Stochastics and Statistics

Solving average cost Markov decision processes by means of a two-phase time aggregation algorithm

E.F. Arruda a,⇑, M.D. Fragoso b

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 Center for Systems and Control, National Laboratory for Scientific Computation, Av. Getúlio Vargas, 333, Petrópolis, RJ 25651-075, Brazil

ARTICLE INFO

Article history:
Received 10 August 2012
Accepted 14 August 2014
Available online xxxx

Keywords:
Dynamic programming
Markov decision processes
Embedding
Time aggregation
Stochastic optimal control

1. Introduction

Recent developments in the theory and simulation techniques for Markov decision processes (MDP) (e.g., Busoniu, Ernst, Schutter, & Babuska, 2010; Cao, Ren, Bhatnagar, Fu, & Marcus, 2002; Chang, Fu, Hu, & Marcus, 2007; Leizarowitz & Shwartz, 2008; Powell, 2007) have led to a growing body of literature on MDP modeling for real world problems (e.g., Anderson, Boulanger, Powell, & Scott, 2011; Arruda & do Val, 2008; Pennesi & Paschalidis, 2010; Zhang & Archibald, 2011). Much of this increased interest is due, in part, to the development of powerful techniques to deal with MDPs of very large dimensions, encompassed in a framework known as approximate dynamic programming (ADP) (Bertsekas & Tsitsiklis, 1996; Powell, 2007; Sutton & Barto, 1998).

In the context of the ADP framework, there is a vast literature covering a variety of techniques, such as heuristic search (Hansen & Zilberstein, 2001) and real-time dynamic programming (Barto, Bradtke, & Singh, 1995; Bonet & Geffner, 2003), which make use of asynchronous updates and heuristic search to accelerate convergence, as well as topological value iteration (Dai & Goldsmith, 2007; Dai, Mausam, & Weld, 2009), that processes information related to the graphical features of MDPs to decide the optimal ordering of the value function updates. Asynchronous updates are also exploited in (Akramizadeh, Afshar, Menhaj, & Jafari, 2011; Moore & Atkeson, 1993), while a sequence of increasingly accurate approximate models is used in (Arruda, Ourique, LaCombe, & Almudevar, 2013).

Among the most popular ADP techniques one finds value function approximation (e.g., Arruda, Fragoso, & do Val, 2011; Boyan & Moore, 1995; Li & Littman, 2010), and simulation coupled with state space reduction (e.g., Arruda & do Val, 2008; Cao et al., 2002; Chang et al., 2007). There is a wide range of theory and applications within the ADP framework covering value function approximation techniques, especially for discounted cost MDP problems (see, e.g., Powell, 2012). In particular, the abstract representation of the value function in terms of algebraic decision diagrams (e.g., Hoey, St-aubin, Hu, & Boutilier, 1999; Joshi & Khardon, 2011; St-aubin, Hoey, & Boutilier, 2000) can be efficiently used to solve some large scale discounted MDPs. While much progress has been made and a few promising directions have been devised (e.g., Arruda et al., 2011; Ormoneit & Sen, 2002; Powell, 2007; Tsitsiklis & Van Roy, 1997)), convergence results for general approximation architectures remain to be proved. Moreover, performance bounds for such techniques tend to be very specialized (e.g., Gordon, 1995; Tsitsiklis & Van Roy, 1997; Lin, Hui, Hua-Yong, & Lin-Cheng, 2009).

State space reduction techniques, known as embedding or time aggregation, can be traced back, in the context of control theory, at least to (Zhang & Yu-Chi, 1991), but we shall be particularly interested in the works of (Cao et al., 2002; Chang et al., 2007;
Leizarowitz & Shwartz, 2008). It is well known that a great advantage of time aggregation is that, unlike state aggregation (e.g. Bertsekas, 2012), it preserves the Markov property. As a result, it can be used to produce an equivalent formulation with reduced state space. In that context, Fainberg (1986) studied the construction of embedded MDP models for the total cost criterion, whereas Leizarowitz and Shwartz (2008) investigated embedding techniques for average cost MDPs. An earlier work, (Cao et al., 2002), investigated embedding in a scenario where the control policy within a certain region of the state space is fixed and focused on reducing the computational burden of the solution procedure. This approach was later extended to deal with a continuous time stochastic control problem (Xu & Cao, 2011), and also inspired further work on algorithms for embedded (time aggregated) MDPs e.g., Ren and Krogh (2005), Sun, Zhao, and Luh (2007), Arruda and Fragoso (2011). Similar concepts were applied in the context of discount MDPs (Hauskrecht, Meuleau, Kaelbling, Dean, & Boutilier, 1998), where hierarchical models were employed to decompose the process. A thorough discussion of compact representations for MDPs can be found in (Boutilier, Dean, & Hanks, 1999).

The time aggregation approach, which transforms an MDP into another equivalent MDP with reduced state space, can be of great assistance when one wishes to find approximate solutions in reduced computational time. To accomplish such reduction, one can trade speed for accuracy and specify a priori an outer policy that prescribes a pure control action to each state outside of a prescribed region of interest $F$ of the state space $S$. An appropriately optimized inner policy is then obtained and both outer and inner policies are composed to result in a sub-optimal policy over $S$. This policy minimizes the long term average cost over all control policies that adopt the prescribed outer policy. Note that optimality cannot be guaranteed unless the outer policy is optimal, i.e., it is comprised of optimal control actions for every state in $F^c = S \setminus F$. In particular, optimality can be attained for large scale MDPs with a large number of uncontrollable states, i.e. states for which only a single control action is available (see Cao et al., 2002).

A distinguishing feature of this paper is that, unlike the current literature in time aggregation, it addresses also the problem of iteratively refining the outer policy. The rationale is simple: to apply time aggregation iteratively, but refining the outer policy at each iteration, until the outer policy converges to an optimal outer policy. At this point, the time aggregation approach is able to retrieve an optimal policy for the original MDP, over the entire state space $S$. The proposed outer policy refinement routine can be seen as a policy improvement step of the classical policy iteration algorithm e.g., Bertsekas (2012), which makes use of the value function of the latest policy obtained by time aggregation. A novel contribution of this paper is the way we derive this value function, making use of some new results that are introduced in this paper. Firstly, we prove that the value function obtained by the time aggregation algorithm for each state in the subset $F$ is numerically equal to the value function obtained by a classical policy evaluation algorithm for this same state. We then make use of this result to derive the value function for each state in $F^c$ as the value of a classical stochastic shortest path problem starting from this state to reach any state in the target region $F$.

To sum up, we propose a two-phase time aggregation algorithm to solve MDPs to optimality. The two phases of the algorithm, which are applied successively up to convergence, work as follows: in the first phase, time aggregation is applied for some prescribed outer policy; then in the second phase, a policy improvement step is applied that refines the outer policy. We prove that the proposed algorithm converges monotonically to the optimal policy under general conditions on the structure of the MDP. It is worth pointing out that the proposed approach can be seen as a variation of the classical policy iteration algorithm with a policy search in the subset $F$ at each iteration. The policy search is performed by the time aggregation algorithm, which finds the best possible policy in $F$ given that the policy in $F^c$ is fixed.

