Operations Research Letters 44 (2016) 267–272

Contents lists available at ScienceDirect

Operations Research Letters

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

Improved bound on the worst case complexity of Policy Iteration

Romain Hollanders [*], Balázs Gerencsér, Jean-Charles Delvenne [1], Raphaël M. Jungers [2]

Euler Building, Avenue G. Lemaître 4, 1348 Louvain-la-Neuve, Belgium

ARTICLE INFO

Article history:
Received 19 May 2015
Received in revised form
26 January 2016
Accepted 26 January 2016
Available online 8 February 2016

Keywords:
Policy Iteration
Complexity
Markov Decision Process
Acyclic Unique Sink Orientation

ABSTRACT

Solving Markov Decision Processes is a recurrent task in engineering which can be performed efficiently in practice using the Policy Iteration algorithm. Regarding its complexity, both lower and upper bounds are known to be exponential (but far apart) in the size of the problem. In this work, we provide the first improvement over the now standard upper bound from Mansour and Singh (1999). We also show that this bound is tight for a natural relaxation of the problem.

© 2016 Elsevier B.V. All rights reserved.

1. Introduction

Markov Decision Processes (MDPs) have been found to be a powerful modeling tool for the decision problems that arise daily in various domains of engineering such as control, finance, queuing systems, PageRank optimization, and many more (see [23] for a more exhaustive list).

MDPs are described from a set of n states in which a system can be. When being in a state, the controller of the system must choose an available action in that state, each of which induces a reward and moves the system to another state according to given transition probabilities. In this work, we assume that the number of actions per state is bounded by a constant k. A policy refers to the stationary choice of one action in every state. Choosing a policy implies fixing a dynamics that corresponds to a Markov chain. Given any policy (there are at most k^n of them), we can associate a value to each state of the MDP that corresponds to the infinite-horizon expected reward of an agent starting in that state. By solving an MDP, we mean providing an optimal policy that maximizes the value of every state. Depending on the application, a total-, discounted- or average-reward criterion may be best suited to define the value function. In every case, an optimal policy always exists. See, e.g., [20] for an in-depth study of MDPs.

A practically efficient way of finding the optimal policy for an MDP is to use Policy Iteration (PI). Starting from an initial policy π_0, $i = 0$, this simple iterative scheme repeatedly computes the value of π_i at every state and greedily modifies this policy using its evaluation to obtain the next iterate π_{i+1}. The modification always ensures that the value of π_{i+1} improves on that of π_i at every state. The process is then repeated until convergence to the optimal policy π^* in a finite number of steps (obviously at most k^n steps—the maximum number of policies). We refer to the ordered set of explored policies as the PI-sequence. A more precise statement of the algorithm as well as some important properties are described in Section 2.

Every iteration of the algorithm can be performed in polynomial time and its number of steps has been shown by Ye to be strongly polynomial in the important particular case of discounted-reward MDPs with a fixed discount rate [24] (the bound in this result was later improved in [14,21]). Building on this result, similar conclusions were obtained for other special cases of MDPs [19,4,1,6]. Ye's result does however not extend to *Value Iteration* and *Modified Policy Iteration*, the two standard and closely related competitors of PI [5,7].

In contrast to these positive results, the number of iterations of PI can be exponentially large in general. Based on the work of Friedmann on Parity Games [8], PI has been shown to require at least $\Omega(2^{n/7})$ steps to converge in the worst case for the total- and average-reward criteria [3] and for the discounted-reward criterion [17]. Friedmann's result was also a major milestone for the study of the Simplex algorithm for Linear Programming as it led to exponential lower bounds for some critical pivoting rules

[*] Corresponding author. Tel.: +32 0 473 66 25 97.
E-mail addresses: romain.hollanders@gmail.com (R. Hollanders), balazs.gerencser@uclouvain.be (B. Gerencsér), jean-charles.delvenne@uclouvain.be (J.-C. Delvenne), raphael.jungers@uclouvain.be (R.M. Jungers).

