Integrating POMDP and SARSA(λ) for Service Composition with Incomplete Information

Hongbing Wang[1(✉)], Xingzhi Zhang[1], and Qi Yu[2]

[1] School of Computer Science and Engineering, Southeast University, Nanjing, China
hbw@seu.edu.cn, zhangxingzhi777@163.com

[2] College of Computing and Information Sciences, Rochester Institute of Technology, Rochester, USA
qi.yu@rit.edu

Abstract. As a powerful computing paradigm for constructing complex distributed applications, service composition is usually addressed as a planning problem since the goal is to optimize a path for combining services to satisfy special requirements. Some planning methods assume that the state of running environment can be fully observed and monitored. However, the dynamic internet environment and opaque internal status, such as QoS attributes and invoking results, make the assumption too strict and not generally applicable. In this paper, we introduce a Partially Observable Markov Decision Process (POMDP) to model a service composition, which views the environment as partially observable and generates a policy with incomplete information. The partial observability relaxes the previous assumption and can handle the difficulties occurring in a dynamic and unpredictable environment. Based on this model, we propose a reinforcement learning algorithm to compute the optimal strategy. We conduct a series of experiments to verify the proposed algorithm, and compare it the comparison with other two algorithms. The results show the correctness and effectiveness of our algorithm.

1 Introduction

In a Service Oriented Computing (SOC) environment, services that are autonomous, loosely coupled and self-describing, are leveraged as fundamental components to develop interoperable distributed applications. Since a single web service may not satisfy the requirements of complex business requirements, service composition has attracted extensive attention in service computing [5].

In a service composition process, it mainly faces an uncertain and dynamic web environment. On the one hand, in most service-oriented systems, there exists uncertainty in services. For example, some behavior of a service may not be deterministic (which could depend on the input values) and hence the invocation result of the service may be uncertain. As another example, some QoS values of a service may not known in advance and others (e.g., response time and throughput) may change with the dynamic environment. These uncertain

Q.Z. Sheng et al. (Eds.): ICSOC 2016, LNCS 9936, pp. 677–684, 2016.
DOI: 10.1007/978-3-319-46295-0_47

factors can not be detected precisely and affect the process of a service composition. On the other hand, considering that the online services may evolve with the dynamic external environment, a service composition needs to be adaptive to these changes [13]. In summary, a viable solution to service composition should take both the incomplete information and a dynamic environment into consideration, and provide an effective mechanism to satisfy the uncertainty and adaptivity.

To address the challenges of dynamic implicit information in service composition, we propose a novel method that combines Partially Observable Markov Decision Process (POMDP) with reinforcement learning (RL) for service composition. The POMDP refers to a sequential decision-making problem under uncertainty [1,4]. This model does not assume that the environment is fully observable. Instead only some of the features are observed.

We use introduce the POMDP to model the service composition process. In the POMDP service composition framework, an agent does not need the exact state information, which is unavailable. It perceives the environment and gathers observations as computational basis at runtime. Then, the goal is adjusted to find a composition policy that maximizes the reward by the observations and past records. Inspired by the idea of RL, we present a learning algorithm combining eligibility trace (SARSA (λ)), to compose web services in the POMDP framework. Hence, our approach can handle the uncertainty issue in service composition while achieving the adaptivity and efficiency. The rest of this paper is organized as follows. An overview of related work is presented in Sect. 2. The problem formulation and key definitions are given in Sect. 3. The model and algorithm proposed in this paper are described in detail in Sect. 4. Experimental results are presented in Sect. 5 to verify the effectiveness of our approach. Finally, the conclusion is given in Sect. 6.

2 Related Work

In this section, we give an overview of existing approaches that have contributed to service composition, including Markov Decision Process (MDP), reinforcement learning (RL) and Partially Observable Markov Decision Process (POMDP).

Gao et al. [6] and Doshi et al. [3] proposed a web service composition approach based on MDPs. Given the QoS description, Gao et al. described some web service composition patterns, such as sequential, conditional and iterative, in an MDP framework. Doshi et al. developed a policy-based method to model workflow composition to address the issues in classical planning. In [14,16], the authors extended their previous studies to a multi-agent scenario. By combining reinforcement learning and multi-agent mechanism, their methods can improve the efficiency in service compostion.