This paper is organized as follows. Section 2 presents the studied problem. Section 3 features the time aggregation approach and derives a novel result on the correspondence between the value functions of the embedded MDP and the original MDP, for a fixed control policy. This result is then applied in Section 4 to derive a two phase algorithm for the studied problem. The convergence of the proposed algorithm to the optimal solution is then proved in Section 4.1. Numerical experiments are presented in Section 5 to illustrate the approach, and Section 6 concludes the paper.

2. Preliminaries and the studied problem

Consider a time homogeneous discrete time Markov decision process (MDP) with a finite, possibly very large, state space $S$. Let $A(i) \in \mathbb{N}$ denote the set of feasible control actions at state $i$ and define $A := \{A(i), \ i \in S\}$, and suppose that a function $f : S \times A \to \mathbb{R}$, represents the one-period cost of the process, where $\mathbb{R}$ denotes the set of nonnegative real numbers.

Let $\mathcal{L} : S \to A$ represent a stationary control policy over the state space $S$, and let $\mathcal{L} \in \mathcal{L}$ be the set of all feasible stationary control policies. Under policy $\mathcal{L}$, one selects control action $a = \mathcal{L}(i)$ at each time the controlled process visits state $i \in S$. Following a visit to state $i \in S$, and the application of a control action $a \in A(i)$, the process moves to state $j \in S$ with probability $p_{ij}^a$. Hence, the evolution of the controlled processes under a control policy $\mathcal{L} \in \mathcal{L}$ is governed by a Markov chain $(X_t, \ t \geq 0)$, and the one-period transitions are determined by the transition matrix $P^a = (p_{ij}^a)_{i,j \in S}$. Following Cao et al. (2002), we assume that the controlled process is ergodic under all policies. Assume that the one-period cost function $f$ is a measurable positive real-valued function and let

$$\eta^c = \lim_{N \to \infty} \frac{1}{N} \sum_{k=0}^{N-1} f(X_k, \mathcal{L}(X_k))$$

be the long term average cost of the controlled chain. Because the controlled chain is ergodic, this cost is independent of the initial state. The objective of the decision maker is to find the optimal policy $\mathcal{L}^* \in \mathcal{L}$, which satisfies

$$\eta^{c,*} = \eta^c, \ \forall \mathcal{L} \in \mathcal{L}.$$  

(2)

3. Fixing an outer policy: the time aggregation approach

Now let us select a subset $F \subset S$, and define an outer policy $\mathcal{L}_{out} : F^c \to A$ as the set of control actions prescribed by policy $\mathcal{L}$ for all states outside of $F$, i.e. $\mathcal{L}_{out} = \{(\mathcal{L}(i), i \in F^c)\}$, where $F^c = S \setminus F$ is the complement of $F$. Similarly, an inner policy $\mathcal{L}_{in} : \subset A$, $\mathcal{L}_{in} = \{(\mathcal{L}(i), i \in F)\}$ denotes the control strategy prescribed by policy $\mathcal{L}$ for the subset $F$. Clearly, we have $\mathcal{L} = \mathcal{L}_{in} \cup \mathcal{L}_{out}$. We let $\mathcal{L}_{in}$ and $\mathcal{L}_{out}$ denote the sets of all feasible inner and outer policies, respectively. Fig. 1 illustrates the concepts of inner and outer policies.

Typically the set $F$ is a relatively small subset of $S$. It may be comprised, for example, of the states which are more important from some control standpoint, or the states that are expected to be visited more frequently under some particular class of control policies. For example, in a storage control problem, one would expect the set $F$ to be comprised of the states that are within some desirable vicinity of the zero-stock state.

Now let us select an outer policy $d : F^c \to A$, $d \in \mathcal{L}_{out}$, a priori and define the following problem

Please cite this article in press as: Arruda, E. F., & Fragoso, M. D. Solving average cost Markov decision processes by means of a two-phase time aggregation algorithm. European Journal of Operational Research (2014), http://dx.doi.org/10.1016/j.ejor.2014.08.023
Minimize \( \eta^2 \), \( \mathcal{L} \in \mathbb{L} \)
subject to \( \mathcal{L}_{out} = d \).

The time aggregation approach solves Problem (3), for some outer policy \( d \) prescribed a priori at the decision maker discretion. Naturally, unless \( d \) is the only feasible outer policy, solving Problem (3) involves some loss of performance, and optimality can no longer be guaranteed to the original problem. Indeed, selecting the outer policy \( d \) in such a way as to mitigate the loss of performance depends on the sensibility and prior knowledge of the decision maker.

With respect to any optimal policy \( \mathcal{L}^* \) which solves (2), we define:

\[
\mathcal{L}_{in}^* = \mathcal{L}^*(i), \quad i \in F,
\]

and

\[
\mathcal{L}_{out}^* = \mathcal{L}^*(i), \quad i \in F^*,
\]

for some prescribed subset \( F \in S \). The following Lemma states the somewhat intuitive result that, once the outer policy is chosen to be guaranteed to the original problem. Indeed, selecting the outer policy \( d \) in such a way as to mitigate the loss of performance depends on the sensibility and prior knowledge of the decision maker.

**Theorem 1.** Suppose \( d = \mathcal{L}_{out}^* \), then Problem (3) is equivalent to Problem (2).

**Proof.** Let \( \eta_d \) solve Problem (3). It is not difficult to see that the solution space of (3) is a subset of \( \eta_d \), the solution space of (2). Hence, we conclude that \( \eta_d \geq \eta. \) Moreover, we also know that \( \mathcal{L}^* \) is a feasible solution to (3). This implies \( \eta_d \leq \eta^* \). From the two inequalities, we conclude that \( \eta_d = \eta^* \). \( \square \)

### 3.1. Policy evaluation by time aggregation

Now, let us concentrate on solving Problem (3). Let \( \tau_0 = 0 \) and define \( \tau_i = \min\{t > \tau_{i-1} : X_t \in F\}, i = 1, 2, \ldots \) to represent the times at which the process visits a prescribed subset \( F \) of the state space \( S \). The embedded chain \( \{Y_t := X_{\tau_i}, i \geq 0\} \) maps the sequence of visits of the original MDP to the subset \( F \). As a result, it has state space \( F \) and its properties can be determined by investigating the trajectories between consecutive visits to \( F \). We recall that, in Problem (3), all feasible policies \( \mathcal{L} \) share the same structure \( \mathcal{L}_{out} \in \mathbb{L}_{out} \) within the set \( F^* \).

