation and scalability in a highly dynamic environment, a novel approach that is built upon the GISC-MDP model based on Monte Carlo tree search (MCTS)[26] is proposed. MCTS is a tree search algorithm used to find optimal decision-making through a random simulation and is thus very suitable for the GISC problem. Furthermore, the MCTS with an on-policy temporal-difference (TD) learning algorithm[27], namely state-action-reward-state-action (SARSA(λ)), is enhanced to make it capable of adapting to the dynamic environment and obtaining the optimal policy. The contributions of this study are summarized as follows.

(1) A new model is introduced to handle the GISC problem based on the MDP in a dynamic and complex environment.

(2) A novel approach is devised to achieve better adaptation and scalability, using the MCTS algorithm.

(3) An efficient approach is proposed to obtain the optimal policy by integrating MCTS and an on-policy TD learning algorithm with a softmax exploration strategy.

The remainder of this paper is organized as follows. Section 1 introduces related research technologies. Section 2 presents a model for GISC. Section 3 proposes a composition algorithm based on MCTS-TD. Section 4 shows the experimental results. A discussion is given in Section 5. Section 6 concludes the paper and discusses future work.

1 Background

1.1 RL

RL is an important branch of machine learning and artificial intelligence areas. In an RL algorithm, the agent is asked to interact with a dynamic environment and learn how to action optimally through constant trial and error[28]. RL has a self-learning ability, the goal of which is to obtain the optimal policy to maximize the total reward value. The fundamental elements of an RL algorithm are the agent, the environment, the state, the action, and the reward (Fig. 1). The interaction between the agent and the environment goes something like this: the environment will change to a new state s_{t+1} when the agent chooses an action a_t and simultaneously feeds back an immediate reward signal r_{t+1} (positive or negative) to the agent based on the

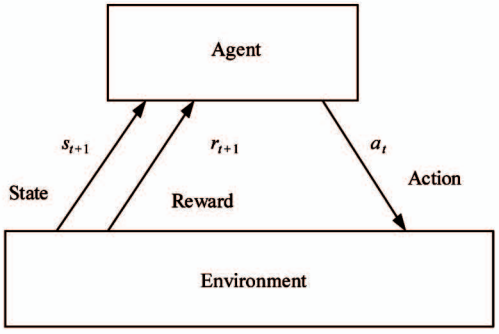

Fig. 1 Reinforcement learning model

taken action a_t. According to the current state and reward signals, the agent then selects the next action under a certain policy, the principle of which is to maximize long-term cumulative rewards. Finally, an RL algorithm obtains an optimal policy for the decision-making sequence through a continuous learning process.

1.2 TD learning

TD learning is the core idea of reinforcement learning, which is a combination of Monte Carlo (MC) and dynamic programming (DP) methods[29]. Like the MC method, TD algorithms learn from historical experience without a model of the domain and update the value function of the current state with the value function of the successive state, similar to DP.

The TD algorithm is divided into two parts: TD prediction and TD control. The former is used to calculate the state value, and the latter is used to get the optimal policy. For TD prediction, the TD method is faster than the MC method in terms of computing time. The formula for updating the state S_t at time t is

$$V(S_t) \leftarrow V(S_t) + \alpha[R_{t+1} + \gamma V(S_{t+1}) - V(S_t)] \tag{1}$$

For TD control, there are usually two methods: on-policy and off-policy methods. In an on-policy algorithm, such as SARSA, the strategy for choosing an action is the same as that for updating the action-value function. For an off-policy algorithm, the opposite is true.

1.3 MCTS

MCTS is a tree search algorithm that combines the generality of random simulation with the accuracy of the tree search and does not need prior knowledge in a specific domain[30]. It uses random simulation to estimate the value of an action and builds a search tree to find the optimal policy. In theory, MCTS can be used for any field that can be described in the form of {state, action} and predicted by simulation.

Initially, MCTS is an empty tree. Afterward, it iteratively repeats four phases until the terminate condition is met; finally, an asymmetric search tree is built. The phases of a typical MCTS algorithm shown in Fig. 2 are as follows[31].

Selection From the root node s_0, a child node is selected recursively with a tree policy. If the current node is a non-terminal state and an incompletely expanded node, then the node is expandable.

