Accepted by Journal of AI Communications, 2018

Two-phase Selective Decentralization to Improve Reinforcement Learning Systems with MDP

Thanh Nguyen [a] and Snehasis Mukhopadhyay [a]

[a] *Department of Computer and Information Science, Indiana University Purdue University Indianapolis, 723 W Michigan St SL 280 Indianapolis, Indiana 46202, United States*
E-mails: thamnguy@iupui.edu, smukhopa@iupui.edu

Abstract. In this paper, we explore the capability of selective decentralization in improving the reinforcement learning performance for unknown systems using model-based approaches. In selective decentralization, we automatically select the best communication policies among agents. Our learning design, which is built on the control system principles, includes two phases. First, we apply system identification to train an approximated model for the unknown systems. Second, we find the suboptimal solution of the Hamilton-Jacobi-Bellman (HJB) equation to derive the suboptimal control. For linear systems, the HJB equation transforms to the well-known Riccati equation with closed-form solution. In nonlinear system, we discretize the approximation model as a Markov Decision Process (MDP) in order to determine the control using dynamic programming algorithms. Since the theoretical foundation of using MDP to control the nonlinear system has not been thoroughly developed, we prove that the control law learned by the discrete-MDP approach is guarantee to stabilize the system, which is the learning goal, given several sufficient conditions. These learning and control techniques could be applied in centralized, completely decentralized and selectively decentralized manner. Our results show that selective decentralization outperforms the complete decentralization and the centralization approaches when the systems are completely decoupled or strongly interconnected.

Keywords: decentralized control, Hamilton-Jacobi-Bellman equation, Markov process, multi-agent systems

1. Introduction

To deal with the complexity AI systems, decentralization and multi-agent learning has been one of the major approaches in reinforcement learning. Decentralization decouples the entire system‘s state variables into subsystems using domain knowledge or partition techniques and assigns an agent for each subsystem. Each agent is responsible to learn the optimal control strategy for the assigned subsystem. With decentralization, the learning algorithms operate on less number of state variables and are less susceptible to uncertain system parameters [1]. In addition, decentralization makes the system more adaptive to structural changes than the corresponding centralized systems [2]. Another benefit of decentralization is that if one agent fails in learning, the other agents could compensate for it in the overall learning problem resulting in only graceful degradation of performance [3]. Although decentralization is a promising approach for large-scale reinforcement learning, this type of approach is likely to suffer from instability in the presence of interconnections among subsystems regardless of the interconnection strength [1, 4].

To overcome the stability issue, one of the key questions in decentralized learning is to set up a communication policy among the learning agents. The question of how to choose a suitable communication policy to use is still open because the number of communication policies grows following the Bell‘s number, which is more than exponential [5]. To the extent of our knowledge, there are two classical approaches in designing communication policy in decentralized learning: partial communication and multi-model switching. In partial communication, each agent is responsible to select the other agents to communicate with, depending on the agent‘s state variables and communication costs [6]. Some of the recent state-of-the-art techniques in partial communication demonstrate how each agent decides the communication in Q-learning problems [7–9], partially ordered subsystems [10], fuzzy logic systems [11, 12] and probabilistic control shar-

This is the author's manuscript of the article published in final edited form as:

Nguyen, T., & Mukhopadhyay, S. (2018). Two-phase selective decentralization to improve reinforcement learning systems with MDP. AI Communications, 31(4), 319–337. https://doi.org/10.3233/AIC-180766

ing systems [13]. In multi-model-switching, the entire system has K policies to allow the agents to communicate, and the entire system has a central communicator who is responsible to switch the communication policy depending on the resulting performance [14–17]. Also, criteria to decide policy switch may depend on the domain-specific optimization of the problem, such as power efficiency function in energy system [18, 19] and aerodynamic performance in hypersonic vehicle systems [20]. In addition, communication among agents also depends on the characteristics of the tasks, or the final goals, of the entire system. From this perspective, the communication policy and learning algorithms could be categorized into fully cooperative tasks, explicit coordination mechanisms, fully competitive tasks and mixed tasks [3]. Although the communication policy problem has been broadly explored, the existing solutions still require full or partial knowledge about the agents‘ connectivity and operating regimes. Other practical questions in decentralization are how to create and justify the subsystem decompositions, and how fast the decentralized learning algorithms converge.

