On Supervised Online Rolling-Horizon Control for Infinite-Horizon Discounted Markov Decision Processes

Hyeong Soo Chang, Senior Member, IEEE

Abstract—This note revisits the rolling-horizon control approach to the problem of Markov decision process (MDP) with infinite-horizon discounted expected reward criterion. Distinguished from the classical value-iteration approaches, we develop an asynchronous online algorithm based on policy iteration integrated with a multipolicy improvement method of policy switching. A sequence of monotonically improving solutions to the forecast-horizon sub-MDP is generated by updating the current solution only at the currently visited state, building in effect a rolling-horizon control policy for the MDP over infinite horizon. Feedbacks from “supervisors,” if available, can be also incorporated while updating. We focus on the convergence issue with a relation to the transition structure of the MDP. Either a global convergence to an optimal forecast-horizon policy or a local convergence to a “locally-optimal” fixed-policy in a finite time is achieved by the algorithm depending on the structure.

Index Terms—Markov decision process (MDP), policy iteration (PI), policy switching, rolling-horizon control.

I. INTRODUCTION

Consider the rolling horizon control (see, e.g., [2], [5], [8]) with a fixed finite forecast-horizon $H$ to the problem of a Markov decision process (MDP) $M_\infty$, with infinite-horizon discounted expected reward criterion. At discrete time $k \geq 1$, the system is at a state $x_k$ in a finite-state set $X$. If the controller of the system takes an action $a$ in a finite action set $A(x_k)$ at $k$, it obtains a reward of $R(x_k,a)$ from a reward function $R : \{(x,a)|x \in X, a \in A(x)\} \to \mathbb{R}$, where $A(x)$ denotes an admissible-action set of $x \in X$. The system then makes a random transition to a next state $x_{k+1}$ by the probability specified by $P_{x_k,x_{k+1}}$. Let $B(X)$ be the set of all possible functions from $X$ to $\mathbb{R}$. The zero function in $B(X)$ is referred to as $0_X$, where $0_X(x) = 0$ for all $x \in X$. Let also $\Pi(X)$ be the set of all possible deterministic mappings from $X$ to $A$ where for any $\sigma \in \Pi(X)$, $\sigma(x) \in A(x)$ for all $x \in X$. Let an $H$-length policy of the controller be an element in $\Pi(X)^H$, $H$-ary Cartesian product, $h > 1$. That is, $\pi^h \in \Pi(X)^h$ is an ordered $h$-tuple $(\pi^h_1, \ldots, \pi^h_h)$ where the $k$th entry of $\pi^h_k$ is equal to $\pi^h_k \in \Pi(X)$, $k \geq 1$, and when $\pi^h_k$ is applied at $x \in X$, then the controller looks ahead of (or forecasts) the remaining horizon $h-k$ for control. An infinite-length policy is an infinite element in the infinite Cartesian product of $\Pi(X)$, denoted by $\Pi(X)^\infty$, and referred to as just a policy. We say that a policy $\pi^\infty \in \Pi(X)^\infty$ is stationary if $\pi^\infty_k = \pi$ for all $k \geq 1$ for some $\pi \in \Pi(X)$. Given $\pi \in \Pi(X)$, $[\pi]^\infty$ denotes a stationary policy in $\Pi(X)^\infty$ constructed from $\sigma$ such that $[\pi]^\infty_k = \sigma$ for all $k \geq 1$. Note that only deterministic (Markovian) policies are considered in this work for simplicity because an optimal deterministic policy exists in $\Pi(X)^\infty$ for $M_\infty$ and an optimal deterministic $h$-length policy exists in $\Pi(X)^h$ for $M_h$.

Define the $h$-horizon value function $V^h_\pi$ of $\pi^h \in \Pi(X)^h$ for $h \geq 1$ such that for $h = 1, \ldots, H$

$$V^h_\pi(x) = E \left[ \sum_{l=1}^{h} \gamma^{l-1} R(X_l, \pi^h_l(X_l)) + \gamma^h g(x_{h+1}) \mid X_1 = x \right]$$

where $X_l$ is a random variable that denotes a state at the level (of forecast) $l$ by following the $l$th-entry mapping $\pi^h_l$ of $\pi^h$, and a fixed discounting factor $\gamma$ is in $(0,1)$, and the “terminal” reward function $g$ is in $B(X)$. We set $V^0_\pi = g$ for any $\pi$. The expectation operator is applied with the probability distribution over all possible $h$-length trajectories of the state and the action sequences followed by $\pi^h$ given an initial state.

In the sequel, any operator is applied componentwise for the elements in $B(X)$ and in $\Pi(X)$, respectively. Given $\pi^h \in \Pi(X)^h$, the $\pi^h$ is set to be $(\pi^h_1(x), \ldots, \pi^h_h(x))$ meaning the “$x$-coordinate” of $\pi^h$ for $x \in X$. As is well known then, there exists an optimal $h$-length policy $\pi^h_*$ such that for any $x \in X$, $V^h_\pi(x) = \max_{\pi \in \Pi(X)^h} V^h_\pi(x) = V^h_\pi(x)$, where $V^h_\pi$ is the optimal $h$-horizon value function. In particular, $V^h_\pi(x)$ uniquely satisfies the optimality principle

$$V^h_\pi(x) = \max_{a \in A(x)} \left( R(x,a) + \gamma \sum_{y \in X} P_{x,y} V^h_{\pi(x)}(y) \right), x \in X$$