Expansion An unexplored child node is added to the current node by taking a valid action.

Simulation According to the default policy, a simulation is performed from the new node to the terminal node s_t or a predetermined depth in the tree and the simulation result is obtained.

Backpropagation The simulation result of the new node is backpropagated to all selected nodes on the current iteration path, and the number of visits is updated.

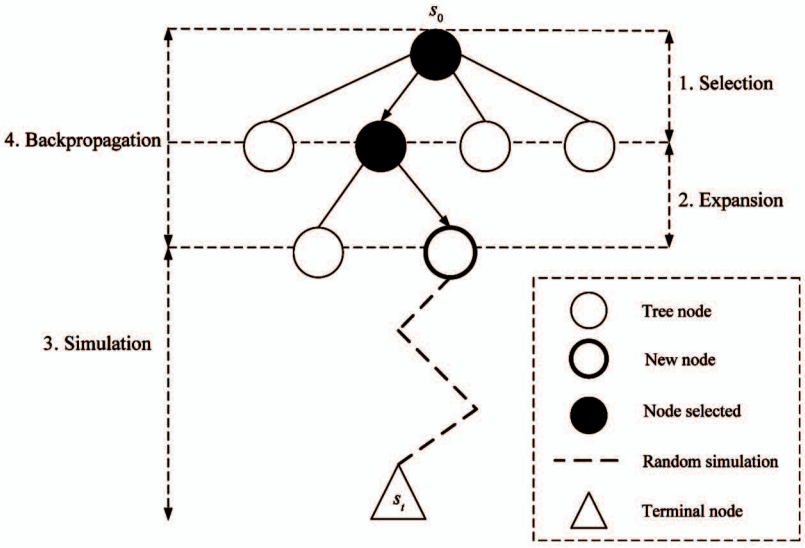

Fig. 2 MCTS iteration process

At present, the most popular of these algorithms is the upper confidence bound for trees (UCT) algorithm. The UCT algorithm combines the upper confidence bound (UCB) with Monte Carlo simulation to expand the search state. The UCB is a policy of the multi-armed bandit problem; it provides a trade-off between exploration and exploitation with the aim of optimizing the expected total reward. The UCB can be expressed as follows:

$$UCB(a_i) = \overline{X(a_i)} + C\sqrt{\frac{2\ln n(a)}{n(a_i)}} \quad (2)$$

where, $\overline{X(a_i)}$ is the average value of the action a_i and is normalized to the interval $[0,1]$, and $n(a_i)$ and $n(a)$ are respectively the number of visits to the action a_i and the total number of selections that perform and lead to the current action a_i, and C is a parameter for better tuning of the trade-off between exploration and exploitation.

2 An MDP model for geographic information service composition

This section starts with the problem description of GISC and then some basic concepts relevant to GISC are defined. Finally, to formalize the approach, the concept of the MDP is employed to schematically model the GISC process.

2.1 Problem description

The aim of GISC is to automatically integrate and combine small-granularity GI services as a complex composite service that can fulfill a user's functional requirements with an optimal policy. In this study, the functional requirements are achieved by a predefined task model that is the abstract representation of the composite service. In the current service environment, there exist many services with similar functionality, but with different quality. These services have been grouped as candidate service sets using clustering algorithms. The work entails selecting the appropriate service for each subtask in the task process model at runtime, to generate a concrete composite service chain with the best quality.

For instance, the task process model is composed of n abstract services, as shown in Fig. 3. The abstract service is defined as a geooperator (which will be described in the next subsection). Services with the same functional properties are clustered into a candidate service set, such as $A(s_1)$, $A(s_2)$, and $A(s_n)$. Then, in turn, an appropriate service gs_{nk} for each abstract service $Goper_n$ from candidate service set $A(s_n)$ is selected to composite a concrete service chain that maximizes user satisfaction.