In real-world applications, due to the dynamic network environments, uncertain performance of services and opaque QoS values, sometimes we can only obtain partial descriptions about the current state. The work in [10] described a

POMDP method to obtain better solutions for QoS-aware service composition, which utilizes the provenance data to assess POMDP distributions. In [15], the POMDP was applied to address the self-healing issue in service-oriented systems. The model works with belief states, which can help determine the best maintenance policy. In [8], the authors introduced the POMDP to model service composition as an uncertain planning problem. They also proposed a time-based learning method to balance the exploration and exploitation. These studies show the promising trend of using the POMDP in service composition.

3 Problem Formulation

In this section, we briefly present the problem description and some preliminaries, such as Web service composition and the Partially Observable Markov Decision Process (POMDP) for web service composition.

Definition 1 (Web Service). *A Web service can be modeled as a tuple* $WS =< ID, Pr, E, QoS >$*, where*

- ID *is a unique id of a Web service;*
- Pr *specifies the preconditions that need to be fulfilled to successfully invoke the Web service;*
- E *specifies the effect to the environment after executing the Web service (including both successful and unsuccessful executions);*
- QoS *is a n-tuple* $< attr_1, \ldots, attr_n >$*, where* $attr_i (1 \leq i \leq n)$ *represents a QoS attribute (e.g., Availability, Reliability, Throughput and Response time);*

Using MDP to model the service composition process has become popular, which reduces the computation cost based on the Markov property [3,6,14]. However, this model assumes that the environment information is fully observable and complete, which is too strict and not suitable for practical application scenarios. As a generalization of an MDP, a POMDP refers to a model for sequential stochastic decision problems that considers the information of system state as uncertain and partially observable [7]. This paper introduces the POMDP model into web service composition (WSC). The definition of POMDP in service composition is given as follows.

Definition 2 (WSC-Partially Observable Markov Decision Process). *A WSC-Partially Observable Markov Decision Process(WSC-POMDP) is formally described as a 6-tuple* $< S, A, R, T, O, Z >$*, where*

- S *denotes a finite set of states of an agent;*
- A *denotes the set of Web services that can be executed;*
- $R : S \times A \rightarrow R$ *denotes the reward function. When an agent invokes a service* $ws \in A$*, the world transits from* s *to* s'*, and the agent receives an immediate reward* r *from the environment;*
- $T : S \times A \times S \rightarrow [0, 1]$ *denotes the transition function. When an agent invokes a service* $ws \in A$*, the world transits from* s *to* s' *with a probability recorded as* $T(s, a, s') = Pr(s'|s, a)$*;*

- O *denotes the observable information received by the agent;*
- $Z : S \times A \times O \rightarrow [0, 1]$ *is the observation function indicating the probability distribution of observations. Formally,* $Z(a, s', o) = Pr(o|s', a)$ *means the probability of observation* o *after agents invoke a service* a *and change to state* s'*;*

An observation history h in a POMDP is defined as a sequence of actions executed and observations received that record the whole evolution of the process [2]. The goal of the agent is to learn a policy π, which maps the observation history h_t into an action a_t at time t to maximize the expected discounted cumulative reward $E[\sum_{t=0}^{T} \gamma^t r^t]$. However, such form of memory can grow indefinitely over time, making it impractical for long planning horizons. Fortunately, we can summarize the unbounded history $h_{1:t-1}$ into a sufficient statistic that compresses all the information of the past actions and observations. This compression pattern has been recognized by numerous studies, and the sufficient statistic is defined as a belief state [7,11], which is a probability distribution over the state space S. Therefore, a POMDP can be cast into a framework of a belief MDP, where the belief states comprise the continuous, but fully observable, MDP state space.

4 Adaptive Service Composition

In this section, we firstly introduce the belief MDP model for web service composition. Then, a RL algorithm is presented to compute the optimal policy in the composition model.