From the theoretical point of view, a reinforcement learning AI problem could be considered as an adaptive control problem [21], in which solving the Hamilton-Jacobi-Bellman (HJB) equation is the theoretical key in the reinforcement learning and control system theory. Most of the decentralization techniques focus on learning linear systems [2, 4], in which the centralized and decentralized system could be uniformly represented in matrix form. For the linear system, the HJB equation becomes the well-known Riccati equation with a complete solution [22]. However, in most of the real-world cases, the system is nonlinear where the closed-form solution for HJB equation is very difficult to find. Solving the nonlinear HJB equation in decentralized manner is even more difficult. Therefore, researchers have been focusing on approximation methods to tackle nonlinear HJB equation problem such as [23, 24, 26, 35]. Generally, these efforts focus on the nonlinear feedback-linearization system, in which the closed-form solution for the approximation of HJB equation has been found [27]. Theoretically, the HJB equation could be solved with dynamic programming [28]. Therefore, a simple idea is to discretize the nonlinear system to convert it into a Markov-Decision-Process (MDP) and solve it by the policy iteration algorithm [29]. Such discretization of continuous-state nonlinear control systems has been studied in [30–32]. Results of MDP convergence for decentralized learning in Markov systems have been derived in [33, 34]. In addition, the matrix-properties of MDP could support the representation of decentralized learning and control. With this discretization approach, we successfully solved the nonlinear control problem in several case-studies. However, from our knowledge, the theoretical proof about the existence and approximation of the MDP‘s solution in the general form HJB equation has not been widely explored.

In addition, the adaptive control and reinforcement learning has another problem due to the unknown nature of the systems. However, this problem could be tackled by system identification techniques. System identification constructs an approximation to model the dynamical changes of the system and environment [35]. For linear system identification, the gradient descent is one of the most robust methods as shown in [36]. For nonlinear system, neural network is one of the most well-known approaches for identification. Neural networks have been known for their capability to approximate a large and general class of nonlinear functions over compact domains. Theoretical foundation and application of neural network as such universal functional approximators in control systems can be found in [23, 37, 38].

In this paper, we make two major contributions: 1. Inspired by the model-switching ideas, we propose the ***selective decentralization method***, to learn how to control the completely unknown-interconnection system in two-phase approach: system identification and control in fully cooperative tasks problem. This method also allows the learning agents to learn the suboptimal communication policy when the agents‘ connectivity and operating regimes are completely unknown. 2. We design a ***discretized-MDP approach*** to tackle the nonlinear HJB equation in the most general form, due to the assumption that the AI reinforcement learning system is completely unknown. The discretized-MDP approach helps in the control phase in the nonlinear-system case. We also provide theoretical analysis about the necessary conditions for the MDP‘s discrete state vector to converge to the real continuous state vector asymptotically. In addition, we also prove that the MDP‘s solution guarantees to stabilize the learning systems in general form when the systems satisfy certain conditions.

From our knowledge, the approach using the decentralized method with system identification to unknown system, especially beyond the feedback-linearization systems, is relatively unexplored. Our focus in this work in the nonlinear system. However, we inclue sev-

eral examples of linear system to demonstrate how the selective decentralization perfrom in a well-known and well-sloved problem. We also compare the control performance of our selective decentralization method with the completely decentralized method and the centralized method using simulation studies.