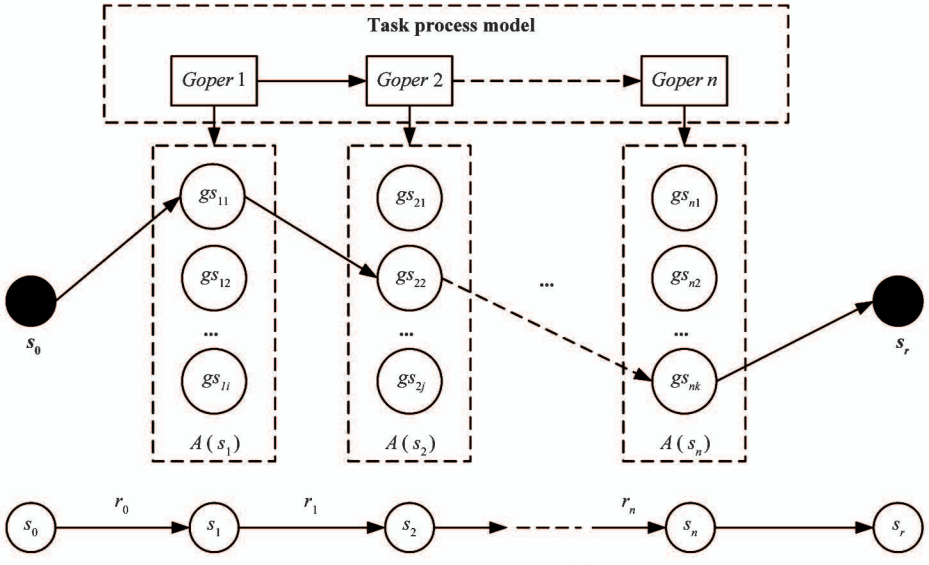

Fig. 3 GISC-MDP model

2.2 Problem formalization

This subsection gives the formal description of the main concepts and definitions relevant to the GISC problem.

Definition 1 (GI service) A GI service is a functionally complete self-governed resource following open and standard service specifications that can be published and accessed on the Web. A GI service can be defined as a triple:

$$GIService = (ID, Geooperator, QoGIS) \quad (3)$$

where *ID* is the identifier of a service. *Geooperator* is the abstract class of service, and *QoGIS* is the quality attribute values of the GI service. *QoGIS* can be represented by an n-tuple $(q_1, q_2, \cdots, q_n)$, where each q_i is a *QoGIS* attribute value, and n is the number of attributes of interest.

Definition 2 (*Geooperator*) The concept of the *Geooperator* was proposed in Ref. [32]. It provides an abstract representation and formalization for GI services with similar functionality but different non-functional properties and also provides building blocks for a task process model. For simplicity, in this study, the fol-

lowing definition is used:

$$Geooperator = (Input, Output, Desc) \quad (4)$$

where *Input* is a set of spatial datasets or non-spatial parameters of the service, *Output* represents the output items of the service and *Desc* is the functional description.

Definition 3 (Task process model) A task process model defines the basic structure of a sequential execution workflow, which is an acyclic directed graph that links multiple geooperators. It can be denoted as follows:

$$TPM = (GV, GE) \quad (5)$$

where GV is a finite set of n vertices $\{gv_1, gv_2, \cdots, gv_n\}$, with each node $gv_i (i = 1, 2, \cdots, n) \in GV$ representing the ith geooperator, n is the total number of geooperators, and GE is a finite set of directed edges $\{ge_1, ge_2, \cdots, ge_n\}$. Each edge $\{ge_i\} \in GE$ can be characterized by a tuple $(pair_{ab}, conn_{ab})$. $pair_{ab} = < gv_a, gv_b >$ is an ordered pair that represents execution precedence between gv_a and gv_b; in other words, gv_a is ahead of gv_b in the sequence of the workflow. $conn_{ab}$ represents the control flow connector between two geooperators, which includes sequence, branching, loop, and so forth.

Definition 4 (Candidate service set) The candidate service set is a finite set of alternative services that have the same *Goper* but differ in *QoGIS*. A candidate service set can be defined as a set $\{gs_1, gs_2, \cdots, ge_m\}$, where m is the number of services in the set.

2.3 GISC-MDP model

GISC is the process of selecting the appropriate service for each *Geooperator* in the task process model to generate a concrete composite service chain, which is actually a sequential decision-making process. The MDP, as a universal model for the planning problem of agents, can effectively describe the sequential decision-making problem in a dynamic environment; thus it is especially desirable for the GISC problem.