4.1 Belief MDP for Web Service Composition

Based on the above discussion, we convert a WSC-POMDP to a WSC-belief MDP by introducing the concept of belief state, which can be defined as follows:

Definition 3 (Web service composition belief MDP(WSC-belief MDP)). *A WSC-belief MDP is defined as a 5-tuple*$< B, A, O, \Gamma, \rho >$*, where*

- B *is a set of belief states in the continuous space;*
- A *is a set of Web services;*
- O *is a set of possible observations;*
- $\Gamma : B \times A \times O \rightarrow B$ *is the belief transition function;*
- $\rho : B \times A \rightarrow R$ *is the reward function in the belief space, derived from the original reward function on world state,* $\rho(b, a) = \sum_{s \in S} b(s)R(s, a)$*;*

When an agent executes a service a and perceives observation o, the belief state b will be updated to b' based on Bayes rule as follows:

$$b_o^a(s') = Pr(s'|b, a, o) = \eta Pr(o|s', a) \sum_{s \in S} Pr(s'|s, a)b(s) \tag{1}$$

where $\eta = 1/Pr(o|b,a)$ denotes a normalizing constant with $Pr(o|b,a) = \sum_{s' \in S} Pr(o|s',a) \sum_{s \in S} Pr(s'|s,a)b(s)$.

The policy is a function mapping belief state $b(b \in B)$ into service $a(a \in A)$, i.e., $\pi(b) \rightarrow a$. A policy $\pi(b_0)$ can be characterized by the value function $V^{\pi}(b_0)$ that is defined as the expected sum of discounted rewards received by following policy π starting at belief b_0:

$$V^{\pi}(b_0) = \sum_{t=0}^{h} \gamma^t \sum_{s \in S} b_t(s) R(s, \pi(b_t)) \tag{2}$$

where $\pi(b_t)$ represents the action specified by policy π at belief b_t. Based on above definition, it can be derived that an optimal policy π^* is a policy that can maximize its value function V^*(i.e., $V^*(b) \geq V^{\pi}(b)\ \forall(b)$). It prescribes the optimal action for each belief b to execute on time t and assumes that the agent will also act optimally in the future. The optimal value function V^* satisfies the *Bellman's equation*

$$V^*(b) = \max_a [\rho(b,a) + \gamma \sum_o Pr(o|b,a) V^*(b_o^a)] \tag{3}$$

where $Pr(o|b,a) = \sum_{s' \in S} Pr(o|s',a) \sum_{s \in S} Pr(s'|s,a)b(s)$, $\rho(b,a) = \sum_{s \in S} b(s)R(s,a)$ as defined above. b_o^a is the updated belief after performing action a and gathering observation o that is defined by Eq. (1).

4.2 SARSA(λ) Algorithm Based on Belief MDP

As a popular machine learning algorithm, RL assumes that an agent has no perfect knowledge of the environment. It demands the agent to interact with the dynamic environment and learn the optimal strategy with the reward value by the means of trial-and-error. There are many RL algorithms for MDP problems [14,17], such as Q-learning, SARSA, Monte Carlo and Temporal Difference(TD).

SARSA(λ) is a fast multi-step on-policy learning algorithm that introduces eligibility trace into SARSA(0) [9,12]. The basic concept of SARSA(λ) is to apply the TD(λ) prediction method to a state-action pair. It uses experience to learn estimates of an optimal Q-value function. Besides improving the learning efficiency, SARSA(λ) can make full use of past decisions and observations to optimize the current action and obtain the optimal strategy.

In the belief MDP problem, the available transition information at time step t is $< b(s_t), a_t, r_t, b(s_{t+1}) >$ and the Q-value in a belief MDP can be represented as $Q_t(b(s), a)$. Then, the Q-value update formula in SARSA(λ) with belief state is given by.

$$Q_{t+1}(b(s), a) \leftarrow Q_t(b(s), a) + \alpha * \delta_t * e_t(b(s), a) \tag{4}$$

where $0 \leq \alpha \leq 1$ denotes the learning rate, $\delta_t = r_t(b(s), a) + \gamma Q_t(b(s_{t+1}), a_{t+1}) - Q_t(b(s_t), a_t)$. r_t denotes the immediate reward received by an agent after invoking

service a and making the environment transfer from state s_t to s_{t+1}. In addition, the eligibility traces are initialized to 0, and then are updated as follows:

$$e_t(b(s),a) = \begin{cases} \gamma\lambda e_{t-1}(b(s),a) & if\ b(s) \neq b(s_t) \\ 0 & if\ b(s) = b(s_t)\ and\ a \neq a_t \\ 1 & if\ (b(s),a) = (b(s_t),a_t) \end{cases} \tag{5}$$

where $\gamma(0 \leq \gamma \leq 1)$ denotes the discount factor and $\lambda(0 \leq \lambda \leq 1)$ denotes the decay factor.