with setting $V^0_\pi = g$. Furthermore, $V^h_\pi$ is equal to the function in $B(X)$ obtained after applying the value-iteration (VI) operator $T : B(X) \to B(X)$ iteratively $h$ times with the initial function of $V^0_\pi$; $T^h = T(T(T(V^0_\pi))) = V^h_\pi$, where $T$ is defined such that for any $x \in X$ and $a \in B(X)$, $T(a)(x) := \max_{a \in A(x)} \{R(x,a) + \gamma \sum_{y \in X} P_{x,y} a(y)\}$. This optimal substructure leads to a dynamic programming algorithm, backward induction, which computes $\{V^h_{\pi^h}(x), h = 1, \ldots, H\}$ offline and returns an optimal $H$-horizon policy $\pi^H_*$ that achieves the optimal value at any $x \in X$ for the $H$-horizon sub-MDP $M_H$ of $M_\infty$ by

$$\pi^H_{x,h+1}(x) \in \arg \max_{a \in A(x)} \left( R(x,a) + \gamma \sum_{y \in X} P_{x,y} V^h_{\pi^h_{x,h-1}}(y) \right)$$

$x \in X$, $h = 1, \ldots, H$. Once obtained, the rolling $H$-horizon controller employs the first entry $\pi^h_1$ of $\pi^H_*$ or a stationary policy $[\pi^H_*]^\infty$ over the system time: At each $k \geq 1$, $\pi^h_{x,h+1}(x_k)$ is taken at $x_k$.

Given a stationary $\pi^\infty \in \Pi(X)^\infty$, $V^\infty_\pi$ refers to the value function of $\pi^\infty$ over infinite horizon $\infty$ it is defined such that $V^\infty_\pi(x) = E[\sum_{l=1}^{\infty} \gamma^{l-1} R(X_l, \pi^\infty_l(X_l)) \mid X_1 = x], x \in X$. It is well known that $V^\infty_\pi$ uniquely satisfies that for all $x \in X$

$$V^\infty_\pi(x) = R(x, \pi^\infty_1(x)) + \gamma \sum_{y \in X} P_{x,y} V^\infty_\pi(y).$$

Manuscript received 24 February 2023; accepted 6 May 2023. Date of publication 10 May 2023; date of current version 30 January 2024. Recommended by Associate Editor A. Mahajan.

The author is with the Department of Computer Science and Engineering, Sogang University, Seoul 121-742, South Korea (e-mail: hschang@sogang.ac.kr).

Digital Object Identifier 10.1109/TAC.2023.3274791
The optimal value function $V_{\pi}^* \leq M_\infty$ is then defined such that $V_{\pi}^*(x) = \sup_{a \in B(x)} V_{\pi}^*(x)$, $x \in X$. Hernández–Lerma and Lasserre [8] analyzed the performance of $[\pi_1^H]_\infty$ relative to an optimal policy that achieves $V_{\pi}^*$. The infinity-norm of the difference between the value function of $[\pi_1^H]_\infty$ and $V_{\pi}^*$ is upper bounded by (an error of) $O(\gamma^H \|V_{\pi}^* - V_{\pi}^*\|_{\infty})$. The term $\|V_{\pi}^* - V_{\pi}^*\|_{\infty}$ can be loosely upper bounded by $C/(1 - \gamma)$ with some constant $C$. Due to the dependence on $(1 - \gamma)^{-1}$, the performance worsens around $\gamma$ closer to one. How to set $V_{\pi}^*$ is a critical issue in the rolling horizon control even if the error vanishes to zero exponentially fast in $H$ with the rate of $\gamma$. (The topic about how to set $V_{\pi}^*$ is beyond the scope of this note.)

Considering possible difficulties of obtaining the exact optimal value function $V_{\pi}^*$, there exists a theoretical analysis [5, Ch. 5] about approximate rolling horizon control. It considers $\pi^*(f) \in \Pi(X)^\infty$ when some approximate function $f \in B(X)$ that can replace $V_{\pi}^*$ is available. The performance of $\pi^*(f)$ with respect to $[\pi_1^H]_\infty$, i.e., $\|V_{\pi}^*(f) - V_{\pi}^*(\pi_1^H)\|_{\infty}$ is given in terms of $\|f - V_{\pi}^*\|_{\infty}$. This approximate-control perspective was discussed in [5, Ch. 5] with the examples of online algorithms based on the (parallel) rollout and hindsight optimization. However, no convergence to optimality is guaranteed for these approaches. Furthermore, no result has been presented with a single-state update.

A main novel contribution of this work is development of an algorithmic solution framework within approximate rolling horizon control for infinite-horizon MDPs that incorporates, in a formal way, supervisor’s (possibly multiple) knowledge, if available. While doing so, we devise an update process based on a variant of policy iteration (PI). It merges with “online policy improvement” such that at each decision time, the update is done only at a single current state. This naturally induces an online algorithm. This iterative PI approach with a single-state update is significant in contrast with existing VI approaches (see, e.g., [5] and [12] and the references therein). This note focuses on a theoretical study establishing some important results about its convergence behavior.

Most closely related online algorithms based on the idea of VI or backward induction are adaptive multi-stage sampling (AMS) [7] and upper confidence bounds applied to trees (UCT) [10], [12] employed within the context of rolling horizon control. AMS inspired UCT and UCT is a common way in implementing “Monte Carlo tree search (MCTS),” which plays a fundamental role in the AlphaGo [2] player. In these approaches, at the current state $x$, $V_{\pi}^*(x)$ is approximated with some simulation method independently over decision times. Unfortunately, the convergence to the optimality is guaranteed theoretically only if an infinite number of samplings are drawn at each decision time. When a finite number of samplings are used, an exponentially growing sample-complexity is a problem for a better approximation. Besides no finite-time convergence properties exist about these approaches within approximate receding horizon control. These drawbacks justifiably motivates this work (see, further remarks about the complexity issue in the concluding remark section).