Definition 5 (GISC-MDP) A GISC model based on the MDP can be formulated as a 5-tuple:

$$GISC\text{-}MDP = < S, s_0, s_r, A(\cdot), R > \quad (6)$$

where S is a finite set called the state space, the elements of $s_0 \in S$ and $s_r \subset S$ are the initial state and the possible terminal state set of the GISC-MDP transition graph, respectively, $A(\cdot) = A(s_1) \times A(s_2) \times \cdots \times A(s_n)$ is a joint set of the action space in all states $s_i (i = 1, 2, \cdots, n)$, where $A(s_i)$ represents an action set in the state $s_i \in S$ and $R(s, a)$ is a reward function that specifies an expected immediate reward when $a \in A(s)$ is invoked in the current state s.

In the GISC scenario, as shown in Fig. 3, the set of abstract service class *Geooperator* is taken as the state set S, optional GI services in the candidate set are as the action set, and the QoGIS value as a reward value R that selects the corresponding action. The solution to GISC-MDP is a set of policies, each of which is a process of service selection.

3 Composition algorithm based on MCTS-TD

3.1 Softmax action selection

Tree or action selection policy plays a significant role in the process of building the search tree. In the MCTS algorithm, the node with the highest reward estimate may not be optimal because of the randomness of the simulation. Consequently, action selection must balance exploration versus exploitation of the search tree. Exploration entails discovering the nodes with a low estimate and number of visits but with a possible higher win rate. Exploitation entails selecting the node from experience that gives the highest expected reward.

For the selection step of the MCTS algorithm, the UCB requires that values are distributed in the interval $[0,1]$, but a win rate as a reward estimate cannot be gotten. Inspired by Ref. [33], the softmax exploration approach was adopted as an action selection policy. It is a more complex approach because it takes into account the relative values of all nodes. The approach has been proved to provide a more effective trade-off between exploration and exploitation. The probability of selection of each action is calculated as

$$P(a) = \frac{e^{Q(s,a)/\tau}}{\sum_{b=1}^{n} e^{Q(s,a)/\tau}} \quad (7)$$

Similar to ε-greedy exploration, the extent of exploration is controlled by a single parameter τ. When τ takes a higher value, the probabilities of all actions that can be selected are almost equal, no matter what estimated value is. The parameter τ usually decreases during the learning process, thus enabling the agent to focus on actions with higher values. When τ gets close to zero, a greedy policy is approached.

3.2 Rewards

The ultimate objective of solving GISC-MDP is to get an optimal strategy that can provide a service sequence to maximize users' satisfaction. The reward value $R(s,a)$ of GISC-MDP should be a certain measurement of user satisfaction. In this study, normalized QoGIS values of GI service are employed as reward values.

Because of the inconsistency of quality factor values in their range and units, QoGIS factor values need to be normalized. The quality factors have two characteristics: positive and negative. For positive factors, the higher value represents better quality, i. e. , reliability, availability, and completeness, etc. Negative values have opposite characteristics. The normalized formulas for positive and negative quality factors are

$$q'_{pasitive} = \begin{cases} \dfrac{q - q_{\min}}{q_{\max} - q_{\min}} & \text{if } q_{\max} \neq q_{\min} \\ 1 & \text{if } q_{\max} = q_{\min} \end{cases} \quad (8)$$

$$q'_{negative} = \begin{cases} \dfrac{q_{\max} - q}{q_{\max} - q_{\min}} & \text{if } q_{\max} \neq q_{\min} \\ 1 & \text{if } q_{\max} = q_{\min} \end{cases} \quad (9)$$

Then, the reward formula in the model can use a linear weighted integration function that can be expressed by the following equation:

$$R(s, a) = \sum_{i=1}^{n} (q'_i \cdot w_i) \quad (10)$$

where n is the number of *QoGIS* factors, w_i and q'_i are the weight and the normalized value of the i-th *QoGIS* factors, respectively.

