Contents lists available at ScienceDirect

Expert Systems With Applications

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

Extracting bottlenecks for reinforcement learning agent by holonic concept clustering and attentional functions

Behzad Ghazanfari*, Nasser Mozayani

School of Computer Engineering, Iran University of Science and Technology, Tehran, Iran

ARTICLE INFO

Keywords:
Reinforcement learning
Abstraction
Concept
Holonic/hierarchical clustering
Attention

ABSTRACT

Reinforcement learning is not well scalable in state spaces with high-dimensions. The hierarchical reinforcement learning resolves this problem by task decomposition. Task decomposition is done by extracting bottlenecks, which is in turn another challenging issue, especially in terms of time and memory complexity and the need to the prior knowledge of the environment. To alleviate these issues, a new approach is proposed toward the problem of extracting bottlenecks. Holonic concept clustering and attentional functions are proposed to extract bottleneck states. To this end, states are organized based on the effects of actions by means of a holonic clustering to extract high-level concepts. High-level concepts are used as cues for controlling attention. The proposed mechanism has a better time complexity and fewer requirements to the designer's help. The experimental results showed a considerable improvement in the precision of bottleneck detection and agent's performance for traditional benchmarks comparing to other similar methods.

1. Introduction

Reinforcement learning (RL) is a general approach to maximize reward through trial and error where reward signals may be with delay (Kaelbling et al., 1996; Sutton & Barto, 1998). This functionality enables RL to be considered as a powerful approach to handling complex non-linear control tasks in fields such as economics, robotics, and control. These fields of research typically confront with large state space and RL cannot be well scaled up in such environments, hence the curse of dimensionality (Barto & Mahadevan, 2003). Decomposing learning task into several sub-problems is an approach to address the challenge of scaling up RL in high-dimensional state space. This division and conquer strategy in RL is used in the form of the hierarchical control and the corresponding learning algorithms, i.e. hierarchical reinforcement learning (HRL). HRL approaches such as options (Sutton et al., 1999), MAXQ (Dietterich, 2000) and HAMs (Parr, 1998) increase the speed of learning in RL mainly by utilizing temporal abstraction (also called temporally extended actions). Semi-Markov decision processes (SMDPs) as extended framework of Markov decision process (MDP) can deal with temporally extended actions (in Section 1.1 more details are given).

Creating temporally extended actions or decomposition of a learning task is obtained by extracting key states in the state space context with some kinds of HRL approaches. These states are categorized into sub-goals and bottleneck states depending on whether they are relevant to the current task (Chiu & Soo, 2011; Mcgovern, 2002; Şimşek & Barto, 2004; Stolle 2004). Extracting these key states is a challenging issue for an RL agent in an automatic manner. Several methods have been proposed to alleviate this challenge. Most of which are generally based on graph partitioning (Chiu & Soo, 2010; Kheradmandian & Rahmati, 2009; Mannor et al., 2004; Menache et al., 2002; Şimşek & Barto, 2009; Şimşek et al., 2005) and path processing mechanisms (Digney, 1998; Mcgovern, 2002; Şimşek & Barto, 2004, 2009; Stolle, 2004; Thrun & Schwartz, 1995). They differentiate the key states from other states by clustering or classification mechanisms based on their own criteria.

The proposed method looks for bottlenecks indirectly by seeking frames of doorways, cues, as the special kind of concepts, which make bottlenecks. If probable frames of doorways are close to each other action spaces, the states among them will be considered as bottleneck states. In fact, they are considered as cues for the possibility of being a bottleneck. To achieve this, the researchers have proposed holonic concept clustering (HCC) which is like a hierarchical multi-resolution structure and attentional functions (AF). HCC works based on the effects of actions and AF detects bottleneck states on the basis of HCC output. These are the key and distinguishing features of the proposed method

* Corresponding author.
E-mail addresses: beghazanfari@gmail.com (B. Ghazanfari), mozayani@iust.ac.ir (N. Mozayani).

http://dx.doi.org/10.1016/j.eswa.2016.01.030
0957-4174/© 2016 Elsevier Ltd. All rights reserved.

as compared with other works cited in the literature. To have comprehensive and formal discussions, four criteria have been introduced to evaluate proposed methods in the literature.

In the following, a rough explanation is presented about MDP as a typical framework of RL and about SMDP as an extended version of MDP for HRL. Afterward, the definitions and explanation are offered for sub-goals, bottlenecks, and the differences between them.

1.1. MDPs and SMDPs

A conventional framework used for modeling discrete time RL is (MDPs) $M: < S, A, P, R >$. $S = \{s_1, \ldots, s_n\}$ is a finite set of states and $A = \{a_1, \ldots, a_p\}$ is a finite set of primary actions; $P: S{*}A{*}S \to [0, 1]$ is a one-step probabilistic state transition and $R : S * A \to \mathbb{R}$ is a reward function. The agent aim is to find the Markov policy (the mapping from states to actions is called a policy) π: $S{*}A \to [0, 1]$ that maximizes its accumulating discounted reward ($R = \sum_{i=0}^{\infty} \gamma r_i, \gamma \in [0, 1]$ where γ is a discounted rate) from each of the states (Sutton & Barto, 1998; Sutton et al., 1999).

