Knowledge-Based Systems 000 (2017) 1–7

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Tackling the start-up of a reinforcement learning agent for the control of wastewater treatment plants

Félix Hernández-del-Olmo[a,*], Elena Gaudioso[a], Raquel Dormido[b], Natividad Duro[b]

[a] Department of Artificial Intelligence, National Distance Education University (UNED), Madrid 28040, Spain
[b] Department of Computer Sciences and Automatic Control, National Distance Education University (UNED), Madrid 28040 Spain

ARTICLE INFO

Article history:
Received 22 April 2017
Revised 13 December 2017
Accepted 15 December 2017
Available online xxx

Keywords:
Reinforcement learning
Wastewater systems
Intelligent agent
Adaptive control

ABSTRACT

Reinforcement learning problems involve learning by doing. Therefore, a reinforcement learning agent will have to fail sometimes (while doing) in order to learn. Nevertheless, even with this starting error, introduced at least during the non-optimal learning stage, reinforcement learning can be affordable in some domains like the control of a wastewater treatment plant. However, in wastewater treatment plants, trying to solve the day-to-day problems, plant operators will usually not risk to leave their plant in the hands of an inexperienced and untrained reinforcement learning agent. In fact, it is somewhat obvious that plant operators will require firstly to check that the agent has been trained and that it works as it should at their particular plant. In this paper, we present a solution to this problem by giving a previous instruction to the reinforcement learning agent before we let it act on the plant. In fact, this previous instruction is the key point of the paper. In addition, this instruction is given effortlessly by the plant operator. As we will see, this solution does not just solve the starting up problem of leaving the plant in the hands of an untrained agent, but it also improves the future performance of the agent.

© 2017 Elsevier B.V. All rights reserved.

1. Introduction

Reinforcement learning (RL) is a machine learning paradigm where the agent learns to do better in its environment by interacting with it [29]. RL has been applied to different domains which include: medical applications such as optimization of anemia treatment [12], control of blood glucose variability [10], robotics [27], operations research (e.g. optimization the pricing policy of a cloud service provider [36] or web services [32]). RL has also been successfully applied to the intelligent control of processes. In this domain we could distinguish between process stabilization [30,37,38], process tracking for fault-tolerant controllers (FTC) [17,33,39] or process optimization [1,13]. The latter is the domain we focus on for the control of WasteWater Treatment Plants.

The main function of WWTPs is to provide humans and industries mechanisms for disposing effluents to protect the natural environment. Since WWTPs are significant energy consumers, optimizating them implies cutting operating costs and effluent fines, while rising to the challenges of water quality, sustainable development and even stringent regulations. Traditionally, WWTPs are controlled using standard control techniques [2], fuzzy control [3] or even artificial neural networks [15]. RL has also been successfully used in WWTPs. For example, in [23] a RL approach is proposed in order to increase methane production during anaerobic digestion of wastewater sludge. In [19], we improve the energy and environmental efficiency of a WWTP in N-Ammonia removal proccess by means of an RL agent.

Nevertheless, the high volume of data needed when the agent learns *online* –when the agent learns at the same time it is interacting with its environment– is currently one of the most important limitations for the application of RL [11]. Notice that, different from other kind of machine learning paradigms, there is no supervisor to *label* the data, either for learning or for evaluating the learning. Instead, only a signal that comes from the environment, the reinforcement, tells the agent how it is doing its job in the log run.

In the case of the control of a WWTP, the reinforcement signal can be a measure that combines energy and environmental efficiency [20–22]. In this case, plant operators are in charge of setting up this measure and tunning up the control (or RL agent control) of their WWTP. Nevertheless, as it may seem obvious, operators require that the application works from the very beginning. The problem we face is summarized as follows: how to get these interactive data if we cannot let the agent act on the plant first.

The solution we propose in this paper consists of applying a first stage of *instruction time*: a time in which the agent learns

* Corresponding author.
E-mail address: felixh@dia.uned.es (F. Hernández-del-Olmo).

https://doi.org/10.1016/j.knosys.2017.12.019
0950-7051/© 2017 Elsevier B.V. All rights reserved.

Please cite this article as: F. Hernández-del-Olmo et al., Tackling the start-up of a reinforcement learning agent for the control of wastewater treatment plants, Knowledge-Based Systems (2017), https://doi.org/10.1016/j.knosys.2017.12.019

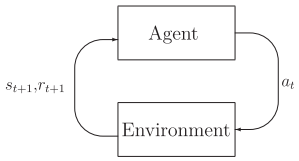

Fig. 1. General schema of a reinforcement learning task.

from the actions taken by the plant operator. After this instruction time, we leave the agent to learn by doing, this is, by interacting with the plant by means of the knowledge acquired during the first instruction stage.

The rest of this paper is organized as follows. Section 2 presents the required reinforcement learning background to grasp the main ideas of this paper. Section 3 presents a brief explanation of our previous work [20] since this paper is a continuation of it. Section 4 focuses on the solution proposed. Section 5 shows the results obtained when we apply the solution proposed and compare them with our previous work. Finally, in Section 6 we discuss these results and some work we have left to do for the near future.

2. Reinforcement learning background

Reinforcement learning (RL) algorithms are based on agent's interaction with its environment. The environment is defined as any external condition that cannot be changed directly by the agent [29], but can be changed through its actions. In fact, this interaction is usually represented as in Fig. 1.