In this work, a supervisor is brought in the MDP model. Here, a supervisor is seen as a policy of a finite or infinite length. By the policy form, a supervisor is supposed to represent some knowledge on the control of the given MDP. Even if this view on the supervisor might have been used somewhere within an MDP model, to the author’s best knowledge, it seems difficult to find any notable work that presents a method of combining the knowledge of supervisor(s) into the (convergent) process of finding solutions to MDPS.

Closest works would be the author’s previous works of offline “policy-set iteration” [3] and “value-set iteration” [4]. Even though the word “supervisor” was not explicitly used in both papers, the multipolicy improvement methods of parallel rollout [6] and policy switching [6] are used in a way, respectively, to improve exogenously given policies, if exist, simultaneously with the output policies produced at the current iteration for the next iteration at all states. Even though the stochastic multiarmed bandit (MAB) model is a special case of MDPs, MABs have been studied extensively on its own over the years. A related work would be the “expert” used in the Exp4 algorithm in [1, Sec. 7]. In a sense, the expert is similar to supervisor here and the expert provides an advice as a form of a probability distribution over the available arms and Exp4 combines the knowledge of the experts into finding an arm to be played in the bandit.

There is no general way of choosing a supervisor. However, for many problems, domain knowledge is available and heuristics can be designed. We wish to employ those heuristics and policies with domain knowledge somehow in a “merged” way but at the same time, improve all of those. Some example MDP problems of scheduling and admission control in such cases are given in [5, Ch. 5]. Those heuristics (e.g., for the scheduling problem, static-priority policy, earliest-deadline-first scheduling methods) can naturally play the role of supervisors. Furthermore, an optimal action at a particular state of a given problem is known or is obtainable directly by an analysis. For example, serving the highest priority job whose deadline is about to expire is an optimal action at all states with a pending highest priority job whose deadline is imminent. Incorporating such control knowledge into the search process of an optimal policy, in order to possibly speed up search, is desirable. For some MDP problems, heuristics are available such that each heuristic policy is expected to be near-optimal for some subset of the set of trajectories. In sum, any knowledge about the problem, including the cases mentioned, can be conceptualized as supervisor or teacher and can be represented as a policy. Therefore, it is very important to devise a formal way of combining these supervisor policies into a single policy whose performance is no worse than any supervisor policy.

In the algorithm presented here, the sub-MDP $M_H$ is not solved in advance. Rather with an arbitrarily selected $\pi_{H,1} \in \Pi(X)^H$ for $M_H$, the algorithm generates a monotonically improving sequence of $\{\pi_{H,k}\}$ over time $k \geq 1$ with respect to $M_H$. To the algorithm, only $\pi_{H,k-1}$ is available at $k > 1$ and it updates $\pi_{H,k-1}$ only at state variables $x_k$. Either we have that $\pi_{H,k} : = \pi_{H,k-1}^*$ or $\pi_{H,k} (x) = \pi_{H,k-1} (x)$ for all $x \in X \setminus x_k$ but $\pi_{H,k} (x_k) \neq \pi_{H,k-1} (x_k)$. The algorithm has a design flexibility in the aspect that a feedback of an action to be used at state variables $x_k$ by some supervisor can be incorporated while generating $\pi_{H,k}$. By setting $\phi_k^* = \pi_{H,k}^*$ at each $k \geq 1$, a policy $\phi_{\infty} \in \Pi(X)^\infty$ is in effect built sequentially for the controller. Once $\phi_{\infty}^*$ is available to the controller, it takes $\phi_{\infty}^*(x_k)$ to the system and the underlying system of $M_\infty$ moves to a next random state $x_{k+1}$ by the probability of $P_{\pi_{\infty}^*} (x_k)$. The behavior of such a control policy is discussed by focusing on the convergence issue with a relation to the transition structure of $M_\infty$.

We are concerned with a question about the existence of a finite time $K < \infty$ such that $\phi_{\infty}^* = \pi_{H,k}^*$ for all $k > K$ for the infinite sequence $\{\phi_{\infty}^*\}$.

II. OFFLINE SYNCHRONOUS POLICY ITERATION WITH POLICY SWITCHING (PIPS)

To start with, we present an algorithm of offline synchronous PI combined with a multipolicy improvement method of policy switching for solving $M_H$. Throughout the note, we assume that the value of $H$ is fixed unless stated otherwise. We also assume that $q = V_0 = 0_X$ for any $h \in \Pi(X)^h$, $h = 1, \ldots, H$, for simplicity.

Given $\pi_H^*$ and $\pi_{H} \in \Pi(X)^H$, we say that $\pi_H^*$ improves $\pi_H$ (over the horizon $H$) if $V_{\pi_H}^H \geq V_{\pi_H^*}^H$ in which case we write as $\pi_H^* \geq_H \pi_H$. By policy switching [6], we switch to a best policy with respect to each
initial state over a policy set. The resulting policy improves all policies in the policy set.

**Theorem 2.1 (Theorem 2 [6]):** Given a nonempty $\Delta \subseteq \Pi(X)^H$, construct policy switching with $\Delta$ in $\Pi(X)^H$ backwards as $\pi^H_\delta(\Delta)$ such that for each possible pair of $x \in I$ and $h = 1, \ldots, H$,

$$\pi^H_\delta(\Delta)_{H-h+1}(x) = \phi_{H-h+1}(x)$$

where

$$\phi_{H-h+1}(x) = \arg \max_{\phi \in \Delta} V^h_{\phi}(x)$$

for each $\phi \in \Delta$, $\phi^h = (\phi^H_{H-h+1}, \ldots, \phi^H_1) \in \Delta^h$. Then,

$$V^h_{\phi}(\Delta) \supseteq V^h_{\phi}$$

for all $\phi \in \Delta$.