2. Problem statement

In this paper, we focus on discrete time, continuous-state, time-invariant system in the general format

$$\mathbf{x}(t+1) = f(\mathbf{x}(t), \mathbf{u}(t)) \tag{1}$$

Where $\mathbf{x} \in \Re^N$ stands for the N-dimensional bounded state vector, $\mathbf{u} \in \Re^M$ stands for the M-dimensional bounded control unit, t stands for the iteration number, $\mathbf{x}(0)$ is given and $f : \Re^N \times \Re^M \to \Re^N$ is a continuously differentiable unknown function. Here, the symmetric boundaries $[-\chi, \chi]$ and $[-\mu, \mu]$ for all components of $\mathbf{x}$ and $\mathbf{u}$ are known. Let p: $\Re^N \to \Re$ and q: $\Re^M \to \Re$ be the two continuously semi-definite negative and differentiable reward functions with the following properties

$$p(\mathbf{x}_1) \leqslant p(\mathbf{x}_2) \Leftrightarrow \|\mathbf{x}_1\| \geqslant \|\mathbf{x}_2\|, p(\mathbf{0}) = 0 \tag{2}$$

$$q(\mathbf{u}_1) \leqslant q(\mathbf{u}_2) \Leftrightarrow \|\mathbf{u}_1\| \geqslant \|\mathbf{u}_2\|, q(\mathbf{0}) = 0 \tag{3}$$

where $\|\mathbf{x}\|$ denotes the second norm of $\mathbf{x}$. The main objective is to learn the control unit $\mathbf{u}$ such that

$$\mathbf{x}(t) \to \mathbf{0}, \mathbf{u}(t) \to \mathbf{0} \text{ as } t \to \infty \tag{4}$$

To formulate a control or learning problem, we convert the objective in (4) into a more formal control problem with discount factor $0 < \gamma \to 1$ [39]

$$J(\mathbf{x}_0 = \sum_{t=0}^{\infty} (p(\mathbf{x}(t)) + q(\mathbf{u}(t))) \tag{5}$$

Thus, the goal is to optimize $J(\mathbf{x}_0)$. The function $J(\mathbf{x})$ defined in (5) is called the state value function [29]. Since f is unknown, in the model-based approach, the intermediate goal is to find the approximated $\hat{f}$ such that with the predicted state vector

$$\hat{\mathbf{x}}(t+1) = \hat{f}(\mathbf{x}(t), \mathbf{u}(t)) \tag{6}$$

the identification error

$$e(t) = \|\mathbf{x}(t) - \hat{\mathbf{x}}(t)\| \tag{7}$$

approaches 0 as $t \to \infty$.

3. Learning the near-optimal control

3.1. Linear system

In the linear system

$$\mathbf{x}(t+1) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \tag{8}$$

in which $\mathbf{B}$ is a known $N \times M$ and $\mathbf{A}$ is an unknown $N \times N$ matrix. Suppose that the reward functions are $p(\mathbf{x}) = -\mathbf{x}^T\mathbf{Q}\mathbf{x}$ and $q(\mathbf{u}) = -\mathbf{u}^T\mathbf{R}\mathbf{u}$, where $\mathbf{Q}$ and $\mathbf{R}$ are positive-definite matrices. To compute control vector $\mathbf{u}$, we find the solution $\mathbf{P}$ of the Riccati equation [40]

$$\mathbf{A}^T\mathbf{P}\mathbf{A} - \mathbf{P} - \mathbf{A}^T\mathbf{P}\mathbf{B}(\mathbf{B}^T\mathbf{P}\mathbf{B} + \mathbf{R})^{-1}\mathbf{B}^T\mathbf{P}\mathbf{A} + \mathbf{Q} = 0 \tag{9}$$

We use DARE algorithm implemented by Arnold et al [41] to solve for $\mathbf{P}$. At each iteration, by replacing $\mathbf{A}$ by the approximator $\hat{\mathbf{A}}(t)$ in (9) and solution $\hat{\mathbf{P}}(t)$, we compute the control vector $\mathbf{u}(t)$ by

$$\mathbf{u}(t) = -(\mathbf{R} + \mathbf{B}^T\hat{\mathbf{P}}(t)\mathbf{B})^{-1}\mathbf{B}^T\hat{\mathbf{P}}(t)\hat{\mathbf{A}}(t)\mathbf{x}(t) \tag{10}$$

To find the approximator $\hat{\mathbf{P}}(t)$, we could apply the techniques in [36].