3.3 SARSA (λ)

The primitive MCTS algorithm maintains a total reward for each node, which is updated in the backpropagation step. However, SARSA(λ) is introduced into the MCTS algorithm. SARSA(λ) is an on-policy TD algorithm that combines eligibility traces with the SARSA algorithm. First, an action in the current state using the softmax selection policy is selected and the instant reward on each step is calculated (Eq. (10)). Then, the nodes with the rewards collected from the current node and all its descendants in the backpropagation step are updated. The trace, denoted as $E(s, a)$, is:

$$E_t(s, a) = \lambda\gamma E_{t-1}(s, a) \quad (11)$$

where γ is a discount factor, and λ is called the trace-decay parameter. In some sense, $\lambda\gamma$ shows how far the state s is from the current node. The TD error for each node is calculated as

$$\delta_t = R_t + \gamma Q_t(S_{t+1}, A_{t+1}) - Q_t(S_t, A_t) \quad (12)$$

Based on this, all its ancestors with the TD error are updated by

$$Q_{t+1}(s, a) = Q_t(S_t, A_t) + \alpha\delta E_t(s, a) \quad (13)$$

4 Experiments and analysis

4.1 Test data set

In the approach used here, the *QoGIS* value of GI service is adopted as the immediate reward value. Owing to the lack of datasets for GI services, the test dataset used is the WS-DREAM[34] dataset. The dataset describes real-world QoS measurements, including response-time and throughput, obtained from 339 users on 5825 Web services. Table 1 lists some data segments of response-time in the test dataset. The WS-DREAM dataset is preprocessed by taking the average value of 339 users as the quantitative value of 5825 Web services and then normalizing the response-time and throughput factors.

Table 1 Data segments of response-time in the WS-DREAM data set

Webservice / User	WS_1	WS_2	WS_3	WS_4	…	WS_{5825}
$User_1$	5.982	0.228	0.237	0.221	…	6.777
$User_2$	2.130	0.262	0.273	0.251	…	0.263
$User_3$	0.854	0.366	0.376	0.357	…	-1
$User_4$	0.693	0.226	0.233	0.220	…	0.173
…	…	…	…	…	…	…
$User_{339}$	1.285	0.222	0.232	0.215	…	0.130

4.2 Experiment Setting

In the following experiment, the response-time and throughput attributes provided by the preprocessed WS-DREAM datasets are considered. The reward value is calculated using Eq. (10) and the weights of response-time and throughput attributes are 0.7 and 0.3, respectively. All the experiments are conducted on a PC with a 3.10 GHz Intel Core i5-4440 CPU with 16 GB of RAM. Unless mentioned otherwise, the experimental parameters are set as follows: the discount factor γ is 0.9, the trace-decay parameter γ is 0.2, and the value ε of greedy policy is 0.1.

To indicate the efficiency of the proposed model, some comparative experiments were performed. Three other scenarios: (1) removing the softmax exploration strategy, only using the ε-greedy strategy (-softmax); (2) removing the TD algorithm (-TD); (3) Flat Monte Carlo, a classic algorithm, are evaluated.

4.3 Learning quality under a varying number of abstract services

The main objective of this experiment is to verify the ability of the MCTS-TD approach to obtain GISC with high quality. The total accumulated reward is used as a measure to compare with the other three approaches. In the first stage of the evaluation, two task process models, consisting of 20 abstract services and 50 abstract services, are considered in the experiment. The number of candidate services for each abstract service is set to 20, 50, and 100, respectively. The learning episode is set to 300. Fig. 4 and Fig. 5 depict the accumulated rewards obtained by these approaches with the different number of abstract services. As shown in the figures, when the proposed method removes the softmax exploration strategy and TD algorithm, the accumulated reward declines considerably. The accumulated reward of the proposed approach is found to be significantly superior to that of three comparative methods, independent of the number of abstract services and candidate services for each abstract service.