The time aggregation approach involves dividing the original Markov chain into segments, each segment corresponding to a trajectory between two successive visits to the set \( F \). Hence, the segment corresponding to a given state \( Y_t = X_{\tau_i} \) is \( \{X_{\tau_i}, X_{\tau_{i+1}}, \ldots, X_{\tau_{i+1}}\} \), and the expected cost of a segment starting at state \( i \in F \), under control action \( \mathcal{L}(i) = a \), is

\[
h_f(i, a) = E\left[ \sum_{t=0}^{\tau_{i+1}-1} f(X_t, \mathcal{L}(X_t)) | X_0 = i, \mathcal{L}(i) = a \right]
= E\left[ \sum_{t=0}^{\tau_{i+1}-1} f(X_t, \mathcal{L}(X_t)) | X_0 = i, \mathcal{L}(i) = a \right], \quad i \in F, \mathcal{L} \in \mathbb{L},
\]

where the second equality in the first expression follows from the strong Markov property. Note, from Eq. (6), that the outer policy \( \mathcal{L}_{out} \in \mathbb{L}_{out} \) and the control action \( a \) applied at state \( i \) completely determine the quantity \( h_f(i, a) \). Note that the time aggregated process can be seen as a semi-Markov process, where the duration of each period starting at state \( Y_t \in F \) corresponds to the length of the trajectory \( \{X_{\tau_j}, X_{\tau_{j+1}}, \ldots, X_{\tau_{j,n}}\} \). Hence, in order to solve for the optimal policy, we need to keep track of the expected elapsed time of the trajectories starting at each state \( i \in F \), which can be defined as:

\[
h_s(i, a) = h_f(i, a), \quad \text{for } f(i, a) = 1, \quad \forall j \in S, a \in A(j).
\]

For the embedded process \( \{Y_t, t \geq 0\} \), the transition probability from state \( i \in F \) to state \( j \in F \), under control \( a \in A(i) \) and outer policy \( \mathcal{L}_{out} \in \mathbb{L}_{out} \) is given by \( p_{ij} \). Let \( \tau = \min\{t > 0 : X_t \in F\} \) be the first return time to \( F \) and define \( p_{ij}^\tau (\tau) = P(Y_\tau = j | X_0 = i) \), \( i, j \in F^* \). Then, for any \( i \in F^* \), this quantity can be calculated iteratively as:

\[
p_{ij}^\tau (\tau) = p_{ij}^\tau (\tau) + \sum_{k \in F^*} p_{ik}^{\tau out} (\tau) \cdot p_{kj}^\tau (\tau).
\]

That results in \(|F|\) systems of linear equations, each of them with \(|F^*|\) equations and the same number of unknowns. The time complexity to solve for all the transition probabilities is \( O(|F|^2 |F^*|^2) \). Alternatively, these systems lead to:

\[
p_{ij}^\tau (\tau) = \left( I - p_{ij}^\tau (\tau) \right)^{-1} \cdot p_{ij}^\tau (\tau),
\]

where

\[
p_{ij}^\tau (\tau) = \left\{ \begin{array}{ll} p_{ij}^{\tau out}(i, j) & \text{if } i, j \in F^* \text{,} \\ p_{ij}^{\tau out}(i) & \text{if } i \in F^*, j \in F. \end{array} \right.
\]

As a result, the transition probabilities for the embedded process are given by:

\[
p_{ij}^\tau (\tau) = p_{ij}^{\tau out} + \sum_{k \in F^*} p_{ik}^{\tau out} \cdot p_{kj}^\tau (\tau).
\]

Thus, the complexity of evaluating exactly the transition probabilities for all state-action pairs is \( O(|F| \cdot |A| \cdot |F^*|^2) \). Taking into account the evaluation of the last term in the equality above, we get a complexity of \( O(|F| \cdot (|A| + |F^*|)^2) \). Similar arguments yield a complexity of \( O(|F| \cdot |A| + |F^*|^2) \) for the evaluation of both functionals \( h_s \) and \( h_f \), for every state-action pair in \( F \). However, as in (Cao et al., 2002, Section 6), simulation based techniques can be used to estimate these quantities which are more economical in terms of computational cost for problems with very large state spaces.

Let now \( \bar{n} = \bar{n}(i, j, F) \) be the set of steady state probabilities of the embedded process \( \{Y_t, t \geq 0\} \) under a control policy \( \mathcal{L} \), and define:

\[
\bar{n}^\tau = \sum_{i,j \in F^*} \pi(i, j, \mathcal{L}(i)), \quad \mathcal{L} \in \mathbb{L}, \quad \mathcal{L}^*.
\]

It is proved in Cao et al. (2002) that, for each control policy \( \mathcal{L} \), the long term average cost of the original MDP is numerically equal to

\[
\eta^\tau = \lim_{M \to \infty} \frac{1}{M} \sum_{m=0}^{M-1} \bar{n}^\tau (\mathcal{L}(Y_m), \mathcal{L}(Y_m)) = \sum_{i,j \in F^*} \pi(i, j, \mathcal{L}(i)) \cdot h_f(i, \mathcal{L}(i)),
\]

or

\[
= \frac{\eta^\tau_f}{\eta^\tau}, \quad \mathcal{L} \in \mathbb{L},
\]

Please cite this article in press as: Arruda, E. F., & Fragoso, M. D. Solving average cost Markov decision processes by means of a two-phase time aggregation algorithm. European Journal of Operational Research (2014), http://dx.doi.org/10.1016/j.ejor.2014.08.023
where $\pi$ is the row vector of steady state probabilities of the embedded process \( \{Y_t, t \geq 0\} \) and \( H^*_f \) is a column vector of the individual values \( h_f(i, L(i)), i \in F \).

Now, let us fix \( L_{out} = d, d \in L_{out} \) and solve Problem (3). To solve this problem, one can apply the Incremental Value Iteration (IVI) algorithm in (Sun et al., 2007, Algorithm 2, page 2180). Loosely speaking, the IVI algorithm solves a series of parametric MDPs with a modified cost function $r_t = (h_t - \delta_t h_t)$, refining the parameter $\delta_t$ until the average cost of the modified MDP converges to zero. They demonstrate that when that happens, the parameter $\delta_t$ is numerically equal to the average cost of the original problem (Sun et al., 2007, Corollary 1, Page 2179). Each parametric MDP is solved by means of value iteration, hence the complexity of each iteration of the IVI algorithm is $O(|F|^2)$. This process results in a pair \((V_\pi, \eta_\pi)\), which satisfies the Poisson Equation in the time aggregated domain (Sun et al., 2007):

\[
h_f(i, a) - \eta_\pi h_f(i, a) + \sum_{j \in S} \pi_{ijk} V_\pi(j) = V_\pi(d_j), \quad i \in F, \quad a = L'(i),
\]

(10)

where \( V_\pi: F \to \mathbb{R} \) is a real valued function in \( F \) and \( \eta_\pi = \eta_{OF} \) is a scalar that also solves Eq. (1) for policy \( L' \). The optimal policy for Problem (3), denoted by \( L_{opt} = \frac{\arg \min}{a \in A} \), can be recovered by using the expression below:

\[
L_{opt}(i) = \arg \min_{a \in A} \left[ h_f(i, a) - \eta_\pi h_f(i, a) + \sum_{j \in S} \pi_{ijk} V_\pi(j) \right], \quad i \in F.
\]

(11)

Note that Eq. (10) in this paper corresponds to Eq. (24) in Step 2 of the IVI algorithm in (Sun et al., 2007, page 2180), whereas our Eq. (11) corresponds to the expression in Step 5 of the referred algorithm. In both expressions, \( \eta_\pi \) is equivalent to \( \delta_t \) and \( V_\pi(d_j) \) corresponds to \( g_m(j) \).