3.2. Nonlinear system

Theoretically, the solution for the nonlinear control system described from (1)-(5) is the solution of the corresponding HJB equation [27]. Since in general the closed-form solutions for the nonlinear HJB equations are unknown and we know the boundary of the state and control vectors, we discretize the state and control vector to construct an MDP problem closed to the underlying nonlinear function. We use the solution of the MDP problem as the near-optimal solution for the nonlinear system (1)-(5). Since the solution for an MDP problem has been extensively studied, to be brief, we use policy iteration algorithm to compute the optimal policy [29]. In this section, we will focus more on the discretization and set up the MDP process.

3.2.1. Discretizing the state and control vector space

Let K be the number of intervals in each dimension of $\mathbf{x}$ and $\mathbf{u}$ for which we uniformly divide the dimension into small grids. Therefore, the entire state space is divided into K^N small hypercubes and the control space is divided into K^M small hypercubes. All points inside a hypercube are discretely represented by the center of the hypercube. Points on the borders between two hypercubes are represented by the center of the 'left' hypercube. Mathematically, the discretization process is described by the following formulas

$$\mathbf{x}[i] \to \theta_x + \chi/K \forall i \in [1, N] \quad \text{and } \mathbf{x}[i] \in [\theta_x, \theta_x + 2\chi/K) \quad (11)$$

$$\mathbf{u}[i] \to \theta_u + \mu/K \forall i \in [1, N] \quad \text{and } \mathbf{u}[i] \in [\theta_u, \theta_x + 2\mu/K) \quad (12)$$

where $\theta_x \in \{-\chi, -\chi+2\chi/K, -\chi+4\chi/K, ..., \chi-2\chi/K\}$ and $\theta_u \in \{-\mu, -\mu+2\mu/K, -\mu+4\mu/K, ..., \mu-2\mu/K\}$, which are the 'left' boundaries in the hyper cubes.

Let $\delta = \max\{2\chi/K, 2\mu/K\}$. It is easy to see that inside each small hypercube, the largest distance between any two points, or the 'main diagonal', is bounded by

$$\sqrt{\delta^2 + \delta^2 + ... + \delta^2} = \sqrt{N\delta^2} = \sqrt{N}\delta \quad (13)$$

in the state space and by $\sqrt{M}\delta$ in the control space. The left side of (13) has N terms for $\mathbf{x}$ dimension or M terms for $\mathbf{u}$ dimension. Trivially, $K \to \infty \Leftrightarrow \delta \to 0$, which means that the discretization is more precise.

From this point, for any state vector $\mathbf{x}$, we denote $\mathbf{x}_{\text{dis}}$ as the discretized form of $\mathbf{x}$; for any control vector $\mathbf{u}$, we denote $\mathbf{u}_{\text{dis}}$ the discretized form of $\mathbf{u}$. We also denote $(\mathbf{x}_{\text{dis}})$ and $(\mathbf{u}_{\text{dis}})$ as the hypercube where every discretization of $\mathbf{x}$ and $\mathbf{u}$ is $\mathbf{x}_{\text{dis}}$ and $\mathbf{u}_{\text{dis}}$, correspondingly. Formally, from 11 and 12, we have

$$(\mathbf{x}_{\text{dis}}) = [\mathbf{x}_{\text{dis}}(i) - \chi/K, \mathbf{x}_{\text{dis}}(i) + \chi/K] \forall i \in \{1...N\} \quad (14)$$

and

$$(\mathbf{u}_{\text{dis}}) = [\mathbf{u}_{\text{dis}}(i) - \mu/K, \mathbf{u}_{\text{dis}}(i) + \mu/K] \forall i \in \{1...M\} \quad (15)$$

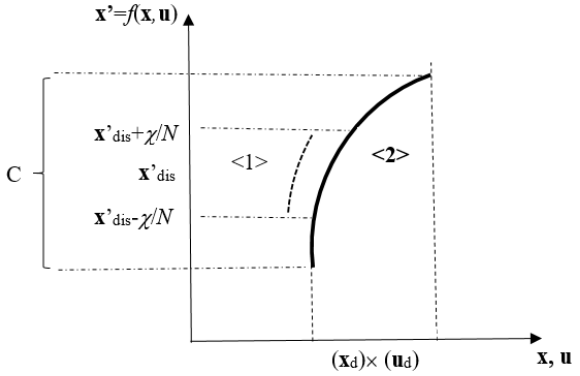

Fig. 1. An example of (16) in one-dimension state space. <1>, the dash surface, is the numerator in (16). <2>, the bold surface, is the denominator of (16).