In the proposed method, the *option* framework (as shown by (Sutton et al., 1999)) is used to define temporally extended actions. An option is a triplet $< I, \pi, \beta >$, where I is initiation set $I \subseteq S$, i.e. all the states that the option can be initiated from; π: $S{*}A \to [0, 1]$ is a policy, i.e. for each state in option's initiation set there is a mapping to a sequence of actions; and $\beta : S^{+} \to [0, 1]$ is a termination condition which indicates with what probability each option in any state must be terminated. Initiation set and terminal condition indicate the states, which an option can be applied on. Options are close-loop policies and each of them has its policy and value function as they can react to the environmental changes.

1.2. Sub-goals and bottlenecks

Options (Sutton et al., 1999) as an approach of HRL methods improves the agent efficiency in learning and transferring knowledge, especially in complex domains. This improvement is resulted from extending primary actions to temporally extended actions. Creating temporally extended actions is generally based on extracting sub-goals or bottlenecks because sub-goal/bottleneck states have the ability to decompose the learning task. Using sub-goals/bottlenecks to form options is not limited to grid world problems or navigation tasks (Mcgovern, 2002). Extracting these states always has been a challenging issue as they must be extracted accurately and preferably in an automatic manner.

Sub-goals are the states that should provide an easy access or high reinforcement gradients. They must also be visited frequently. A category of methods which extracts sub-goals is known as a set of skill-based methods, such as Thrun and Schwartz (1995), that are based on computing commonalities among occurring sub-policies from the solutions of the related tasks for each of the states. Indeed, the effectiveness of these methods for solving a new task is highly related to previously learned tasks and their reward functions.

Bottlenecks are the states, which provide an easy access to the neighboring regions regardless of whether they are on the successful paths or not. According to Mannor (2004) and Menache et al. (2002), the bottleneck is: "a small set of states which are placed between two dense regions of state space in which transition from one region to the other have a low probability". Bottlenecks are defined as independent of the reward function; therefore they can be used in a variety of tasks, like cases in which the environment has a same state transition matrix but different reward function (Şimşek et al., 2005). They are typically extracted by methods based on a state transition graph (Chiu & Soo, 2010; Kheradmandian & Rahmati, 2009; Mannor et al., 2004; Menacheet al., 2002; Şimşek & Barto, 2009; Şimşek et al., 2005).

1.2.1. Topology bottlenecks and value bottlenecks

Bottleneck states and approaches which extract them can be divided into two groups: topology-based and value-based (Kheradmandian & Rahmati, 2009; Mannor et al., 2004; Şimşek & Barto, 2009). In both of them, a graph is formed based on state-action transitions to extract bottlenecks. A mechanism is considered as value-based approach if reward function or states values are somehow directly applied in the process of identifying bottlenecks, like Digney (1998), Mcgovern (2002), Şimşek and Barto (2004, 2009), Stolle (2004), and Thrun and Schwartz (1995), and the extracted bottlenecks are regarded as value bottlenecks. Topology-based methods look for the states which separate the state-action transition graph into parts that are densely connected internally but externally connected sparsely, such as Chiu and Soo (2010), Kheradmandian and Rahmati (2009), Mannor et al. (2004), Menache et al. (2002), Şimşek and Barto (2009), and Şimşek et al. (2005). In this paper, value-based bottlenecks will be considered in an offline manner and topology-based ones in an offline and an online manner.

The organization of the rest of this paper is as follows. Section 2 represents the literature review with regard to the context of creating temporally extended actions in RL. Also, the theoretical background of the proposed method as well as the conceptual similarities of its key modules to other methods in the context of visual processing and cognitive science and the main contributions of the proposed method is presented. Sections 3 and 4 will describe the framework in the form of modules in a formal way and in online manner respectively. And also, the details of how these modules work and their relationships are discussed, and the formulas are calculated and clarified in the form of a simple example. Afterward, the effects of the proposed method on the precision and the speed of learning will be shown in Section 5. In Section 6, the advantages of the mechanism of the proposed method will be described based on its evaluation in the form of four basic criteria. Finally, the conclusion and discussion of the study will be presented in Section 7.

2. Related works

The methods proposed to extract bottlenecks or sub-goals so far can be categorized into two groups. First, methods such as Digney (1998), Mcgovern (2002), Şimşek and Barto (2004, 2009), Stolle (2004), and Thrun and Schwartz (1995) which calculate their criteria among different paths of the agent or shortest paths between nodes of graph. They typically get in trouble in more severe form as state space becomes large, and also when the required number of actions for attaining the goal increases (delay reward). The second group includes methods, like Chiu and Soo (2010), Kheradmandian and Rahmati (2009), Mannor et al. (2004), Menache et al. (2002), Şimşek and Barto (2009), and Şimşek et al. (2005), that calculate some proposed criteria among enormous combinations of state clusters among objects to find bottlenecks. They are generally based on density measures to cluster state space and to evaluate these clusters. The required parameters of these approaches as threshold values are used to indicate possible bottleneck states. Large state spaces issue makes real troubles in regard to the time required for extracting bottlenecks. These methods also disregard some bottlenecks if some clusters have smaller densities comparing to other neighbor's clusters. Both groups of these methods are based on clustering or classification approach to separate sub-goals/bottlenecks. In both approaches, it is not computationally tractable to calculate their metrics regarding different