In the next subsection we study the properties of the value function for a fixed control policy \( L \), in particular with respect to a prescribed stopping time \( \tau \). By prescribing \( \tau \) as the minimum time to reach a subset \( F \subseteq S \) of the state space, Lemma 3 derives the value function in the complementary subset \( F^C \), making use of both the long term average cost \( \eta^F \) of policy \( L \), and the value function in the prescribed subset \( F \).

### 3.2. Classical policy evaluation: properties of the value function

Let us now focus on the classical solution of average cost MDPs. Let \( V \) be the space of real valued functions in \( S \) and let \( V : S \to \mathbb{R} \) be an element of this space. It is well known that the long term average cost of any given policy \( L \in \mathcal{L} \) can be found by solving the Poisson Equation e.g., Puterman (1994, Corollary 8.2.7, page 344), which reads:

\[
f(i, a) + \sum_{j \in S} \pi_{ijk} V^L(j) = V^L(i) + \eta^L, \quad i \in S, \quad a = L(i).
\]

(12)

Furthermore, it is also well known that, for any policy \( L^* \in \mathcal{L} \) which solves (2), it holds that e.g., Puterman (1994, Eq. (8.4.2), page 354):

\[
\min_{a \in A} \left[ f(i, a) + \sum_{j \in S} \pi_{ijk} V^L(j) \right] = V^L(i) + \eta^L, \quad \forall i \in S.
\]

(13)

Lemma 1 derives a multi-step evaluation of the value function, which is applied to a specific stopping time in Corollary 2, which will be applied to derive the value function at a given state as a function of a stopping time. Lemma 3 applies this result to define a specific shortest path evaluation of the value function in a specific region of the state space.

**Lemma 1.** Suppose $\eta^L$ solves (12) for some $L \in \mathcal{L}$. Then, for all positive integers $n$:

\[
E \left[ \left\{ \sum_{k=0}^{n-1} f(X_k, \mathcal{L}(X_k)) + V^C(X_k) \right\} \mid X_0 = i \right] = V^C(i) + \eta^L, \quad i \in S, L \in \mathcal{L}.
\]

(14)

**Proof.** It is not difficult to see that, for policy $L \in \mathcal{L}$,

\[
E[f(X_0, \mathcal{L}(X_0)) + V^C(X_1) \mid X_0 = i] = f(i, a) + \sum_{j \in S} \pi_{ijk} V^C(j), \quad i \in S;
\]

\[
\therefore E[f(X_0, \mathcal{L}(X_0)) + V^C(X_1) \mid X_0 = i] = V^C(i) + \eta^L, \quad i \in S,
\]

where $a = \mathcal{L}(i)$, and the last equality follows from (12), which holds by hypothesis. Hence, it is proved that (14) holds for $n = 1$.

Now suppose it also holds for some positive integer $(n - 1)$, i.e.

\[
E \left[ \left\{ \sum_{k=0}^{n-2} f(X_k, \mathcal{L}(X_k)) + V^C(X_{n-1}) \right\} \mid X_0 = i \right] = V^C(i) + (n - 1)\eta^L, \quad i \in S.
\]

(15)

Applying (12) and the Markov property, one can easily see that:

\[
V^C(X_{n-1}) = \sum_{j \in S} \left[ f(j, \mathcal{L}(j)) + \sum_{k \in S} \pi_{jk} V^C(k) - \eta^L \right] E_{[X_{n-1} = j]},
\]

(16)

where $E_{[X_{n-1} = j]}$ is the indicator function of state $j$, which assumes value 1 (one) whenever $j$ holds true and is zero otherwise, and $a = \mathcal{L}(j)$. Substituting (16) in (15), we obtain

\[
E \left[ \left\{ \sum_{k=0}^{n-1} f(X_k, \mathcal{L}(X_k)) + V^C(X_k) \right\} \mid X_0 = i \right] = V^C(i) + n\eta^L, \quad i \in S,
\]

and that completes the proof. □

The result below follows directly from Lemma 1.

**Corollary 2.** For any stopping time $\tau \geq 0$ and any policy $L \in \mathcal{L}$, we have

\[
E \left[ \left\{ \sum_{k=0}^{\tau-1} f(X_k, \mathcal{L}(X_k)) + V^C(X_k) \right\} \mid X_0 = i \right] = V^C(i) + \eta^L E[\tau], \quad i \in S;
\]

(17)

\[
V^C(i) = E \left[ \left\{ \sum_{k=0}^{\tau-1} f(X_k, \mathcal{L}(X_k)) - \eta^L + V^C(X_k) \right\} \mid X_0 = i \right], \quad i \in S.
\]

(18)

In the remainder of this Section, and inspired by the results in Bertsekas (1998), we consider an associated stochastic shortest path (SSP) problem whose domain is the complementary region $F^C$ of the state space, where $F$ is a subset of interest. In Lemma 3 we show how the solution of this problem can be used to derive the value function in $F^C$ as a function of the value function in $F$, obtained by time aggregation in Section 3.3. Hence, the proposed SSP will be used in conjunction with time aggregation to derive the value function in $F^C$, which will be employed in a policy improvement step in the two-phase algorithm, introduced in Section 4. The policy improvement step will lead to an improved control policy in $F$. This improved policy will be then fed back to the time aggregation step, thus resulting in an improved overall solution to the original problem. This process will be repeated in the two-phase algorithm, detailed in Section 4, until convergence is attained.

We now consider an associated stochastic shortest path problem to visit $F$ for the first time from any state $i \in F^C$. This is obtained by leaving unchanged all the transition probabilities $\pi_{ijk}$, $i \in F^C, j \in S, a \in A(i)$, and by making each state $m \in F$ a terminal state, for some $F \subseteq S$, whose complement is defined as $F^C = S \setminus F$. Once the system reaches a terminal states, it incurs a termination cost and goes to an artificial target state, labeled $t$, which
is never left afterwards. Hence, for a given policy \( L \in \mathbb{L} \), we can define the stochastic shortest path probability matrix \( P_{SSP}^L \) as:
\[
P_{SSP}^L = \begin{bmatrix} P_{i,j}^{L}, P_{i,j}^F \end{bmatrix},
\]
where \( P_{i,j}^{L}, P_{i,j}^F = \{ P_{i,j}^{G(i)}, \ i,j \in F \} \). For each state \( j \in F \), we define an instantaneous cost \( g(i,a) = f(i,a) - \lambda \), where \( \lambda \) is a scalar parameter. We also define a terminal cost \( T(m) \in \mathbb{R} \) for each terminal state \( m \in F \). At the artificial target state, no cost is incurred.