3.2.2. Setting up the state transition matrix for the MDP problem

The state transition matrix for the MDP problem, which contains all conditional probability $\mathrm{P}(\mathbf{x'}_{\text{dis}}|\mathbf{x}_{\text{dis}}, \mathbf{u}_{\text{dis}})$, has the dimension of $K^M \times K^N \times K^M$, where $\mathbf{x'}_{\text{dis}}$ denotes the next discrete state reached by executing action $\mathbf{u}_{\text{dis}}$ at state $\mathbf{x}_{\text{dis}}$. Let $\mathbf{x'} = f(\mathbf{x}, \mathbf{u})$ $\mathfrak{R}^N$ stands for the next state vector observed by executing action $\mathbf{u}$ at state $\mathbf{x}$. Then, we denote $\mathbf{x'}_{\text{dis}}$ as the discrete form of $\mathbf{x'}$. It is easy to observe that for each triple $(\mathbf{x'}_{\text{dis}}, \mathbf{x}_{\text{dis}}, \mathbf{u}_{\text{dis}})$ the conditional probability

$$\mathrm{P}(\mathbf{x'}_{\text{dis}}|\mathbf{x}_{\text{dis}}, \mathbf{u}_{\text{dis}}) = \frac{\iiint_{(\mathbf{x}_{\text{dis}})\times(\mathbf{u}_{\text{dis}})\times(\mathbf{x'}_{\text{dis}})} d\mathbf{x}d\mathbf{u}d\mathbf{x'}}{\iiint_{(\mathbf{x}_{\text{dis}})\times(\mathbf{u}_{\text{dis}})\times \mathrm{C}} d\mathbf{x}d\mathbf{u}d\mathbf{x'}} \quad (16)$$

where C is the subspace containing all possible value of $f(\mathbf{x}, \mathbf{u}) \forall \mathbf{x}, \mathbf{u} \in (\mathbf{x}_{\text{dis}}) \times (\mathbf{u}_{\text{dis}})$. In our problem statement, since f is unknown, we replace f by $\hat{f}$, which is approximated by the neural network. Figure 1 illustrates a simple case of this conditional probability when $N = 1$. Although the integral could be approximated by the Monte Carlo method [43], the simpler method to approximate $\mathrm{P}(\mathbf{x'}_{\text{dis}}|\mathbf{x}_{\text{dis}}, \mathbf{u}_{\text{dis}})$ is as follow.

- Generate a large number of S points $(\mathbf{x}, \mathbf{u})$ following the uniform distribution in $(\mathbf{x}_{\text{dis}}) \times (\mathbf{u}_{\text{dis}})$. Here, we emphasize that the computation of $\mathrm{P}(\mathbf{x'}_{\text{dis}}|\mathbf{x}_{\text{dis}}, \mathbf{u}_{\text{dis}})$ does not use any sample $(\mathbf{x}(t), \mathbf{u}(t))$. These S points are randomly generated without any prior knowledge of the model to avoid bias.
- Count the number of points T such that $\hat{f}(\mathbf{x}, \mathbf{u}) \in (\mathbf{x'}_{\text{dis}})$.
- Then $T/S \to \mathrm{P}(\mathbf{x'}_{\text{dis}}|\mathbf{x}_{\text{dis}}, \mathbf{u}_{\text{dis}})$ when $S \to \infty$.