[1] CORE and NAXYS fellow.

[2] F.R.S./FNRS Research Associate.

http://dx.doi.org/10.1016/j.orl.2016.01.010
0167-6377/© 2016 Elsevier B.V. All rights reserved.

[9,10]. On the other hand, the best known upper bound for PI to date was due to Mansour and Singh with a $13 \cdot \frac{k^n}{n}$ steps bound [18]. In Theorem 1, Section 3, we provide the first improvement in fifteen years to this bound, namely $\frac{k}{k-1} \cdot \frac{k^n}{n} + o\left(\frac{k^n}{n}\right)$.

To obtain our bound, we use a number of properties of PI-sequences. It is of natural interest to explore which of these properties could be further exploited to improve the bound and which ones cannot. It turns out that the properties we actually use to obtain our upper bound cannot lead to further improvements, that is, they are "fully exploited". To formally prove this fact, we introduce in Section 2 the notion of *pseudo-PI-sequence* to describe any sequence of policies satisfying only the properties that we use to obtain our bound from Theorem 1. We then show in Theorem 2, Section 3, that there always exists a pseudo-PI-sequence whose size matches the upper bound of Theorem 1. This confirms that the bound is sharp for pseudo-PI-sequences. Therefore, obtaining new bounds on PI-sequences would require exploiting stronger properties.

An attempt in that direction – based on the so-called *Order-Regular* matrices – has been proposed in [13] and developed in [12]. Based on numerical evidence, Hansen and Zwick conjectured that the number of iterations of PI for $k = 2$ should be bounded by F_{n+2} $(= O(1.618^n))$, the $(n+2)$nd Fibonacci number. If true, this bound would significantly improve ours.

As a final remark, note that our analysis also fits in the frameworks of the *Strategy Iteration* algorithm to solve 2-Player Turn-Based Stochastic Games [15] – a 2-player generalization of MDPs – and of the *Bottom Antipodal* algorithm to find the sink of an Acyclic Unique Sink Orientation of a grid [22,11]. Our bound can also be adapted for these algorithms. It is to be noted that no polynomial-time algorithm is known for either case, which is an additional incentive to improve the exponential bounds (although the strongly polynomial time bound from Ye [24] extends to 2TBSGs as well when a fixed discount factor is chosen [14]).

2. Problem statement and preliminary results

Definition 1 (*Markov Decision Process*). Let $\mathcal{S} = \{1, \ldots, n\}$ be a set of n *states* and $\mathcal{A}_s$ be a set of k *actions* available for state $s \in \mathcal{S}$. To each choice of an action corresponds a *transition probability* distribution for the next state to visit as well as a *reward*. For simplicity, we use a common numbering for the actions, that is, $\mathcal{A}_s \triangleq \mathcal{A} = \{1, \ldots, k\}$ for all $s \in \mathcal{S}$. With this notation, for every pair $(s, a) \in \mathcal{S} \times \mathcal{A}$, the transition probability and reward functions are uniquely defined. Let a *policy* $\pi \in \{1, \ldots, k\}^n$ be the stationary choice of one action for every state. A policy induces a transition probability matrix P^π corresponding to some Markov chain and a reward vector r^π. We may ask how rewarding a policy π is in the long run. This is represented by its *value* vector $v^\pi \in \mathbb{R}^n$ whose sth entry corresponds to the long term reward obtained from starting in state s and following the policy π thereafter. It can be computed by solving a system whose definition depends on the problem studied. For instance, for the standard *infinite-horizon discounted-reward criterion* where the aim is to maximize the discounted sum of rewards, v^π is obtained by:

$$v^\pi = \sum_{i=0}^{\infty} (\gamma P^\pi)^i \, r^\pi = (I - \gamma P^\pi)^{-1} r^\pi,$$

where $0 \le \gamma < 1$ is the discount factor that ensures that $(I - \gamma P^\pi)$ is non-singular. However, in this work, the bounds that we derive hold for the three classical reward criteria, namely the discounted-, total- and average-reward criteria. For the total-reward criterion, we assume the existence of a terminal – reward free – state that we assume to be reachable with any policy, from any starting state. For the average-reward criterion, we need to extend the notion of value vectors to *valuations* as defined in [13, Section 2.3] or [20, Section 8.2.1]. By *solving* an MDP, we mean finding the optimal policy π^* such that for any other policy π, $v^{\pi^*} \ge v^\pi$, that is, $v^{\pi^*}(s) \ge v^\pi(s)$ for all states s. The existence of such a policy is guaranteed [2].