Hence, the cost function for a policy \( L \in \mathbb{L} \) is:
\[
g(i,L(i)) = \begin{cases} f(i,L(i)) - \lambda, & \text{for } i \in F \\ T(i), & \text{for } i \in F \\ 0, & \text{for } i = ts \end{cases},
\]

Following Bertsekas (1998) we call this problem the \( \lambda \)–SSP. We note that the value of \( \lambda \) and the termination cost \( T(m) \) will be appropriately defined in such a way as to make the value function of the prescribed \( \lambda \)–SSP coincide with that of the original problem for all states \( i \in F \), as detailed in Lemma 3.

Note that our definition of a \( \lambda \)–SSP problem is slightly different from that of Bertsekas (1998) in that we define a target region \( F \subset S \), instead of a single target state. Note also that, from a control standpoint, it suffices to define an outer policy \( L_{out} : F' \rightarrow A \), since the \( p_{out}^L = 1 \) \( \forall i \in F \). For more details on stochastic shortest paths, we refer to Bertsekas (2012).

Let \( h_{out,i}(j) \) be the value function of a feasible stationary policy \( L_{out} : F' \rightarrow A \) for this problem, starting from state \( i \in F \). That is
\[
h_{out,i}(j) = E\left[ \sum_{k=0}^\infty g(i_k) + T(i_k) \bigg| X_0 = i \right], \quad i \in F,
\]
with \( \tau = \min\{t > 0 : X_t \in F \} \). The finiteness of \( \tau \) is guaranteed under the initial hypothesis that all policies are ergodic (Cao et al., 2002; Leizarowicz & Shwartz, 2008). Making use of one-step transition probabilities, this value can also be expressed as:
\[
h_{out,i}(j) = f(i,L(i)) - \lambda + \sum_{j \in F} p_{i,j}^{out} \cdot T(j) + \sum_{k \in F} p_{i,k}^{out} \cdot h_{out,k}(j), \quad i \in F.
\]

Hence, the exact calculation of the function \( h_{out,i} : F' \rightarrow \mathbb{R} \) can be made by solving a system of \( |F'| \) equations and the same number of unknowns, directly or iteratively. The complexity of such a task is \( O(|F'|^2) \). However, as in [Cao et al., 2002, Section 6], simulation based techniques can be used to estimate these quantities which are more economical in terms of computational cost for problems with very large state spaces.

Lemma 3. Consider a \( \lambda \)-SSP problem with \( \lambda = \eta^2 \) and \( T(m) = V^C(m) \), \( \forall m \in F \), and \( L_{out}(i) = L(i) \), \( \forall i \in F' \), for some \( L \in \mathbb{L} \). Then
\[
h_{out,i}(j) = V^C(j), \quad \forall i \in F.
\]

Proof. To prove the result, it suffices to substitute \( \lambda = \eta^2 \) and \( T(m) = V^C(m) \), \( \forall m \in F \) in (19) and compare it to (18). \( \square \)

3.3. Policy evaluation in the original MDP

From the discussion in Section 3.1, it follows that solving Problem 3 results in a control policy \( L_d = L_{out}^d \cup d \) in the original MDP described in Section 2. Hence, the pair \( (V^d, \eta_d) \) satisfies Eq. (12), since \( V^d : S \rightarrow \mathbb{R} \) is a value function of a fixed control policy in the original problem and \( \eta_d \) is its long term average cost.

The following theorem establishes the correspondence between the time aggregated value function \( V_d^t : F \rightarrow \mathbb{R} \) in (10) and \( V^d \), which is the value function of following policy \( L_d^t \) in the original MDP.

Theorem 2. Let \( (\overline{V}_d, \eta_d) \) solve Eq. (10) for some \( F \in S \) and assume \( L_d^t : F \rightarrow A \) satisfies Eq. (11). Then, letting \( L_d^t = L_{out}^d \cup d \) denote the corresponding control policy in the original MDP, it follows that:
\[
V_d^t(i) = \overline{V}_d(i), \quad \forall i \in F,
\]
and \( \eta^d = \eta_d \), where the pair \( (V^d, \eta_d) \) solves (12).

Proof. That \( \eta^d = \eta_d \) is already established in Eq. (10). Now let \( \tau = \min\{t > 0 : X_t \in F \} \). Then, considering that \( p_{out}^d = p(X_t = j|X_0 = i, L(i) = a, \forall j \in F \) (18) applied to \( L_d^t \) becomes:
\[
V_d^t(i) = E\left[ \sum_{k=0}^\infty f(i_k) + T(i_k) \bigg| X_0 = i \right], \quad a = L_d^t(i),
\]
V_d^t(i) = h_f(i,a) - \eta_d h_t(i,a) + \sum_{j \in F} p_{i,j}^d V_d^t(j) + T(i), \quad i \in F, \quad a = L_d^t(i),
where the last equality is obtained by applying the definitions in (6) and (7), and by substituting \( \eta_d \) for \( \eta^d \). And the proof is concluded by noting that the last equality corresponds to Eq. (10). \( \square \)

4. A two phase algorithm for finding the value function

In this section, we make use of Theorem 2 and Lemma 3 to determine the complete solution of the Poisson Equation for the policy \( L_d^t \) obtained via time aggregation. Theorem 2 implies that, by solving the time aggregated problem, we obtain the partial solution of the Poisson Equation for the subset \( F \subset S \). Then, by solving a \( \eta^d \)-SSP as defined in (19), and making use of Lemma 3, we are able to find the partial solution of the Poisson Equation for the subset \( F \).

Then, in possession of the whole solution of the Poisson equation for the current policy, a policy improvement step (Puterman, 1994, Chapter 8) can be executed within the original MDP to find an improved outer policy \( d \). The whole procedure can then be repeated for the newly found outer policy and so on, until an optimal solution is found. The proposed algorithm is presented below:

Algorithm 1. Two Phase Time Aggregation

1. Choose \( F \), select \( L_{out} = d_0, d_0 \in \mathbb{L} \) and make \( k = 0 \).
2. Solve Problem 3 for \( (\overline{V}_d, \eta_d) \), with \( d = d_k \). Make \( \eta_{d_k} = \eta_d \).
3. Find \( L_{out}^d 
4. Make \( L_d = L_{out}^d \cup d_k \) and \( V_d^t(i) = \overline{V}_d(i), \forall i \in F \), in the original MDP.
5. Solve a \( \lambda \) – SSP-Eq. (19) for \( \lambda = \eta_d \) and \( T(m) = V^C(m) \), \( \forall m \in F \). Use Lemma 3 to obtain \( V^t_{\lambda} \), \( \forall i \in F \).
6. Apply a policy improvement step for all \( i \in F \): \( d_{k+1}(i) = \arg\min_{a \in A} \left[ f(i,a) + \sum_{j \in F} p_{i,j}^d V^t_{\lambda}(j) \right] \), \( i \in F \).
7. If \( d_{k+1}(i) = d_k(i) \), STOP. Otherwise, go to Step 8.
8. \( k \rightarrow k + 1 \); return to Step 2.