3.2.3. State value function in MDP problem

In (5), from Bellman's principle of optimality [44], for the solution $\mathbf{u}(t)$ of the HJB equation (1)-(5), we have

$$J(\mathbf{x}(t)) = p(\mathbf{x}(t)) + q(\mathbf{u}(t)) + \sum_{\tau=t+1}^{\infty} \gamma^{\tau}(p(\mathbf{x}(\tau)) + q(\mathbf{u}(\tau))) = p(\mathbf{x}(t)) + q(\mathbf{u}(t)) + J(\mathbf{x}(t+1)) \quad (17)$$

Because f is stable at the origin, from (2) and (3), $\mathbf{J}(0) = 0$. Since the state value function in the HJB equation (1)-(5) contains a discount factor, we define the corresponding value function in the MDP as

$$R(\mathbf{x}_{\text{dis}}(t)) = p(\mathbf{x}_{\text{dis}}(t)) + q(\mathbf{x}_{\text{dis}}(t)) + \gamma \sum_{\forall \mathbf{x'}_{\text{dis}}} \mathrm{P}(\mathbf{x'}_{\text{dis}} | \mathbf{x}_{\text{dis}}, \mathbf{u}_{\text{dis}}) R(\mathbf{x'}_{\text{dis}}(t+1)) \quad (18)$$

And $R(\mathbf{x}_{\text{dis}}) = 0$ if $(\mathbf{x}_{\text{dis}})$ contains 0 or has 0 on the boundary.

4. Analysis of the discretized MDP for near optimal nonlinear control

In this section, we examine several conditions for the trajectory of discrete state and control obtained by the discretized MDP method, denoted as $\mathbf{x}_{\text{MDP}}(t)$ and $\mathbf{u}_{\text{MDP}}(t)$, converge to $\mathbf{x}(t)$ and $\mathbf{u}(t)$ when $t \to \infty$. More specifically, we answer the following questions. First, suppose that we know an admissible control $\mathbf{u}(t) = g(\mathbf{x}(t))$ and discretize this admissible control (without the MDP policy iteration algorithm), what is the boundary of $|\mathbf{x}(t)\mathbf{x}_{\text{MDP}}(t)|$? In the long term, at any time t, if the discrete state (computed or sampled by the MDP) could be closed to the real state (computed by the real system), then the MDP solution will be useful to control the real system. Second, without any knowledge of the admissible control, in which condition the MDP solution could near-optimally stabilize the system? To simplify the analysis, in this section, we assume that f is known. Although this assumption is against our initial problem statement, this assumption is logical given that the neural network, as the functional approximator $\hat{h}$, could approximate any arbitrary function given sufficient training sample [23, 37, 38].

4.1. The autonomous system

When we linearize an autonomous system using Taylor series expansion

$$\mathbf{x}(t+1) = f(\mathbf{x}(t)) \quad (19)$$

at point $\mathbf{p}$ in the domain of f, we have

$$f(\mathbf{x}) \approx f(\mathbf{p}) + \mathbf{M}(\mathbf{x} - \mathbf{p}) \quad (20)$$

where $\mathbf{M}$ is the matrix of partial derivative of f on $\mathbf{x}$ at $\mathbf{p}$

$$\mathbf{M} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}\Big|_{\mathbf{x}=\mathbf{p}} & \frac{\partial f_1}{\partial x_2}\Big|_{\mathbf{x}=\mathbf{p}} & \cdots & \frac{\partial f_1}{\partial x_n}\Big|_{\mathbf{x}=\mathbf{p}} \\ \frac{\partial f_2}{\partial x_1}\Big|_{\mathbf{x}=\mathbf{p}} & \frac{\partial f_2}{\partial x_2}\Big|_{\mathbf{x}=\mathbf{p}} & \cdots & \frac{\partial f_2}{\partial x_n}\Big|_{\mathbf{x}=\mathbf{p}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1}\Big|_{\mathbf{x}=\mathbf{p}} & \frac{\partial f_n}{\partial x_2}\Big|_{\mathbf{x}=\mathbf{p}} & \cdots & \frac{\partial f_n}{\partial x_n}\Big|_{\mathbf{x}=\mathbf{p}} \end{bmatrix} \quad (21)$$

In this section, we will refer $\mathbf{M}$ as the partial derivative matrix and general and $\mathbf{M}_{\mathbf{x}}$, where the state stands at the subscript, as the partial derivative matrix at a specific state $\mathbf{x}$.

