European Journal of Operational Research 229 (2013) 190–198

Contents lists available at SciVerse ScienceDirect

European Journal of Operational Research

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

Decision Support

Accelerating the convergence of value iteration by using partial transition functions

Edilson F. Arruda [a,*], Fabrício O. Ourique [b], Jason LaCombe [c], Anthony Almudevar [c]

[a] Industrial Engineering Program, Alberto Luiz Coimbra Institute: Graduate School and Research in Engineering, Federal University of Rio de Janeiro, Caixa Postal 68507, Rio de Janeiro, RJ 21941-972, Brazil
[b] Federal University of Santa Catarina, Campus Araranguá, Rua Pedro João Pereira, 150, Araranguá, SC 88900-000, Brazil
[c] Department of Biostatistics, University of Rochester Medical Center, 601 Elmwood Avenue, Box 630, Rochester, NY 14642, USA

ARTICLE INFO

Article history:
Received 27 April 2012
Accepted 14 February 2013
Available online 27 February 2013

Keywords:
Dynamic programming
Markov processes
Optimization

ABSTRACT

This work proposes an algorithm that makes use of partial information to improve the convergence properties of the value iteration algorithm in terms of the overall computational complexity. The algorithm iterates on a series of increasingly refined approximate models that converges to the true model according to an optimal linear rate, which coincides with the convergence rate of the original value iteration algorithm. The paper investigates the properties of the proposed algorithm and features a series of switch-hover queue examples which illustrates the efficacy of the method.

1. Introduction

Information is a very valuable resource in optimization and decision making. Indeed, in the particular case of Markov decision problems (MDPs), algorithms that use a model of the environment – which can either be available a priori or learned from data – tend to perform better while trying to retrieve the optimal control policy for such problems, (e.g., Atkeson and Santamaria, 1997; Sutton and Barto, 1998). While reinforcement learning techniques and approximate dynamic programming (e.g., Sutton and Barto, 1998; Powell, 2007; Arruda et al., 2011) are becoming increasingly popular for problems with incomplete information and/or very large state spaces, dynamic programming (DP) (Puterman, 1994) remains as the most powerful technique for optimally solving problems with moderate dimension and complete information, which are the problems we address in this paper.

MDP and dynamic programming are natural tools for modeling and solving operations research problems with uncertainties (e.g., He et al., 2012; Saure et al., 2012; Arruda and do Val, 2008; Hao et al., 2008). Value iteration (VI) and policy iteration are the most popular DP algorithms (Puterman, 1994), and VI is arguably the most implemented of the two. It is a very elegant tool to build the optimal solution iteratively while making use of the complete information about the problem to be solved. Given that the algorithm is very effective and uses all information about the probabilistic transitions and value function estimations, a natural question that comes to mind is whether the underlying method can be improved with a more efficient use of information. An efficient alternative would be to focus the computational resources within promising areas of the search space. That is the motivation of prioritized sweeping (PS), an influential algorithm based on value iteration and proposed by Moore and Atkeson (1993), for which several extensions were proposed and applied (e.g., Wingate and Seppi, 2005; Akramizadeh et al., 2011). These algorithms, which make use of a priority queue to guide asynchronous updates of the value function aimed at better exploiting the most promising regions of the state space, can be very effective (Wingate and Seppi, 2005). Their performance, however, depends on the tuning of parameters which determine how much processing of the priority queue is permitted at each iteration, the actual size of the priority queue in the implementation, as well as the minimum requirements for a state to enter the queue. For details on the implementation of the PS algorithm, we refer to Moore and Atkeson (1993). Other popular techniques are heuristic search (Hansen and Zilberstein, 2008), and real-time dynamic programming (RTDP) (Barto et al., 1995), which makes use of asynchronous updates and heuristic search to accelerate convergence. A labeling scheme was proposed by Bonet and Geffner (2003a,b) to improve the performance of RTDP algorithms. Later, Dai and Goldsmith (2007) introduced the Topological Value Iteration Algorithm, that makes use of graphical features of MDPs to decide the order of value function updates. Such an algorithm was later refined to eliminate sub-optimal actions by means of heuristic search (Dai et al., 2009).

This paper develops the Partial Information Value Iteration Algorithm (PIVI), an algorithm intended to take advantage of partial information in order to improve the convergence properties of the value iteration algorithm in terms of computational

* Corresponding author. Tel.: +55 21 2562 8255; fax: +55 21 2270 9702.
E-mail address: efarruda@po.coppe.ufrj.br (E.F. Arruda).

0377-2217/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.ejor.2013.02.029

complexity. The idea underlying PIVI is to increase the efficiency in the use of the modeling data, which consist of the transition function and the cost function of the MDP. The objective is to save computational resources (time) in the early iterations by using gross approximate models that are increasingly refined as one gets closer to the fixed point. The approximate models are generated by truncating the transition probability distributions at each iteration within a prescribed tolerance, while decreasing this tolerance at each iteration. We make use of the theoretical results introduced in two earlier papers (Almudevar, 2008; Almudevar and Arruda, 2012). These results are applied to make sure that the sequence of approximate models is refined in an optimal way with respect to the computational complexity, ensuring minimization of the overall computation time.

The proposed algorithm is inspired by coarse-to-fine multigrid algorithms (Chow and Tsitsiklis, 1991). Like multigrid algorithms, PIVI iterates on a sequence of increasingly accurate approximate models. However, unlike multigrid methods, which usually require convergence at each approximate model, the proposed algorithm spends a single iteration at each approximate model, requiring convergence only for the exact model. Moreover, PIVI is designed according to a theory that guides the optimal choice of the refinement rate for the approximate models, in such a way that the overall computation time is minimized (Almudevar and Arruda, 2012).