Define switchable action set $S^H_{\phi}(x)$ of $\pi^H \in \Pi(X)^H$ at $x \in X$ for $h = 1, \ldots, H$, as

$$S^H_{\phi}(x) = \left\{ a \in A \left| R(x, a) + \gamma \sum_{y \in X} P^a_{xy} V^{h-1}_{\phi}(y) > V^h_{\phi}(x) \right\} \right.$$ 

where $\pi^h$, $h = 1, \ldots, H$, is obtained from $\pi^H$ such that $\pi^h = (\pi^H_{H-h+1}, \ldots, \pi^H_1)$ and also improvable state set of $\pi^H$ for $h$ as

$$I^h = \left\{ (h, x) \mid S^H_{\phi}(x) \neq \emptyset, x \in X \right\}.$$ 

Set $I^{\phi^H, H} = \bigcup_{h=1}^H I^h$.

The switchable action set of a given policy $\pi^H$ at $x$ for horizon $h$ is the set of actions in $A(x)$ that can switch or change from $\pi^H_{H-h+1}(x)$ at $x$ so that a switchable action at $x$ is taken instead of $\pi^H_{H-h+1}(x)$, the resulting value (by taking action and then following $\pi^H$) is bigger than just following $\pi^H$ over $h$-horizon. The improvable-state set of $\pi^H$ for $h$ is the set of all possible pairs of horizon size $h$ and states such that the switchable action at a given set at a state for $\pi^H$ is not empty. (Observe (based on an induction argument) that if $I^{\phi^H, H} = \emptyset$ for $\pi^H \in \Pi(X)^H$, then $\pi^H$ is an optimal $H$-length policy for $M_H$.

The following theorem provides a result for $M_H$ in analogy with the key step of the single-policy improvement (see, e.g., [11]) in $P_{\pi_n}$. Because Banach’s fixed-point theorem is difficult to be invoked in the finite-horizon case unlike the standard proof for the infinite-horizon case, we provide a proof for the completeness.

**Theorem 2.2:** Given $\pi^H \in \Pi(X)^H$ with $I^{\phi^H, H} \neq \emptyset$, construct $\tilde{\pi}^H \in \Pi(X)^H$ with any $I$ satisfying $0 \subseteq I \subseteq I^{\phi^H, H}$ such that

$$\tilde{\pi}^H_{H-h+1}(x) = S^H_{\phi}(x)$$

for all $(h, x) \in I$ and

$$\tilde{\pi}^H_{H-h+1}(x) = \pi^H_{H-h+1}(x)$$

for all $(h, x) \in \{(1, \ldots, H) \times X \} \setminus I$. Then $\tilde{\pi}^H \geq H \pi^H$.

**Proof:** The base case holds because $V^0_{\phi} = V^H_{\phi} = 0$. For the induction step, assume that $V^h_{\phi} \geq V^h_{\phi}$. For all $x$ such that $\tilde{\pi}^H_{H-h+1}(x) = \pi^H_{H-h+1}(x)$,

$$V^h_{\phi}(x) = R(x, \pi^H_{H-h+1}(x)) + \gamma \sum_{y \in X} P^y_{xy} V^{h-1}_{\phi}(y) \geq R(x, \pi^H_{H-h+1}(x)) + \gamma \sum_{y \in X} P^y_{xy} V^{h-1}_{\phi}(y) = V^h_{\phi}(x).$$

On the other hand, for any $x \in X$ such that $\tilde{\pi}^H_{H-h+1}(x) \in S^H_{\phi}(x)$,

$$V^h_{\phi}(x) = R(x, \tilde{\pi}^H_{H-h+1}(x)) + \gamma \sum_{y \in X} P^y_{xy} V^{h-1}_{\phi}(y) \geq R(x, \tilde{\pi}^H_{H-h+1}(x)) + \gamma \sum_{y \in X} P^y_{xy} V^{h-1}_{\phi}(y) = V^h_{\phi}(x).$$

by induction hypothesis $V^h_{\phi} \geq V^h_{\phi}$. Putting the two cases together makes $V^h_{\phi} \geq V^h_{\phi}$.

The previous theorem states that if a policy was generated from a given $\pi^H$ by switching the action prescribed by $\pi^H$ at each improvable state with its corresponding level in a particularly chosen nonempty subset of the improvable-state set of $\pi^H$, then the policy constructed improves $\pi^H$ over the relevant finite horizon. However, in general, even if $\pi \geq_H \phi$ is known, for $\pi^H$ and $\phi^H$ obtained by the method of Theorem 2.2, respectively, $\pi^H \geq_H \phi^H$ does not hold necessarily. (Note that this is also true for the infinite-horizon case.) It can be merely said that $\pi^H$ improves $\pi^H$ and $\phi^H$ improves $\phi^H$, respectively. Motivated by this, we consider the following. For a given $\pi^H$ in $\Pi(X)^H$, let $\beta^\pi^H$ be the set of all policies no worse than $\pi^H$ obtainable from $I^{\phi^H, H}$: If $I^{\phi^H, H} = \emptyset$, $\beta^\pi^H = \emptyset$. Otherwise

$$\beta^\pi^H = \{ \tilde{\pi}^H \in \Pi(X)^H \mid \exists I \subseteq 2^{I^{\phi^H, H}} \setminus \emptyset \quad \forall (h, x) \in I$$

$$\tilde{\pi}^H_{H-h+1}(x) \subseteq S^H_{\phi}(x) \text{ and } \forall(h, x) \in \{(1, \ldots, H) \times X\} \setminus I$$

$$\tilde{\pi}^H_{H-h+1}(x) = \pi^H_{H-h+1}(x) \},$$

where $2^L$ denotes the power set of a set $L$. Once obtained, policy switching with respect to $\beta^\pi^H$ is applied to find a single policy no worse than all policies in the set.