Definition 2 (*Domination*). Given two policies π and π', if $v^{\pi'}(s) \ge v^\pi(s)$ for all states $s \in \mathcal{S}$, then we say that π' *dominates* π and we write $\pi' \succeq \pi$. If moreover $v^{\pi'}(s) > v^\pi(s)$ for at least one state, then the domination is strict and we write $\pi' \succ \pi$. An analogous definition of domination can be obtained for valuations with the average reward criterion [13, Section 2.3].

Definition 3 (*Switching*). Let U be a collection of state–action pairs (s, a). We say that U is *well-defined* if it contains every state $s \in \mathcal{S}$ at most once. In that case, we define $\pi' = \pi \oplus U$ to be the policy obtained from π by *switching* the action $\pi(s)$ to a for each (s, a)-pair in U.

Definition 4 (*Improvement Set*). We define the *improvement set* of a policy π as:

$$T^\pi = \left\{(s, a) \mid \pi \oplus \{(s, a)\} \succ \pi\right\},$$

and the set of *improvement states* S^π of π as the set of states that appear in T^π.

Proposition 1. *Let π be a policy and $U \neq \emptyset$ be any well-defined subset of its improvement set T^π. Then $\pi \oplus U \succ \pi$.*

Proposition 2. *For a given policy π, if $T^\pi = \emptyset$, then π is optimal.*

Proofs of Propositions 1 and 2 can be found, e.g., in [13]; see Theorems 2.2.12, 2.3.9 and 2.4.6 for the discounted-, average- and total-reward criteria, respectively. Alternative statements can also be found in [20,2]. Based on Propositions 1 and 2 we may define the *Policy Iteration* algorithm to find the optimal policy. In the rest of our analysis, we only assume that these two propositions hold.

Definition 5 (*Policy Iteration*). Algorithm 1 describes *Policy Iteration* (PI). The standard way of choosing $U_i \subseteq T^{\pi_i}$ is the greedy update rule, namely choose any U_i with maximal cardinality $|S^{\pi_i}|$. We refer to the corresponding algorithm as *Greedy PI*, which is the focus of our work.

```
Initialization: $\pi_0, i = 0$
while $T^{\pi_i} \neq \emptyset$ do
    Select a non-empty and well-defined $U_i \subseteq T^{\pi_i}$
    $\pi_{i+1} = \pi_i \oplus U_i$
    $i \leftarrow i + 1$
end
return $\pi_i$
```

Algorithm 1: Policy Iteration

Definition 6 (*Comparable*). We say that two policies π and π' are *comparable* if either $\pi \preceq \pi'$ or $\pi \succeq \pi'$. We call two policies *neighbors* if they differ in only one state. Neighbors are always comparable (Lemma 3 in [18]).

Definition 7 (*Partial Order*). For a given MDP, we consider the *partial order* PO of the policies defined by the domination relation. A set of policies $\pi^{(1)}, \ldots, \pi^{(k)}$ is called a *sequence* if $\pi^{(1)} \preceq \cdots \preceq \pi^{(k)}$.

Definition 8 (*PI-sequence*). We refer to the sequence of policies $\pi_0, \ldots, \pi_{m-1}$ explored by greedy PI as a *PI-sequence* of length m.

We aim to solve the following problem.

Problem 1. Find the longest possible PI-sequence.

Download English Version:

https://daneshyari.com/en/article/1142229

Download Persian Version:

https://daneshyari.com/article/1142229

Daneshyari.com