The schedule at which new probabilities are added is an important issue for the method. One wants to release new probabilities slowly enough so that we can take advantage of partial information, but quickly enough so that we do not waste computation in intermediate models, thus compromising the computational gains. Fortunately, results from a previous work yield that the optimal rate of model refinement is linear and coincides with the convergence rate of the original algorithm (Almudevar and Arruda, 2012). This rate is optimal in the sense that it ensures convergence with minimum expenditure of computation (Almudevar and Arruda, 2012, Section IV). Hence, the proposed algorithm has a performance guarantee: it is designed based on a theory that guides the optimal choice of the refinement rate, in such a way that the fastest rate of convergence with respect to the computation time is attained. We also stress that the optimal tolerance refinement schedule, which is derived, depends on no user defined parameters.

We investigate the properties of the proposed algorithm and design network experiments that highlight its strengths and weaknesses, while also contrast its performance to that of the PS algorithm. We also evaluate its efficacy on a series of switchover queue problems for which significant gains in convergence time are obtained over the standard VI algorithm. For more details on switchover queue problems and their applications, we refer to Rosa-Hatko and Gunn (1997). We selected the switchover queue because its inherent structure, which typically induces exponential probability distributions, makes it an excellent application for PIVI, see Section 5.

This paper is organized as follows. Section 2 introduces the studied problem and the proposed PIVI algorithm. Section 3 presents general convergence properties of approximate VI algorithms, whereas Section 4 focuses on the convergence properties of PIVI. Section 5 investigates the evolution of the computational complexity for the proposed algorithm and is followed by Section 6, which analyzes the strengths and drawbacks of the algorithm and contrasts it to PS, in the light of simple network examples. Section 7 features a series of switchover queue examples and Section 8 concludes the paper.

2. Mathematical formulation

We consider an infinite-horizon discrete-time stochastic control problem, modeled as a Markov decision process (MDP) with a finite discrete state space S, finite action space, and discounted cost. Note that the discrete-time approach that we study can also be used to model (finely) discretized versions of continuous time stochastic control problems. For detailed treatments of Markov decision processes, we refer to Puterman (1994), Bertsekas (1995).

Let A be a finite discrete action set. For each $i \in S$, any action from nonempty set $A(i) \subset A$ is available. This defines state-action space $Z = \{(i,a): i \in S, a \in A(i)\}$. The selection of a given deterministic action $a \in A(i)$ at any given period $k \geqslant 0$ results in a probabilistic transition to some state $j \in S$ at period $k+1$, and $p_j(i,a)$ denotes the probability that the system reaches state $j \in S$ in the next period. For such a problem, a stationary or deterministic (control) policy is a mapping $\pi: S \to A$, which, for any given state $i \in S$, prescribes the same action $\pi(i) \in A(i)$, any time the system reaches state i. Thus, regardless of the system's history, every time a state $i \in S$ is visited, the corresponding action $\pi(i)$ is taken. Standard dynamic programming (DP) theory yields that an optimal policy for this problem belongs to the class of stationary policies, (e.g., Bertsekas, 1995). Therefore, the search space of the VI algorithm can be restricted to this class.

Once a stationary policy π is fixed, the sequence of states i_k becomes a homogeneous Markov chain with transition probabilities

$$P(i_{k+1} = j | i_k = i) = p_j(i, \pi(i)),$$

or $p_j(i,\pi)$ for short. We say that a stationary control policy is feasible if, for each state $i \in S$, it prescribes a control action that is available for state i, i.e., if $\pi(i) \in A(i))$, $\forall i \in S$. Let Π be the set of all feasible stationary control policies. A cost described by a bounded cost function, denoted as $c: Z \to \mathbb{R}$, is incurred at each period. Assume:

$$\min_{a \in A(i)} c(i,a) < \infty, \quad \text{for all } i \in S, \tag{1}$$

Letting $\alpha \in (0,1)$ be a discount factor, the total expected cost starting from a state $i \in S$ and using a policy π is

$$V^{\pi}(i) = \lim_{N \to \infty} E\left\{ \sum_{k=0}^{N-1} \alpha^k c(i_k, \pi) | i_0 = i \right\}. \tag{2}$$

where $c(i,\pi)$ is short for $c(i,\pi(i))$. Note that, since the cost function, defined in (1), is bounded, the $V^{\pi}: S \to \mathbb{R}$ exists and is finite (Puterman, 1994, Theorem 6.1.1). For more details on the conditions for the existence of the limit above, we refer to Puterman (1994). The optimal *cost-to-go* or *value* function starting from state i is

$$V^*(i) = \min_{\pi \in \Pi} V^{\pi}(i). \tag{3}$$

2.1. The value iteration algorithm

Let $\mathcal{V}$ denote the space of real valued functions and let $V \in \mathcal{V}$ be a function in that space. Define the following mapping:

$$TV(i) = \min_{a \in A(i)} \left[c(i,a) + \alpha \sum_{j \in S} V(j) p_j(i,a) \right]. \tag{4}$$

We assume throughout that the minimum in Eq. (4) exists, which will be the case, for example, if $A(i)$ is finite and $\|V\| < \infty$, e.g., (Puterman, 1994, Lemma 5.6.1). The *value iteration* (VI) algorithm applies mapping (4) to the iterative sequence

$$V_k = TV_{k-1}, \quad k = 1, 2, \ldots \tag{5}$$

given a starting element $V_0 = v_0 \in \mathcal{V}$, where T is an operator on a normed space $(\mathcal{V}, \|\cdot\|)$. Standard dynamic programming results yield that the sequence V_k, $k \geqslant 0$ successively approximates the value function (3), e.g., (Puterman, 1994, Theorem 6.3.1).

It is well known that the VI algorithm converges linearly with the discount rate α, as a consequence of the contraction property of mapping T, which reads (Puterman, 1994):