We are ready to derive an offline synchronous algorithm, PIPS, that generates a sequence of $H$-length policies for solving $M_H$. Define arbitrarily $\pi^H_{n+1}(X)$. Loop with $n = 1, 2, \ldots$, until $I^{\phi^H, n+1} = \emptyset$

$$\pi^H_{n+1}(x) = \pi^H_{n+1}(x).$$

The convergence to an optimal $H$-length policy for $M_H$ is trivially guaranteed because $\pi^H_{n+1} \geq H \pi^H_{n+1}, n \geq 1$, and both $H$ and $A$ are finite. Note that $\beta^\pi^H(n)$ in $\beta^\pi^H(H)$ can be substituted with any $\Delta_n \subseteq \Pi(X)^H$ as long as $\Delta_n \cap \beta^\pi^H(n) \neq \emptyset$ for all $n$ to keep the monotonicity. The generality of $\Delta_n$ then provides a broad design-flexibility of PIPS.

The idea behind policy switching used in PIPS with $\beta^\pi^H(n)$ can be attributed to approximating the steepest ascent direction while applying the steepest ascent algorithm. At the current location $\pi^H_{n+1}$, we find ascent “directions” relative to $\pi^H_{n+1}$ over the local neighborhood of $\beta^\pi^H(n)$: A steepest ascent direction, $\pi^H_{n+1}(\beta^\pi^H(n))$, is then obtained by “combining” all of the possible ascent directions. In particular, the greedy ascent direction $\phi^H$ that satisfies that

$$T(V_{\phi^H}^{n+1}(x)) = R(x, \phi^H_{H-h+1}(x)) + \gamma \sum_{y \in X} P^y_{xy} V_{\phi^H}^{n+1}(y) \geq \gamma \sum_{y \in X} P^y_{xy} V_{\phi^H}^{n+1}(y)$$

for all $x \in X$ and $h = 1, \ldots, H$, is always included while combining.

**III. Offline Asynchronous PIPS**

An asynchronous version can be inferred from the synchronous PIPS by the following improvement result when a single $H$-length policy in $\Pi(X)^H$. $M_H$ is updated only at a single state: Given $x \in X$ and $\pi^H \in \Pi(X)^H$, consider a subset $I^x_{\phi^H} \subseteq I^{\phi^H, H}$ such that $I^x_{\phi^H} \cap \Delta_n$ contains $(h, x)$ if $x$ is an improvable state of $\phi^H$ for $h = 1, \ldots, H$. In a sense, we make “projection onto $x$” from $I^{\phi^H, H}$. Then, given $I^x_{\phi^H}$, consider
also a subset $\beta_2^H,H \subseteq \beta^H,H$ such that $\beta_2^H,H$ is the set of all policies no worse than $\pi^H$ obtainable from $I^H,H$. That is

$$\beta_2^H,H = \left\{ \pi \in \beta^H,H \mid \exists I \subseteq 2^{\mathcal{X}^H} \setminus \{\emptyset\} \quad \forall (h,x) \in I \right\}$$

$$\pi^{H,h-1}(x) \in S_h^H(x) \text{ and } (h,x') \in (\{1,H\} \times \mathcal{X}) \setminus I$$

$$\pi^{H,h-1}(x') = \pi^{H,h-1}(x')$$.

Because $I^H,H \subseteq I^N,H$ and $\beta_2^H,H \subseteq \beta^H,H$, obviously the following corollary of Theorem 2.2 holds.

**Corollary 3.1**: Given $x \in \mathcal{X}$ and $\pi^H \in \Pi(X)^H$, suppose that $I^H,H \neq \emptyset$. Then, for any $\phi^H \in \beta_2^H,H$, $\phi^H \geq H \pi^H$.

This result leads to an offline convergent asynchronous PIPS for $M_H$: Select $\pi^H \in \Pi(X)^H$ arbitrarily. Loop with $n = 1,2,\ldots$ If $I^H,H \neq \emptyset$, select $x_n$ if there exists $h$ such that $(h,x_n) \in I^H,H$ and construct $\pi^{H,n+1}(x_n)$ such that

$$\pi^{H,n+1}(x_n) = \pi^n_H(\beta^{H,n}(x_n))$$

and $\pi^{H,n+1}(x) = \pi^H(x)$ for all $x \in \mathcal{X} \setminus \{x_n\}$. If $I^H,H \neq \emptyset$, stop. Because $x_n$ is always selected to be an improvable-state from nonemptiness $I^H,H, \pi^{H,n+1} \subseteq \pi^H,H$ for all $n \geq 1$. Therefore, $\{\pi^H,n\}$ converges to an optimal $H$-length policy for $M_H$. Note that policy switching is applied only at a single state $x_n$. The same result of Theorem 2.1 still holds with switching at a single state and the proof can be done by an inductive argument similar to the proof of Theorem 2.1.