Suppose that at time t, region $(\mathbf{x}_{\text{MDP}}(t))$ contains $\mathbf{x}(t)$ as showed in (11). Let C_{η} be the set of all $\mathbf{x}(t+\eta)$ computed by tracking all points in $(\mathbf{x}_{\text{MDP}})$ on f after η time points. Obviously C_{η} has to be a close region because it is spanned from a close region by a continuous function. Therefore, there exists two points $\mathbf{x}_1(t+\eta)$ and $\mathbf{x}_2(t+\eta)$ such that $|\mathbf{x}_1(t+\eta) - \mathbf{x}_2(t+\eta)|$ is the maximum for all pairs of points in C_{η}. There must exist two chains: $\mathbf{x}_1(t), \mathbf{x}_1(t+1), ..., \mathbf{x}_1(t+\eta-1)$ and $\mathbf{x}_2(t), \mathbf{x}_2(t+1), ..., \mathbf{x}_2(t+\eta-1)$ such that $\mathbf{x}_1(t+\eta) = f(\mathbf{x}_1(t+\eta-1)) = ... = f^n(\mathbf{x}_1(t))$ and $\mathbf{x}_2(t+\eta) = f(\mathbf{x}_2(t+\eta-1)) = ... = f^n(\mathbf{x}_2(t))$. Applying the Taylor series expansion, we have

$$\mathbf{x}_1(t+\eta) - \mathbf{x}_2(t+\eta) = f^n(\mathbf{x}_1(t)) - f^n(\mathbf{x}_2(t)) = \frac{\partial f^n}{\partial \mathbf{x}_2(t)}(\mathbf{x}_1(t) - \mathbf{x}_2(t)) + O(\delta^2) \quad (22)$$

Applying the derivative chain rule for $\frac{\partial f^n}{\partial \mathbf{x}_2(t)}(\mathbf{x}_1(t) - \mathbf{x}_2(t))$, we have

$$\mathbf{x}_1(t+\eta) - \mathbf{x}_2(t+\eta) = \frac{\partial f}{\partial(\mathbf{x}_2(t+\eta-1))} \times$$

$$\frac{\partial f}{\partial(\mathbf{x}_2(t+\eta-2))} \times ... \times \frac{\partial f}{\partial(\mathbf{x}_2(t))}(\mathbf{x}_1(t) - \mathbf{x}_2(t)) + O(\delta^2) \quad (23)$$

Therefore,

$$\|\mathbf{x}_1(t+\eta) - \mathbf{x}_2(t+\eta)\| \leqslant \|\mathbf{M}_{\mathbf{x}_2(t+\eta-1)} \times \mathbf{M}_{\mathbf{x}_2(t+\eta-2)} \times ... \times \mathbf{M}_{\mathbf{x}_2(t)}(\mathbf{x}_1(t) - \mathbf{x}_2(t))\| \quad (24)$$

where each matrix $\mathbf{M}$ is setup according to (21). From (21)-(24), we have the following necessary conditions for the $\mathbf{x}_{\text{MDP}}(t+\eta)$ approaches to $\mathbf{x}(t+\eta)$

1. If all matrices $\mathbf{M}$ generated by (21) have no eigenvalue outside the unit circle on the complex plane, then $\mathbf{x}_{\text{MDP}}(t+\eta)$ approaches to $\mathbf{x}(t+\eta)$ as $K \to \infty$.

The proof is as follow. Let λ be the most prominent eigenvalue of all matrices $\mathbf{M}$ with the largest magnitude. Then from (24)

$$\begin{aligned}\|\mathbf{x}_1(t+\eta) - \mathbf{x}_2(t+\eta)\| &\leqslant \|\mathbf{M}_{\mathbf{x}_2(t+\eta-1)} \times \mathbf{M}_{\mathbf{x}_2(t+\eta-2)} \\ &\times ... \times \mathbf{M}_{\mathbf{x}_2(t)}(\mathbf{x}_1(t) - \mathbf{x}_2(t))\| \\ &\leqslant \|\lambda\|^{\eta}\|\mathbf{x}_1(t) - \mathbf{x}_2(t)\| \quad (25)\end{aligned}$$