Please cite this article in press as: Arruda, E. F., & Fragoso, M. D. Solving average cost Markov decision processes by means of a two-phase time aggregation algorithm. European Journal of Operational Research (2014), http://dx.doi.org/10.1016/j.ejor.2014.08.023
Step 1 of Algorithm 1 selects a partial control policy in $F$. This partial policy is used in Step 2, where the algorithm solves a time aggregation problem. The optimal policy for this problem, a partial policy in $F$, is recovered in Step 3. Both partial policies are composed to result in a control policy for the original MDP in Step 4. Step 5 solves a $\lambda$-SSP and applies Lemma 3 to recover the value function for the current policy, within the subset $F$. The complete value function is then used in a policy improvement step in Step 5, where a new partial control policy in $F$ is obtained, and the process restarts in Step 2 until convergence is attained.

The computational issues of Steps 2 and 5 are discussed in Sections 3.1 and 3.2, respectively. It is worth mentioning, however, that one can directly solve (3) (Step 2 of Algorithm 1) and obtain the associated value function (Step 5 of Algorithm 1) at the same time by solving a modified MDP in the original state space, in which the feasible actions for all states in $F$ are limited to those prescribed by the outer policy $d_0$, with no restriction to the control actions in $F$. In that sense, the time to solve this modified MDP in the original state space can be seen as an upper bound to the total time of executing Steps 2 and 5 of Algorithm 1. Thus, solving Problem (3) by time aggregation is recommended only when the sampling strategy makes it faster than solving an ordinary MDP with optimization in $F$, or whenever the size of the state space renders the latter algorithm intractable. For the sake of simplicity, we chose to solve the modified MDP described above in our numerical runs in Section 5.

We note that the proposed algorithm is a form of policy iteration with optimization in $F$, in which the outer states in $F$ are updated simply based on a policy improvement step, while at the inner states in $F$ an optimization is realized by means of time aggregation to find the best possible inner policy with respect to the outer policy. As the numerical experiments in Section 5 illustrate, this optimization routine can significantly accelerate the convergence of the algorithm with respect to the traditional policy iteration algorithm. The convergence of Algorithm 1 to the optimal policy is proved in the next subsection.

4.1. Convergence of the two phase time aggregation algorithm

Lemma 4. Let $d_{k+1} \in L_{out}$ be the outer policy selected in Step 6 of Algorithm 1, and let $\overline{\pi}_{d_{k+1}} = \pi_{d_{k+1}}^{F} \cup d_{k+1}$. Then, $\eta_{\pi_{d_{k+1}}} \leq \eta_{d_{k}}$.

Proof. The arguments in this proof follow closely those in (Puterman, 1994, Proposition 8.6.1, page 379). Note that $V^{\pi_{d_{k}}} : \mathcal{I} \rightarrow \mathbb{R}$ is known, as it was previously obtained in Steps 4 and 5 of Algorithm 1. The same applies to $\eta_{d_{k}}$, which was found in Step 2 of the same algorithm. We recall from Section 3.2, that these quantities satisfy Eq. (12), the Poisson Equation in the original domain. Rearranging this expression and converting it to matrix notation, one obtains:

$$
\eta_{\pi_{d_{k}}} \cdot e = \overline{f} \pi_{d_{k}} + P^{d_{k}} V^{\pi_{d_{k}}} - V^{p_{d_{k}}},
$$

$$
\eta_{d_{k}} \cdot e = f \pi_{d_{k}} + (P^{d_{k}} - I) V^{p_{d_{k}}},
$$

where $e$ is a vector of unitary components with cardinality $|S|$. Let $\pi^{\overline{f}}_{d_{k+1}}$ be a row vector that represents the invariant distribution associated with applying policy $\overline{\pi}_{d_{k+1}}$ in the original MDP, which satisfies

$$
\pi^{\overline{f}}_{d_{k+1}} (P^{d_{k+1}} - I) = 0, \quad \pi^{\overline{f}}_{d_{k+1}} \cdot e = 1.
$$

Then, standard MDP results yield that:

$$
\eta_{\pi_{d_{k+1}}} = \pi^{\overline{f}}_{d_{k+1}} \overline{f}.
$$

Adding and subtracting $\eta_{\pi_{d_{k}}}$ and making use of (22), we can write:

$$
\eta_{\pi_{d_{k+1}}} = \eta_{\pi_{d_{k}}} + \pi^{\overline{f}}_{d_{k+1}} [f^{\pi_{d_{k}}} - \eta_{\pi_{d_{k}}} e] + \pi^{\overline{f}}_{d_{k+1}} (P^{d_{k+1}} - I) V^{p_{d_{k+1}}} = \eta_{\pi_{d_{k}}} + \pi^{\overline{f}}_{d_{k+1}} [f^{\pi_{d_{k}}} - \eta_{\pi_{d_{k}}} e + (P^{d_{k+1}} - I) V^{p_{d_{k+1}}}].
$$

Note that Eq. (21) implies that the term in brackets is zero if $L_{d_{k+1}} = L_{d_{k}}$. Hence, since $L_{d_{k}}$ is chosen to minimize the quantity in brackets in Step 6 of Algorithm 1, this quantity is upper bounded by zero (it is less than or equal to zero). Thus, we have

$$
\eta_{\pi_{d_{k}}} \leq \eta_{\pi_{d_{k+1}}}.
$$

Theorem 3. Let $L_{d_{k}} \in \mathcal{L}$ be the policy found in the Step 4 of the $k$-th iteration of Algorithm 1. Then, for any successive iterations of Algorithm 1 it holds that,

$$
\eta_{\pi_{d_{k}}} \leq \eta_{d_{k+1}},
$$

Furthermore, note that $\eta_{\pi_{d_{k}}} = \eta_{d_{k+1}}$, where $\eta_{d_{k+1}}$ is determined in Step 2 of Algorithm 1. Hence, we have

$$
\eta_{\pi_{d_{k}}} \leq \eta_{d_{k+1}}.
$$

The expression above holds true because $\overline{L}_{d_{k}}$ is a feasible solution for the optimization problem solved in Step 2 of Algorithm 1. Therefore, from Eq. (23) and (24), it follows that

$$
\eta_{\pi_{d_{k}}} \leq \eta_{d_{k+1}},
$$

and that ends the proof.

From Theorem 3, one can see that each new policy determined by Algorithm 1 is better than its previous counterpart in terms of average cost. That means that the algorithm converges in a finite number of iterations because, as the state and action spaces are finite, there are only a finite number of policies to choose from, and the average cost cannot be improved forever. It remains to be shown that the policy to which Algorithm 1 converges is indeed an optimal policy, and this result is proved in the next theorem.

Theorem 4. Let $d_{k+1} = d_{k}$, for some integer $k \geq 1$ in Algorithm 1. Then, $L_{d_{k}} = L^*$, where $L_{d_{k}}$ is the policy found in Step 4 of this algorithm and $L^*$ is a solution of Problem 2.

Proof. Suppose, for the sake of contradiction, that $d_{k} \neq L_{out}$, where $L_{out}$ was defined in (5). Then, Eq. (13) is not satisfied for at least one state $i \in F$, which implies that $d_{k+1}(i) \neq d_{k}(i)$ for some $i \in F$ in Step 6 of Algorithm 1. This, in turn, implies that $d_{k+1} \neq d_{k}$, which contradicts the initial hypothesis. Hence, it follows that $d_{k} = L_{out}$.