5 Experiments and Analysis

In this section, we conduct a series of experiments to evaluate our service composition approach. To demonstrate the effectiveness of our approach, we also compare with other similar RL algorithms.

5.1 Experiment Setting

In the experiments, we assign each service node with random QoS values that follow the normal distribution. The algorithm proposed in this paper, i.e., SARSA(λ) for belief MDP, is referred to as b-SARSA(λ). We conduct comparative experiments with Time-based Learning method (referred to as TL) [8], and a multi-agent SARSA algorithm (referred to as multi-SARSA) [14]. A number of key parameters are set as follows. The learning rate α is set to 0.6 similar to the study in [14]. The discount factor γ is set to 0.9 and the ϵ-greedy exploration strategy value is set to 0.6. The experiments are conducted on an Intel i3-2120 3.30 GHz PC with 4 GB RAM.

5.2 Result Analysis

5.2.1 Validation of Effectiveness

In the first experiment, we mainly verify the effectiveness of b-SARSA(λ). The number of successful explorations of an algorithm is used as the standard to compare their convergency rate. The discount cumulative reward received by each algorithm is counted to represent the learning effect.

As shown in Fig. 1(a), b-SARSA(λ) has several superiority compared with other two algorithms. Compared to multi-SARSA, the b-SARSA(λ) algorithm has higher cumulative reward value. The multi-SARSA learns in the MDP model that judges the current state of environment without comprehensive understanding of the environment, while the POMDP computes and analyzes the environment based on belief states. Although multi-agent mechanism can accelerate the convergency rate in multi-SARSA, it obtains less reward value than b-SARSA(λ) in POMDP. On the other hand, in the same POMDP environment, b-SARSA(λ) has faster convergence rate than TL. The reason mainly lies in the introduction of eligibility traces in b-SARSA(λ), which can optimize current action with experience from past decisions to accelerate the learning rate.

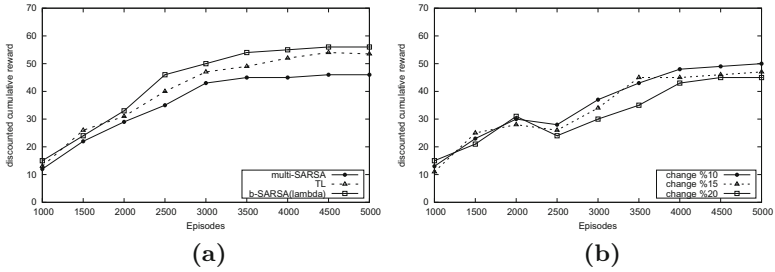

Fig. 1. (a) Validation of effectiveness (b) Validation of adaptability

5.2.2 Validation of Adaptability

Service QoS attributes may change in a dynamic web service environment. Therefore, this experiment mainly studies the adaptability of b-SARSA(λ). We allow 10 %, 15 % and 20 % of changes on the QoS values of a fixed number of candidate services during the learning process to simulate the dynamic environment. Also, we set QoS fluctuations between the 2000th episode and 2500th episode to enhance comparison. Figure 1(b) presents the experiment results. We can see that the b-SARSA(λ) algorithm converges finally in spite of the changes of the QoS, with just different convergence time. Even for huge QoS fluctuations, the reward values temporary decline, but the algorithm will relearn the current strategy and generate a new optimal strategy for the new environment. Therefore, the b-SARSA(λ) algorithm can handle the changes and find the optimal strategy adaptively.