Suppose that the state given at the current step of the previous algorithm is not guaranteed to be obtainable from the improvable-state set of the current policy. Such scenario is possible with the following modified version: Select $\pi^H \in \Pi(X)^H$ arbitrarily. Loop with $n = 1,2,\ldots$ If $I^H,H \neq \emptyset$, stop. Select $x_n \in \mathcal{X}$. If there exists $h$ such that $(h,x_n) \in I^H,H$, then construct $\pi^{H,n+1}(x_n)$ such that

$$\pi^{H,n+1}(x_n) = \pi^n_H(\beta_2^{H,n}(x_n))$$

and $\pi^{H,n+1}(x) = \pi^H(x)$ for all $x \in \mathcal{X} \setminus \{x_n\}$. If no $h$ exists such that $(h,x_n) \in I^H,H$, $\pi^{H,n+1} = \pi^H$. Unlike the previous version, this algorithm’s convergence depends on the sequence $\{x_n\}$ selected. Even if $\pi^{H,n+1} \geq \pi^H$ when $\pi^{H,n+1} \neq \pi^H$, the stopping condition that checks for the optimality (Bellman’s equation) can never be satisfied. In other words, an infinite loop is possible. The immediate problem is then how to choose an update-state sequence to achieve a global convergence. The reason for bringing this issue up with the modified algorithm is that the situation is closely related with the online algorithm to be discussed in the following section. Dealing with this issue here would help understanding the convergence behavior of the online algorithm. We discuss some pedagogical example of choosing an update-state sequence of the modified offline algorithm as follows.

One way of enforcing a global convergence is to "embed" backward induction into the update-state sequence. For example, we generate a sequence of

$$\{x_n\} = \{x_0,0,\ldots,x_{n_1},\ldots,x_{n_2},\ldots,x_{n_3},\ldots,x_{n_H},\ldots\}$$

whose subsequence $\{x_{n_1}\}, h = 1,\ldots,H$ produces $\pi^{H,n_1}$ that solves $M_h$. We need to follow the optimality principle such that $M_h$ is solved before $M_h$, and so forth, until $M_H$ is finally solved. Therefore, the entries of $\pi^{H,*}$ are searched from $\pi^H$ to $\pi^{H,*}$ over $\{x_n\}$ such that

$$\pi^{H,n_1} = (\pi^H,n_1,\ldots,\pi^{H,n_2},\pi^{H,n_3})$$

where $\pi_{i}^{H,n} = V_{i}^{x_i}$, and then

$$\pi^{H,n_2} = (\pi^H,n_2,\ldots,\pi^{H,n_3},\pi^{H,n_4},\pi^{H,n_5})$$

where $\pi_{i}^{H,n_2} = V_{i}^{x_i}$, and then

$$\pi^{H,n_3} = (\pi^H,n_3,\pi^{H,n_4},\pi^{H,n_5})$$

where $\pi_{i}^{H,n_3} = V_{i}^{x_i}$, and then finally

$$\pi^{H,n_4} = (\pi^H,n_4,\pi^{H,n_5},\pi^{H,n_6})$$

with $V_{i}^{x_i}$ being $V_{i}^{x_i}$.

Once $M_{h-1}$ has been solved, an optimal $h$-length policy $\pi^{H,n_h}$ for $M_h$ can be found exhaustively. The corresponding update-state subsequence from $x_{n_1,n_2-1}$ to $x_{n_2}$ can be any permutation of the states in $X$. Visiting each $x$ in $X$ at least once for updating causes an optimal $h$-length policy for $M_h$ to be found because if not empty, $\beta^{H,m,h}$, where $m = n_{h-1} + 1,\ldots,n_{h}$, includes an $H$-length policy whose $(H+h-1)$th entry mapping maps $x$ to an action in $\arg \max_{a \in A(x)}(R(x,a) + \sum_{y \in X} P_{xy}(y, \gamma))$. Even though visiting each state at least once makes the approach enumerative, our point is showing that there exists an update-state sequence that makes a global convergence possible.

Let us consider a very simple toy example given as follows: $X = \{x_1, x_2\}, A(x_1) = \{s, m\}, A(x_2) = \{s\}$, where $s$ represents "stay" and $m$ means "move." $R(x_1, s) = 5, R(x_1, m) = 10, R(x_2, s) = 1$, and $q = 0.5$. $P_{x_1,x_2} = P_{x_2,x_2} = 0.5, P_{x_2,x_1} = 1, P_{x_2,x_2} = 1$. Let $\gamma = 0.95$. We choose $H = 3$. The optimality equations are then for $h \geq 1$.

$$V^{h}(x_1) = \max\{5 + \gamma(0.5V^{h}_{x_2}(x_1) + 0.5V^{h}_{x_2}(x_2)), 10 + \gamma V^{h}_{x_2}(x_2)\}$$

and $V^{h}(x_2) = -1.95 + \gamma(0.5V^{h}_{x_2}(x_1) + 0.5V^{h}_{x_2}(x_2))$. We have that $V^{h}_{x_1}(x_1) = 10, V^{h}_{x_1}(x_2) = 1$ and $V^{h}_{x_2}(x_2) = 2$. $V^{h}_{x_2}(x_2) = 9.275 = \max(9.275, 9.05), V^{h}_{x_2}(x_2) = 9.275$. Similarly, $V^{h}_{x_2}(x_1) = \max(8.4794, 8.1475)$ and $V^{h}_{x_2}(x_2) = 8.8523$. The optimal policy $\pi^H$ is given such that $\pi^H = (\pi^1, \pi^2, \pi^3)$ where $\pi^1(x_1) = s, \pi^2(x_2) = s, \pi^3(x_2) = s$. Similarly, $\pi^H$ is referred to as stay-policy.