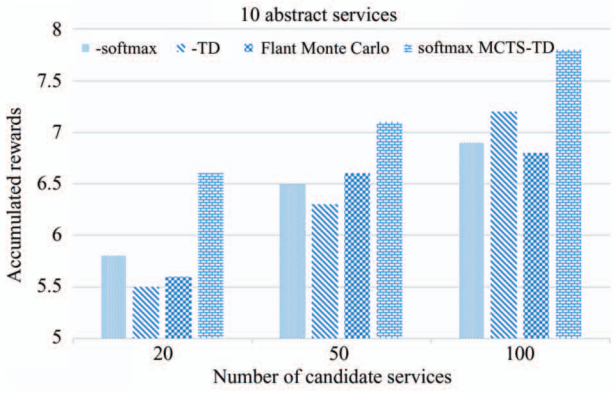

Fig. 4 Accumulated rewards for 10 abstract services

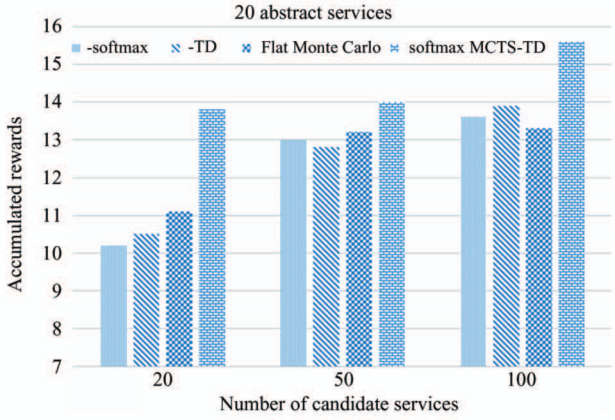

Fig. 5 Accumulated rewards for 20 abstract services

4.4 Average rewards under a varying number of candidate services

The main purpose of the second experiment is to examine the performance of this approach under different scales of service environments. Average reward is an important performance indicator for service composition methods. In the experiment, the service environment scale is represented by the number of candidate services assigned to each abstract service. For this, a task process model consisting of 10 abstract services is considered. The number of available candidate services for each task is then set to 100 and 200, respectively. The number of learning episodes is varied from 0 to 300 and compared the results with those of the other methods under different environmental scales. As can be seen from Fig. 6 and Fig. 7, the proposed MCTS-TD approach with a softmax policy slightly outperforms other methods, regardless of the number of available candidate services per task. The approach suggested here achieves higher average accumulated rewards through the learning process, but superiority is not obvious. The main reason for this may be due to having fewer learning episodes and candidate services in the experiment.

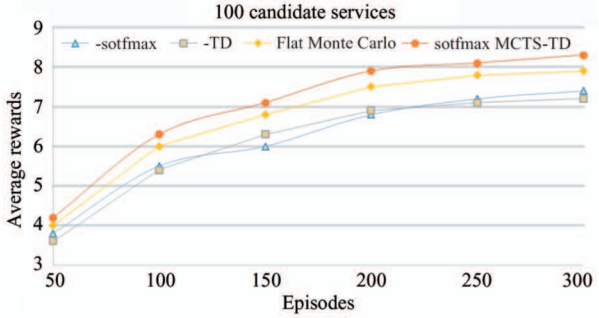

Fig. 6 Average rewards for 100 candidate services

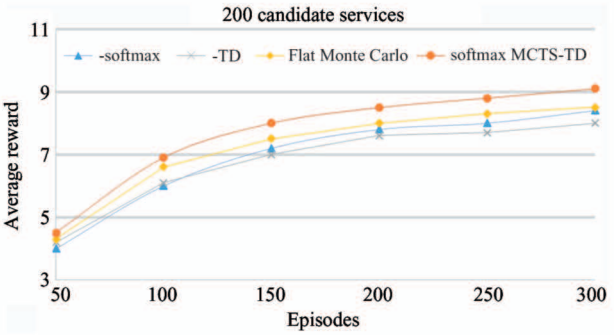

Fig. 7 Average rewards for 200 candidate services

5 Discussion

The GISC is a complex problem, and the proposed method solves it as a sequential decision-making

task based on the MDP. In contrast to the existing approaches, the mechanism used here has more advantages, as specified in the following.

The tolerance time for diverse users is inconsistent in the GISC problem. To help satisfy the flexibility requirements, the proposed method adopts the MCTS algorithm (see Section 3), which can terminate at any time and provide a decision-making scheme.

In addition, the proposed approach not only obtains the optimal service composition but also offers multiple alternatives by ranking with reward estimate values. The alternative services can be quickly put to use when an optimal service is not available in the GISC. The proposed approach has self-healing and self-adaptive capability.