Hence, by Theorem 1 it follows that the output of Step 4 of Algorithm 1 must be a solution to Problem 2, and that concludes the proof.

We conclude this section with some remarks on the numerical efficiency of the two-phase method.

Remark 1. As mentioned in Section 3.1, the complexity of Step 2 of Algorithm 1 is of order $O(|F| \cdot |A| + |F|^3) + O(|F|^3)$, while the complexity of Step 5 is of order $O(|F|^3)$, as outlined in Section 3.2. Hence, the complexity of one iteration of the two-phase algorithm is of order $O(|F| \cdot |A| + |F|^3) + O(|F|^3) + O(|F|^3)$, while...
the complexity of one iteration of a standard policy improvement is of order $O(|S|^3)$. For the pure control policy, which the numerical example of the paper belongs to, $|A| \approx |S|$. For applications such that $|F| \ll |F^c|$ and thus $|F^c| \approx |S|$, $O(|F^c| \cdot |A| + |F|^3) + O(|F|^3) + O(|F^c|^3) \approx O(|F|^3) \approx O(|S|^3)$; for applications such that $|F| = |F^c| = c$ and $|S| = 2c$, $O(|F| \cdot |A| + |F|^3) + O(|F|^3) + O(|F^c|^3) \approx 3c^3 \approx O(|S|^3)$. In all these applications, the complexity of an iteration of two-phase algorithm is similar to that of a standard policy improvement.

Remark 2. The superiority of the proposed method over policy iteration comes from the fact that it obtains better bounds on the value function and on the long term average cost at each iteration, thus making a better usage of the computational resources. The improvement comes from the optimization in Step 2 of Algorithm 1, which allows a search for the optimal policy within the subset $F$, given a prescribed policy in $F^c$. The region $F$ can be selected to be comprised of the most attractive states in $S$ with respect to the cost function $c : S \to \mathbb{R}$, which tend to contribute more to the long term average cost. Hence, an optimization in $F$, which can have little impact in the overall computation if $|F| \ll |S|$, can lead to substantial gains over a single policy improvement step in this region, thus accelerating convergence and leading to good solutions in a reduced time. This effect is illustrated in Section 5.

5. Numerical experiments

To illustrate the proposed approach, we solve a production and inventory problem with 3 classes of customers and a single machine, which can produce one product at a time and can alternate between products without any significant setup time. The demands for each class of customers is Poisson and arrives at rates $3.2$ and $1$, respectively, while the production time follows an exponential distribution with rate $8$. We also assume that each demand order is comprised of a single item. The stock/deficit cost is given by:

$$c(x) = |x_1| + 2|x_2| + 3|x_3|.$$ 

Thus, the stock/deficit costs are ordered in the inverse order of the demand rates, with the most demanded items having the smallest stock/deficit cost. Whenever a new demand arrives or the production of a new product is finished, the decision maker decides whether to continue the production for the same class of customers or to produce for another class of customers, or alternatively, to halt production. If the production is halted at the time a decision is made, the decision maker decides whether to keep the production halted or to start the production for one of the classes of customers. We also note that the production facility can continue production even if there is no customer waiting.

The maximal allowed backlog for each product is of 100 units, with no demand being accepted when this backlog level is reached and the maximal stock level allowed for each product is 25 units. Hence, the stock level of each product is an integer variable in the range $\{-100, \ldots, 25\}$. Hence, the state space $S$ is comprised of $126^3 \approx 2 \times 10^8$ elements, each corresponding to a possible stock/deficit combination for the three products. The objective is to find the policy $\pi$ which minimizes the average cost and satisfies expression (2).

The proposed problem was solved up to a tolerance of $10^{-5}$ in a Intel Core i5, 2.8 GHz, 4 GB RAM Computer, running Windows 7. We compared the results obtained by Algorithm 1 with those obtained by a standard policy iteration algorithm (see, e.g., Puterman (1994)). Both algorithms yielded a long term average stock/deficit cost of 3.4095 monetary units, and their convergence is depicted below. For the proposed algorithm, we pick the region $F$ as the set of states whose stock/deficit levels are in the interval $[-10, 9]$, i.e.,

$$F := \{x \in S : -10 \leq x_1 \leq 9, -10 \leq x_2 \leq 9, -10 \leq x_3 \leq 9\}.$$ 

Both algorithms were initialized with the same ad-hoc policy, which prescribes the production of the product with the lowest stock level, whenever this level is up to a maximum of 10 units. Otherwise, the production is halted. Whenever the lowest stock level is common to two or more products, we choose to produce that with the highest stock/deficit holding cost.

Fig. 2 shows the average cost evolution versus cumulative computation time for both the standard policy iteration algorithm and the proposed algorithm, with each bar corresponding to a single iteration. Note that the decrease in the average cost for the proposed 2-phase algorithm is much steeper, with the algorithm converging in 5 iterations. The policy iteration algorithm, on the other hand, takes 13 iterations to converge, with a less incremental decrease in the value function at each iteration. It is worth pointing out that the immediate decrease in the value function at the first iteration is due to the optimization in the set $F$ that is performed at Step 2 of Algorithm 1. It may be argued that the region $F$ is more important from a control standpoint, once one would expect a properly optimized policy to spend most of the time visiting low stock/deficit states. Hence, it may be expected that the actions taken at those states impact more in the long term average cost, and that is corroborated by the experimental results.

Comparing the results in Fig. 2 in terms of the cumulative computation time, one notes that the proposed approach is also significantly superior in terms of total elapsed time. In fact, the computation time is reduced from around 1422 min to around 260 min, which means that the proposed algorithm converges in about 18.3% of the total policy iteration convergence time. The experimental results are summarized in Table 1 below. One can see that, in this case, optimizing within an attractive region of the state space in terms of instantaneous cost function leads to the vicinity of the optimal policy in just two iterations, while it takes 12 iterations for the standard policy iteration to achieve similar performance. As pointed out earlier, the states in $F$ tend to be visited more often in the long term, thus contributing more to the long term average cost. As a result, an optimization of the control policy in $F$ tends to lead to substantial gains in the long term average cost, as the experiments demonstrate.