We now show the idea of asynchronous PIPS. Let us choose the move-policy $\pi^H$ as a starting policy and $x_1$ as the first update-state, for example. (We remark that even though $x_2$ has only one action, this structure was purposely devised for the simpler computation and illustration.) We obtain $I^{H}_x = \{3, (x_1), (x_2)\}$. Therefore, $\beta^{H,3}_x$ consists of the policies no worse than $\mu^H$ that can be obtained by switching actions at $x_1$ for both $h = 3$ and $h = 2$ or either case. To obtain $\pi^{(\beta^{H,3}_x)}(x_1)$, rather than applying policy switching with all policies in $\beta^{H,3}_x$, we consider only $\{3, x_1\}$. Because arg max$(5 + 0.95(0.5V^{h}_{x_2}(x_1) + 0.5V^{h}_{x_2}(x_2)), 10 + 0.95V^{h}_{x_2}(x_2)) = \arg \max(8.3725, 8.1475)$, this provides an improved policy $\mu^H$ of $\mu^H$, where $\mu^{1}_x(x_1) = s, \mu^{2}_x(x_2) = s, \mu^{3}_x(x_1) = m, \mu^{3}_x(x_2) = s$. We then have that $\mu^{H}(x_1) = \pi^{H}(x_1)$. Therefore, $\beta^{H,3}_x, \beta^{H,3}_x$ can be replaced by $\beta^{3}(x_1), \beta^{3}(x_2)$. In other words, $\beta^{H,3}_x(3, (x_1), (x_2))$. If $x_2$ is chosen for the next
A simple and obvious sufficient condition for such connectivity is that 

\( P_{xy}^\infty > 0 \) for all \( x, y \) in \( X \) and \( \mu \in A(x) \). The key in the convergence here is that which state in \( X \) is visited “sufficiently often” by following \( \{\phi_k\} \) to ensure that an optimal action at the visited state is eventually found. The following result stated in Theorem 4.2 reflects this rationale. Given a stationary policy \( \phi^\infty \in \Pi(X)^\infty \), the connectivity relation \( \chi^\infty \) is defined on \( X \) from the Markov chain \( M^\infty \) induced by fixing \( \phi^\infty \) in \( M_\infty \); If \( x \) and \( y \) in \( X \) communicate each other in \( M^\infty \), \( (x, y) \) is an element of \( \chi^\infty \). Given \( x \in X \), the equivalence class of \( x \) with respect to \( \chi^\infty \) is denoted by \( [x]_{\chi^\infty} \). Note that for any \( x \neq y \), either \( [x]_{\chi^\infty} = [y]_{\chi^\infty} \) or \( [x]_{\chi^\infty} \cap [y]_{\chi^\infty} = \emptyset \). The collection of \( [x]_{\chi^\infty}, x \in X \), partitions \( X \).

**Theorem 4.2:** For any \( \pi^{\infty \infty} \in \Pi(X)^\infty \) and any \( \{\Delta_k\} \) where \( \Delta_k \subseteq \Pi(X)^H, k \geq 1 \), \( \{\pi^{\infty \infty}, \Delta_k\} \) generated by online asynchronous PIPS converges to some \( \lambda^H \) in \( \Pi(X)^H \) such that for some \( K < \infty, \pi^{\infty \infty} = \lambda^H \) for all \( k > K \). Furthermore, \( \lambda^H \) satisfies that \( V^H_\infty \geq V^H_\infty \) for all \( \pi^H \in \bigcup_{x \in [x^1_{\lambda^H}]} \beta^2_{x \in [x^1_{\lambda^H}]} \),

where \( x^* \) is any visited state at \( k > K \).

**Proof:** By the same reasoning in the proof of Theorem 4.1, \( \{\pi^{\infty \infty}, \Delta_k\} \) converges to an element \( \lambda^H \) in \( \Pi(X)^H \) in a finite time \( K \). Because every state \( x \) in \( [x^1_{\lambda^H}] \) is visited infinitely often within \( \{x_k\} \) for \( k > K, I_x^{\infty H} = 0 \) for such \( x \). More no improvement is possible at all states in \( [x^1_{\lambda^H}] \). Otherwise, it contradicts the convergence to \( \lambda^H \).

The theorem that \( \lambda^H \) is “locally optimal” over \( [x^1_{\lambda^H}] \) in the sense that no more improvement is possible at all states in \( [x^1_{\lambda^H}] \).

We remark that the abovementioned local convergence result is different from the result by Bertsekas [2, Ch. 3]. In our case, a subset of \( X \) in which every state is visited infinitely often is not assumed to be given in advance. The sequence of policies generated by online PIPS will eventually converge to a policy and Theorem 4.2 characterizes its local optimality with respect to the communicating classes, in which every state is visited infinitely often, induced by the policy. Another important difference is that the algorithm of [2, Ch. 3] has a local optimality for \( M_\infty \), whereas the result reported here is for \( M_\infty \).

Even if we asserted the convergence property of PIPS to \( \lambda^H \), it is open to provide a performance bound of \( V^{\infty H}_{\lambda^H} \) relative to \( V^{\infty H}_{\lambda^H} \) in terms of the structure or the parameters of a given MDP. This is mainly from the difficulty of establishing a relative performance bound between consecutive policies, \( \pi^{\infty \infty} \) and \( \pi^{\infty \infty} \), produced by PIPS. Even if we know that \( \pi^{\infty \infty} \) improves \( \pi^{\infty \infty} \), the degree of the improvement, i.e., how much it improves is not known yet. This will play a crucial role in analyzing a bound on the performance. In fact, the same difficulty is true of the PI case for solving infinite-horizon MDPs (see, e.g., [2, Sec. 3.3, Ch. 3] for a related remark). The analysis result of the relative distance between consecutive policies will be a key to solving the fundamental open complexity-problem about PI. (There does exist known results about the complexity about the number of iterations of PI (see, e.g., [14]), but not about the performance-improvement behavior of PI.)