The proposed approach employs the softmax exploration strategy instead of the ε-greedy strategy to obtain better optimization results. The comparative experiment verifies its efficiency.

6 Conclusion

In this study, the GISC problem based on the MDP, which is transcribed as a sequential decision-making task, is modeled. As opposed to most current approaches, which implement GISC upon a static service chain, the approach here attempts to develop GISC at run-time so that it can achieve self-adaption in a dynamic environment. To this end, a novel approach that combines MCTS with a TD learning algorithm is presented. The concrete services of abstract services are determined at runtime, based on the environment and the status of component services. The experimental simulation reveals that the proposed approach is effective and adaptive, and it is demonstrated that this approach can be effective, adaptive, and scalable in solving the GISC problem.

In the future, much more work needs to be done and directions developed to pursue optimation of GISC. For example, in noisy environments, a filtering mechanism to exclude services with extremely poor QoGIS will be devised, this can reduce the time spent in the exploration process. In addition, a more generalized decision model will be introduced to adapt to an environment that is only partially observed.

References

[1] Wang H, Zhou X, Zhou X, et al. Adaptive service composition based on reinforcement learning[C]// Proceedings of the International Conference on Service-Oriented Computing, Berlin, Germany, 2010: 92-107

[2] Zhuang C, Xie Z, Ma K, et al. A task-oriented knowledge base for geospatial problem-solving[J]. *ISPRS International Journal of Geo-Information*, 2018, 7(11): 423

[3] Wang H, Chen X, Wu Q, et al. A rule-based description framework for the composition of geographic information services integrating reinforcement learning with multi-agent techniques for adaptive service composition[J]. *ACM Transactions on Autonomous and Adaptive Systems*, 2017, 12(2): 1-42

[4] Lutz M, Lucchi R, Friis-Christensen A, et al. A rule-based description framework for the composition of geographic information services[C]// Proceedings of the International Conference on Geospatial Semantics, Geos, Mexico, 2007: 114-127

[5] Chenga D, Wangb F. An approach on dynamic geospaital information service composition based on context relationship[J]. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2011, 25(4): 33-37

[6] Sun Z, Yue P, Lu X, et al. A task ontology driven approach for live geoprocessing in a service-oriented environment[J]. *Transactions in GIS*, 2012, 16(6): 867-884

[7] Yue P, Zhang M, Tan Z. A geoprocessing workflow system for environmental monitoring and integrated modelling [J]. *Environmental Modelling and Software*, 2015, 69: 128-140

[8] Sun Z, Yue P, Di L. GeoPWTManager: a task-oriented web geoprocessing system[J]. *Computers and Geosciences*, 2012, 47(8): 34-45

[9] Ludäscher B, Altintas I, Berkley C, et al. Scientific workflow management and the Kepler system[J]. *Concurrency and Computation: Practice and Experience*, 2006, 18(10): 1039-1065

[10] Zhang M, Bu X, Peng Y. GeoJModelBuilder: an open source geoprocessing workflow tool[J]. *Open Geospatial Data Software and Standards*, 2017, 2(1): 1-8

[11] Hobona G, Fairbairn D, Hiden H, et al. Orchestration of grid-enabled geospatial web services in geoscientific workflows[J]. *IEEE Transactions on Automation Science and Engineering*, 2009, 7(2): 407-411

[12] Peng Y, Di L, Yang W, et al. Semantics-based automatic composition of geospatial Web service chains[J]. *Computers and Geosciences*, 2007, 33(5): 649-665

[13] Al-Areqi S, Lamprecht A L, Margaria T. Constraints-driven automatic geospatial service composition: workflows for the analysis of sea-level rise impacts[C]// Proceedings of the International Conference on Computational Science and Its Applications, Beijing, China, 2016: 134-150

[14] Cardellini V, Casalicchio E, Grassi V, et al. MOSES: a framework for QoS driven runtime adaptation of Service-Oriented systems[J]. *IEEE Transactions on Software Engineering*, 2012, 38(5): 1138-1159

[15] Ghobaei-Arani M, Souri A. LP-WSC: a linear programming approach for web service composition in geographically distributed cloud environments[J]. *The Journal of Supercomputing*, 2019, 75(5): 2603-2628