6 Conclusions and Future Directions

In a dynamic service composition environment, the services' internal states and their QoS behaviors may be unpredictable and opaque. As a result, the available information for agents can be implicit. Therefore, this paper presents a reinforcement learning algorithm based on a POMDP model to address the non-observability issue in service composition, which utilizes the partially observed information to make decisions. By introducing the belief state, which presents the probability distributions over all states, the POMDP model can be converted into a belief MDP model. We also introduce the SARSA(λ) algorithm to run in a belief MDP, which combines the on-policy SARSA algorithm with the concept of multi-step prediction from eligibility traces. The algorithm can achieve better learning efficiency while enhancing the adaptability. The experiments validate the effectiveness and superiority of our approach in an incomplete information environment.

Acknowledgments. This work was partially supported by NSFC Projects (Nos. 61672152, 61232007, 61532013), Collaborative Innovation Centers of Novel Software Technology and Industrialization and Wireless Communications Technology.

References

1. Astrom, K.: Optimal control of Markov processes with incomplete state information. J. Math. Anal. Appl. **10**(1), 174–205 (1965)
2. Braziunas, D.: Pomdp solution methods. University of Toronto, Technical Report (2003)
3. Doshi, P., Goodwin, R., Akkiraju, R., Verma, K.: Dynamic workflow composition using Markov decision processes. In: Proceedings of the IEEE International Conference on Web Services, 2004, pp. 576–582. IEEE (2004)
4. Drake, A.W.: Observation of a Markov process through a noisy channel. Ph.D. thesis. Massachusetts Institute of Technology (1962)
5. Dustdar, S., Schreiner, W.: A survey on web services composition. Int. J. Web Grid Serv. **1**(1), 1–30 (2005)
6. Gao, A., Yang, D., Tang, S., Zhang, M.: Web service composition using Markov decision processes. In: Fan, W., Wu, Z., Yang, J. (eds.) WAIM 2005. LNCS, vol. 3739, pp. 308–319. Springer, Heidelberg (2005). doi:10.1007/11563952_28
7. Kaelbling, L.P., Littman, M.L., Cassandra, A.R.: Planning and acting in partially observable stochastic domains. Artif. Intell. **101**(1), 99–134 (1998)
8. Lei, Y., Jiantao, Z., Fengqi, W., Yongqiang, G., Bo, Y.: Web service composition based on reinforcement learning. In: 2015 IEEE International Conference on Web Services (ICWS), pp. 731–734. IEEE (2015)
9. Loch, J., Singh, S.P.: Using eligibility traces to find the best memoryless policy in partially observable Markov decision processes. In: ICML, pp. 323–331 (1998)
10. Naseri, M., Ludwig, S.: Automatic service composition using pomdp and provenance data. In: 2013 IEEE Symposium on Computational Intelligence and Data Mining (CIDM), pp. 246–253. IEEE (2013)
11. Smallwood, R.D., Sondik, E.J.: The optimal control of partially observable markov processes over a finite horizon. Oper. Res. **21**(5), 1071–1088 (1973)
12. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT press, Cambridge (1998)
13. Trummer, I., Faltings, B.: Optimizing the tradeoff between discovery, composition, and execution cost in service composition. In: Proceedings of the IEEE International Conference on Web Services (ICWS), pp. 476–483. IEEE (2011)
14. Wang, H., Chen, X., Wu, Q., Yu, Q., Zheng, Z., Bouguettaya, A.: Integrating on-policy reinforcement learning with multi-agent techniques for adaptive service composition. In: Franch, X., Ghose, A.K., Lewis, G.A., Bhiri, S. (eds.) ICSOC 2014. LNCS, vol. 8831, pp. 154–168. Springer, Heidelberg (2014)
15. Wang, H., Wang, X., Yu, Q.: Optimal self-healing of service-oriented systems with incomplete information. In: 2013 IEEE International Congress on Big Data (BigData Congress), pp. 227–234. IEEE (2013)
16. Wang, H., Wang, X., Zhang, X., Yu, Q., Hu, X.: Effective service composition using multi-agent reinforcement learning. Knowl.-Based Syst. **92**, 151–168 (2016)
17. Wang, H., Wu, Q., Chen, X., Yu, Q.: Integrating gaussian process with reinforcement learning for adaptive service composition. In: Barros, A., Grigori, D., Narendra, N.C., Dam, H.K. (eds.) ICSOC 2015. LNCS, vol. 9435, pp. 203–217. Springer, Heidelberg (2015). doi:10.1007/978-3-662-48616-0_13