More specifically, \(||V^{\infty H}_{\lambda^H} - V^{\infty H}_{\lambda^H}|||_\infty \) is bounded above by \(||V^{\infty H}_{\lambda^H} - V^{\infty H}_{\lambda^H}|||_\infty + ||V^{\infty H}_{\lambda^H} - V^{\infty H}_{\lambda^H}|||_\infty ||. The second norm is bounded by the classic result [8] as we mentioned in the introduction section. The first norm is bounded by [5, Th. 5.1].

\[ ||V^{\infty H}_{\lambda^H} - V^{\infty H}_{\lambda^H}||| \leq O((\gamma^H/(1 - \gamma)) + O(\epsilon/(1 - \gamma))) \]
if $\|V_{\pi'}^H - V_{\pi''}^H\|_\infty \leq \epsilon$. As we can see, the degree of approximation $\epsilon$ by $\lambda^H$ with respect to $\pi^H$ in terms of the value function distance will determine the performance of the rolling horizon control by the online asynchronous PIPS algorithm. Unfortunately, it is still open to characterize the value of $\epsilon$ in advance in terms of the parameters of the model or the structure besides the unknown $V_{\pi}^H$ and $V_{\pi'}^H$.

V. Conclusion

Online PIPS is also in the category of “learning” control. Essentially, $V_{\pi}$ can be thought as an initial knowledge of control to the system, or by switching as in the rollout algorithm case (see, [2] for related discussions). The $(|X|^2|A|)^H$-complexity for obtaining the improvable set becomes $O(C'|A|)$, again independently of $|X|$, where $C'$ is the sample-complexity of estimating the switchability of one action for a horizon. In sum, PIPS can be implemented with a much lower sample-complexity (polynomial in $H$) than the VI approaches.

Empirical verification of theories presented in the work and/or an experimental study with practical problems with a study on computational complexities are important. This note focused on the theoretical development of an algorithmic framework and the establishment about its convergence behavior, we leave experimental study as a future work.

Finally, it can be checked that another multipolicy improvement method of parallel rollout [6] does not work for preserving the monotonocity property with asynchronous update when the set of more than one policies is applied to the method for the improvement. Even with synchronous update, the parallel-rollout approach requires estimating a double expectation for each action, one for the next-state distribution and another one in the value function (to be evaluated at the next state). In contrast, in policy switching a single expectation for each policy needs to be estimated leading to a lower simulation-complexity.

References

[1] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. Schapire, “The nonstochastic multiarmed bandit problem,” SIAM J. Comput., vol. 32, no. 1, pp. 48–77, 2002.

[2] D. P. Bertsekas, Lessons From AlphaZero for Optimal, Model Predictive, and Adaptive Control. Belmont, MA, USA: Athena Scientific, 2022.

[3] H. S. Chang, “Policy set iteration for Markov decision processes,” Automatica, vol. 49, no. 12, pp. 3687–3689, 2013.

[4] H. S. Chang, “Value set iteration for Markov decision processes,” Automatica, vol. 50, no. 7, pp. 1940–1943, 2014.

[5] H. S. Chang, J. Hu, M. Fu, and S. I. Marcus, Simulation-Based Algorithms for Markov Decision Processes. London, U.K.: Springer, 2013.

[6] H. S. Chang, R. Givan, and E. K. P. Chong, “Parallel rollout for on-line solution of partially observable Markov decision processes,” Discr. Event Dyn. Syst. Theory Appl., vol. 15, no. 3, pp. 309–341, 2004.

[7] H. S. Chang, M. C. Fu, J. Hu, and S. I. Marcus, “An adaptive sampling algorithm for solving Markov decision processes,” Operations Res., vol. 53, no. 1, pp. 125–139, 2005.

[8] O. Hernandez-Lerma and J. B. Lasserre, “Error bounds for rolling horizon policies in discrete-time Markov control processes,” IEEE Trans. Autom. Control, vol. 35, no. 10, pp. 1118–1124, Oct. 1990.

[9] L. C. M. Kallenberg, “Classification problems in MDPs,” in Markov Processes and Controlled Markov Chains, Z. Hou, J. A. Filar, and A. Chen, Eds., Boston, MA, USA: Springer, 2002.

[10] L. Kocsis and C. Szepesvári, “Bandit based Monte-Carlo planning,” in Proc. Eur. Conf. Mach. Learn., 2006, pp. 282–293.

[11] M. L. Puterman, Markov Decision Processes: Discrete Stochastic Dynamic Programming. New York, NY, USA: Wiley, 1994.

[12] D. Shah, Q. Xie, and Z. Xu, “Nonasymptotic analysis of Monte Carlo tree search,” Operations Res., vol. 70, no. 6, pp. 3234–3260, 2022.

[13] J. N. Tsitsiklis, “NP-Hardness of checking the unichain condition in average cost MDPs,” Operations Res. Lett., vol. 35, pp. 319–323, 2007.

[14] Y. Ye, “The simplex and policy-iteration methods are strongly polynomial for the Markov decision problem with a fixed discount rate,” Math. Operations Res., vol. 36, pp. 593–603, 2011.