[16] Seghir F, Khababa A. A hybrid approach using genetic

and fruit fly optimization algorithms for QoS-aware cloud service composition[J]. *Journal of Intelligent Manufacturing*, 2018, 29(8): 1773-1792

[17] Chen F, Dou R, Li M, et al. A flexible QoS-aware Web service composition method by multi-objective optimization in cloud manufacturing[J]. *Computers and Industrial Engineering*, 2016, 99: 423-431

[18] Bouzary H, Frank Chen F. A hybrid grey wolf optimizer algorithm with evolutionary operators for optimal QoS-aware service composition and optimal selection in cloud manufacturing[J]. *The International Journal of Advanced Manufacturing Technology*, 2019, 101(9): 2771-2784

[19] Bouzary H, Frank Chen F. Service optimal selection and composition in cloud manufacturing: a comprehensive survey[J]. *The International Journal of Advanced Manufacturing Technology*, 2018, 97(1): 795-808

[20] Wang P, Chao K M, Lo C C. On optimal decision for QoS-aware composite service selection[J]. *Expert Systems with Applications*, 2010, 37(1): 440-449

[21] Chen M, Yan Y. Redundant service removal in QoS-aware service composition[C]// Proceedings of the 2012 IEEE 19th International Conference on Web Services, Honolulu, USA, 2012: 431-439

[22] Ren L, Wang W, Xu H. A reinforcement learning method for constraint-satisfied services composition[J]. *IEEE Transactions on Services Computing*, 2017: 1-1

[23] Mostafa A, Zhang M. Multi-objective service composition in uncertain environments[J]. *IEEE Transactions on Services Computing*, 2015, 13(5):786-800

[24] Wang H, Gu M, Qi Y, et al. Large-scale and adaptive service composition using deep reinforcement learning[C]// Proceedings of the International Conference on Service-oriented Computing, Malaga, Spain, 2017: 383-391

[25] Moustafa A, Ito T. A deep reinforcement learning approach for large-scale service composition[C]// Proceedings of the PRIMA 2018: Principles and Practice of Multi-Agent Systems, Tokyo, Japan, 2018: 296-311

[26] Browne C B, Powley E, Whitehouse D, et al. A survey of Monte Carlo tree search methods[J]. *IEEE Transactions on Computational Intelligence and AI in Games*, 2012, 4(1): 1-43

[27] Thrun S, Littman M L. Reinforcement learning: an introduction[J]. *AI Magazine*, 2000, 21(1): 103

[28] Kaelbling L P, Littman M L, Moore A W. Reinforcement learning: a survey[J]. *Journal of Artificial Intelligence Research*, 1996, 4(1): 237-285

[29] Deleva A. TD Learning in Monte Carlo Tree Search[D]. Ljubljana: Faculty of Computer and Information Science, University of Ljubljana, 2015

[30] Ilhan E, Etaner-Uyar A Ş. Monte Carlo tree search with temporal-difference learning for general video game playing[C]// Proceedings of the 2017 IEEE Conference on Computational Intelligence and Games, New York, USA, 2017: 317-324

[31] Fabbri A, Armetta F, Duchêne É, et al. A self-acquiring knowledge process for MCTS[J]. *International Journal on Artificial Intelligence Tools*, 2016, 25(1): 1-20

[32] Brauner J. Formalizations for geooperators-geoprocessing in spatial data infrastructures[D]. Dresden: Geoinformatics Faculty of Environmental Sciences, Technische Universität Dresden, 2015

[33] Vamplew P, Dazeley R, Foale C. Softmax exploration strategies for multiobjective reinforcement learning[J]. *Neurocomputing*, 2017, 263: 74-86

[34] Zheng Z, Lyu M R. Collaborative reliability prediction of service-oriented systems[C]// Proceedings of the 2010 ACM/IEEE 32nd International Conference on Software Engineering, Cape Town, South Africa, 2010: 35-44

Zhuang Can, born in 1988. She is currently a Ph. D candidate in China University of Geosciences. She received her M. S. degree in China University of Geoscience in 2014. She also received her B. S. degree from Shandong University of Finance in 2009. Her research interests include geospatial service composition and geological big data.