2.1. Elements of reinforcement learning

The usual way the environment is modeled in RL is by means of Markov decision processes (MDP). Here, the MDP environment is modeled as (i) a space of states S, (ii) a space of actions $A(s)$ that can be done over this environment, given that the environment is in state s, and (iii) a set of transition probabilities from one state s to another state s' once the agent has executed action a over this environment $P(s'|s, a)$ besides (iv) the expected reward to be obtained from this environment $E\{r|s', a, s\}$ when changing from state s to state s' having executed action a [29].

Once the agent has this model of the environment, the optimal policy $\pi(s, a)$ can be solved by several methods, for instance dynamic programming [4]. However, if the model of the environment is not provided to the agent, it can still learn this model by means of the so called *model-free* RL methods [16]. Thus, with these model-free RL methods, the agent must interact with its environment so as to get, step by step, the model of the environment as well as the optimal policy to act upon it.

More specifically, the agent interacts with its environment making decisions according to its observations, via perception and action. At each step t, the agent observes the current state of the environment s_t and chooses an action to execute, a_t. This action causes a transition between states and the environment provides a new state s_{t+1} and a reward r_{t+1} to the agent. The ultimate goal of the agent is to choose those actions that tend to increase its *return*: the long-term sum of the future reward values r_t. This return, in a continuous environment, is usually set as $R_t = \sum_{t'=t}^{\infty} \gamma^{(t'-t)} r_t$, where $0 < \gamma < 1$ stands for a kind of *Optimization Horizon* (OH, as we will see later). In other words, the higher the γ (up to 1), the further the future time considered into the return R_t. In addition, if the application has a large number of states or if it is continuous, we need to approach the problem by means of function approximation [7].

Finally, in the case of the WWTP domain, a complete MDP model is not usually available, thus, a model-free RL approach is required. That is why an agent has to interact with the environment to estimate it. Notice, however, that the goal does not consist of getting the complete estimation of the MDP, but just to calculate the best policy π. This is, the one that gets a higher return. To this end, in [20] we took advantage of the widely known method called *policy iteration* [7,29].

2.2. Reinforcement learning by policy iteration

The objective of reinforcement learning consists of searching for the policy that provides the agent with the action that, for a given state, will carry the agent to a next state maximizing the next and the following rewards (the agent's return). This is called the *optimal policy* $\pi^*(s)$ and this policy is unique [4]. In reinforcement learning, to find this $\pi^*(s)$, the agent must interact with the environment by means of some previous non-optimal policies $\pi_t(s)$.

For the sake of accuracy, we will define some new concepts in the next paragraphs. The return obtained by a policy π interacting in an MDP when the agent starts at state s is defined as $R^\pi(s)$. When working with a known MDP, this return is usually called $V^\pi(s)$ and it is what we need to solve the problem in a *dynamic programming problem* [4]. However, in reinforcement learning (where we do not have the complete MDP available), we must work with the so called Q values: $Q^\pi(s, a)$. Each $Q^\pi(s, a)$ value is defined as the return obtained when the agent follows the policy π starting at the state s and taking a as the next immediate action.

Policy iteration consists of an iterative process in which the policy π_t followed by the agent at time t is monotonically improved at each step letting *policy evaluation* and *policy improvement* processes interact [29]. Policy evaluation consists of calculating a new estimation of $Q^{\pi_t}(s, a)$ after each interaction with the environment under this policy π_t. Policy improvement consists of getting the new policy π_{t+1} from the last policy evaluation $Q^{\pi_t}(s, a)$. Algorithm 1 shows the pseudocode that details this explanation.

Algorithm 1 Reinforcement learning by policy iteration.

```
1:  function PI(π_0, s_0)
2:      t ← 0
3:      π ← π_0
4:      π' ← π
5:      a ← π(s_0)
6:      repeat
7:          s_{t+1}, r_{t+1} ← Environment(s_t, a)              ▷ Interaction
8:          a' ← π'(s_{t+1})
9:          Q^{π'}_{t+1} ← UpdateQ(Q^π_t, s_t, a, r_{t+1}, s_{t+1}, a')   ▷ Policy
    evaluation
10:         π ← π'
11:         for ∀s do
12:             π'(s) ← arg max_a[Q^π_{t+1}(s, a)]   ▷ Policy improvement
13:         t ← t + 1
14:         a ← a'
15:     until π' = π
    return π* = π
```

Fig. 2 depicts the idea behind.

Finally, notice that in Algorithm 1 the function `UpdateQ` is still to be defined. In the next section we will show the two major ways of updating Q: the so-called *on-policy* and *off-policy* control methods.

2.3. On-policy and off-policy control methods

In this section we will focus on line 9 of Algorithm 1: policy evaluation. This point is a very important one, because the way the agent updates Q can make it converge to Q^* or, instead, make it diverge badly. Thus, there are two major approaches: the on-policy

Please cite this article as: F. Hernández-del-Olmo et al., Tackling the start-up of a reinforcement learning agent for the control of wastewater treatment plants, Knowledge-Based Systems (2017), https://doi.org/10.1016/j.knosys.2017.12.019

Download English Version:

https://daneshyari.com/en/article/6861707

Download Persian Version:

https://daneshyari.com/article/6861707

Daneshyari.com