5.1. Evaluating the effect of the subset $F$

We saw in the last section that the use of the proposed algorithm can significantly enhance the performance with respect to standard policy iteration for a given selected subset $F$. In this subsection we strive to evaluate the effect of the set $F$ on the performance of Algorithm 1. To accomplish that we set the region $F$ as the set of states whose stock/deficit levels are in the interval $I = [I_1, I_2]$, i.e.,

$$F := \{x \in S : I_1 \leq x_1 \leq I_2, I_1 \leq x_2 \leq I_2, I_1 \leq x_3 \leq I_2\}.$$ 

![Fig. 2. Time versus average cost for TA and PI.](image_url)
Table 1

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Proposed algorithm</th>
<th>Policy iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Average cost</td>
<td>Cumulative time</td>
</tr>
<tr>
<td>1</td>
<td>3.4887</td>
<td>24.15</td>
</tr>
<tr>
<td>2</td>
<td>3.4137</td>
<td>63.35</td>
</tr>
<tr>
<td>3</td>
<td>3.4126</td>
<td>116.28</td>
</tr>
<tr>
<td>4</td>
<td>3.4095</td>
<td>182.25</td>
</tr>
<tr>
<td>5</td>
<td>3.4095</td>
<td>259.52</td>
</tr>
<tr>
<td>6</td>
<td>22.7465</td>
<td>744.8</td>
</tr>
<tr>
<td>7</td>
<td>13.3264</td>
<td>482.9</td>
</tr>
<tr>
<td>8</td>
<td>11.6875</td>
<td>606.6</td>
</tr>
<tr>
<td>9</td>
<td>6.3608</td>
<td>744.8</td>
</tr>
<tr>
<td>10</td>
<td>5.8927</td>
<td>894.9</td>
</tr>
<tr>
<td>11</td>
<td>3.5407</td>
<td>1059.1</td>
</tr>
<tr>
<td>12</td>
<td>3.41729</td>
<td>1235.4</td>
</tr>
<tr>
<td>13</td>
<td>3.4095</td>
<td>1422.1</td>
</tr>
</tbody>
</table>

and varied the interval \(I\) from \([-8, 10]\) to \([6, 25]\), adding up a single item in both \(I_1\) and \(I_2\) at each experiment. We note that the cardinality of the set \(F\) is kept constant throughout the experiments, as the intention is to evaluate the effect of the choice of subsets \(F\) of the same cardinality, but with different locations within the state space \(S\).

The results of the experiments showed that as we increased the values of \(I_1\) and \(I_2\), the average cost at the first iteration consistently deteriorated, but such a deterioration did not affect the performance, for the algorithm continued to converge in 5 iterations, with no perceivable change in the computation time. However, as we reached the interval \(I = [4, 23]\), the algorithm took six iterations to converge and required about 20% more computation time than the original interval \(I = [-10, 9]\) that was used in the experiments of Section 5. From this interval \(I = [4, 23]\) to the last interval \(I = [6, 25]\), we note increases both in the total iteration count and also in the overall computation time. The experimental results for selected intervals are displayed in Fig. 3.

To avoid showing many similar results, we selected six intervals to display in Fig. 3. To facilitate the comparison, the cumulative computation time is normalized with respect to that obtained in the experiment of Section 5. Note that, although the average cost at the first iteration varies significantly for intervals \([-9, 10]\) and \([3, 22]\), the algorithm manages to maintain its performance both in terms of iteration count and computation time. For the remaining intervals, however, the changes are more significant: interval \(I = [4, 23]\) requires six iterations and about 120% of the computation time for the original \(F\), while interval \(I = [5, 24]\) requires 7 iterations and 140% of the same reference computation time. The worst performance comes for interval \([6, 25]\), that requires 9 iterations and around 165% of the reference computation time.

The results suggest that, in this case, the algorithm seems to be relatively robust to the choice of \(F\), its performance varying slowly around the set \(F\) suggested in the last section. That is a very interesting result, in that it shows that small variations in the set \(F\) tend not to compromise the performance of the algorithm. In the experiments we saw that the performance deteriorates once we get too far from the zero-stock region. That seems to be, in this case, because the low stock/deficit states tend to be more attractive from a control standpoint, once their costs are the most attractive. Hence, they tend to be the most visited if the system is properly managed, thus contributing more to the overall average cost. Selecting states close to this region ensures, in the present example, that the optimization in Step 2 of Algorithm 1 has the potential to result in substantial gain in terms of the long term performance of the system, as demonstrated by the experimental results. Finally, it is worth mentioning here that it should be necessary to go more deeply on the algorithm robustness issue regarding the choice of \(F\), since the conclusion here was drawn from just one example. This also applies to the claim here regarding the low stock/deficit region as the most attractive region from a control standpoint. Further foray on these issues is certainly necessary.

Considering the discussion in the preceding paragraph, one can infer that the performance will tend to that of the standard policy iteration algorithm for a choice of \(F\) too far the zero stock level. To demonstrate this effect we set up a new experiment where Algorithm 2 is run with \(F := \{x \in S: -100 \leq x_1 \leq -81, -100 \leq x_2 \leq -81, -100 \leq x_3 \leq -81\}\). The computation time is normalized with respect to that of the standard policy iteration algorithm, and the results are displayed in Fig. 4.

As expected, the performance of the proposed algorithm is similar to that of standard policy iteration. That happens because we chose to compose the region \(F\) a set of states which are very far from a control standpoint: large cost states which tend to be seldom visited for any reasonably good production/inventory policy. As a result, an optimization in this region does not result in an effective improvement in performance, as these states tend to contribute very little to the long term average cost. The results in Fig. 4 are consistent with this evaluation.

6. Concluding remarks and future research directions

This paper proposes a two phase time aggregation based algorithm for solving average cost Markov decision processes. The algorithm combines a time aggregation step within a region of interest \(F\) and a stochastic shortest path solution procedure
outside of this region. The first step determines a sub-optimal control in \( F \) and the value function corresponding to the best solution, given a prescribed control policy in \( F \). Such a value function, used as termination cost of the stochastic shortest path problem, allows one to find the value function in \( F \), which in turns permits the application of a classical policy improvement step that leads to the improvement of the control policy in \( F \). Both steps are alternated until the algorithm converges. In addition, the convergence of the proposed algorithm to the optimal solution is proved and a numerical example illustrates the proposed approach.

The numerical experiment suggests that the optimization step in the subset \( F \) of the state space, prescribed by the proposed algorithm, can significantly reduce the convergence time both in terms of iteration count and total computation time with respect to standard policy iteration.

Future research directions include the application of the proposed approach to very large scale problems, possibly with a partial solution of the stochastic shortest path problem, constrained to a specific target region of the set \( F \). Another possible line of research involves applying reinforcement learning techniques to approximate the value function in the complementary subset \( F^c \) for a fixed control policy, which would enable a posterior policy update step in this region.

Acknowledgments

The authors are indebted to the reviewers for their invaluable comments and suggestions, which certainly contributed to the improvement of the paper, both in presentation and in content. In particular, we thank Reviewer #1 for his/her most useful suggestions on the presentation of the results. We also thank Reviewer #2 for motivating the discussion underlying the two remarks at the end of Section 4.1. In fact, the complexity analysis outlined in Remark 1 is essentially an excerpt of his/her review.

This work was partially supported by the Brazilian national research council – CNPq, under Grants 302716/2011-4 and 302501/2010-0 and FAPERJ under Grant E-26/170.008/2008.

References


Please cite this article in press as: Arruda, E. F., & Fragoso, M. D. Solving average cost Markov decision processes by means of a two-phase time aggregation algorithm. European Journal of Operational Research (2014). http://dx.doi.org/10.1016/j.ejor.2014.08.023