In (13), we showed that the distance between any two points in $(\mathbf{x}_{dis})$ cannot be larger than the 'main diagonal' $\delta\sqrt{N}$. Therefore,

$$\begin{aligned}\|\mathbf{x}_1(t+\eta) - \mathbf{x}_2(t+\eta)\| &\leqslant \|\lambda\|^{\eta}\|\mathbf{x}_1(t) - \mathbf{x}_2(t)\| \\ &\leqslant \|\lambda\|^{\eta}\delta\sqrt{N} \quad (26)\end{aligned}$$

Since $\|\lambda\| < 1, \leqslant \|\lambda\|^{\eta}$ is finite with $\eta \to \infty$. Therefore $K \to \infty \Leftrightarrow \delta^{\eta} \to 0$. From the method we used in constructing the MDP, $\mathbf{x}_{\text{MDP}}(t+\eta)$ also falls in C_{η}. Thus, $\|\mathbf{x}(t+\eta) - \mathbf{x}_{MDP}(t+\eta)\| \leqslant \|\mathbf{x}_1(t+\eta) - \mathbf{x}_2(t+\eta)\|$ will also approaches 0.

2. If the system (19) has an asymptotic equilibrium point **x*** such that the linearized matrix $\mathbf{M}_{\mathbf{x}*}$ has all eigenvalues inside the unit circle of the complex plane, then $\mathbf{x}_{\text{MDP}}(t+\eta)$ approaches to $\mathbf{x}(t+\eta)$ as $K \to \infty$.

The proof is as follow. Since the derivative of f is continuous, there must exist a region C_{ε} with size ε around **x*** such that all of the derivative matrices $\mathbf{M}$ in that region have all eigenvalues within the unit complex circle. Let λ be the eigenvalue with the largest magnitude among these matrices. In addition, since (19) has an asymptotic equilibrium point, after a finite time T, $\mathbf{x}(t)$ must be inside C_{ε}. Then, from (24)

$$\begin{aligned}&\|\mathbf{x}_1(t+\eta) - \mathbf{x}_2(t+\eta)\| \leqslant \|\mathbf{M}_{(\mathbf{x}_2(t+\eta-1))} \\ &\times \mathbf{M}_{(\mathbf{x}_2(t+\eta-2))} \times ... \times \mathbf{M}_{(\mathbf{x}_2(t))}(\mathbf{x}_1(t)\mathbf{x}_2(t))\| \\ &= \|\mathbf{M}_{(\mathbf{x}_2(t+\eta-1))} \times \mathbf{M}_{(\mathbf{x}_2(t+\eta-2))} \times ... \times \mathbf{M}_{(\mathbf{x}_2(T))} \\ &\text{(this has } \eta \text{ factors)} \\ &\times \mathbf{M}_{\mathbf{x}_2(T+1))} \times \mathbf{M}_{\mathbf{x}_2(T+2))} \times ... \times M_{\mathbf{x}_2(t)}(\mathbf{x}_1(t)\mathbf{x}_2(t))\| \\ &\text{(this has } T \text{ factors)} \\ &\leqslant \|\lambda\|^{\eta-T+1} \times \|\lambda_T\| \times \|\lambda_{T-1}\| \times ... \\ &\times \|\lambda_1\| \times \|\mathbf{x}_1(t) - \mathbf{x}_2(t)\| \\ &\leqslant \|\lambda\|^{\eta-T+1} \times \|\lambda_T\| \times \|\lambda_{T-1}\| \times ... \\ &\times \|\lambda_1\| \times \delta\sqrt{N} \quad (27)\end{aligned}$$

Because λ is within the complex unit circle, $\|\lambda\|^{\eta-T+1}$ is finite as $\eta \to \infty$. $\|\lambda\|^{\eta-T+1} \times \|\lambda_T\| \times \|\lambda_{T-1}\| \times ...$ $\times \|\lambda_1\|$ is also finite since T is finite. Therefore, $\|\lambda\|^{\eta-T+1} \times \|\lambda_T\| \times \|\lambda_{T-1}\| \times ...$ $\times \|\lambda_1\| \times \delta\sqrt{N}$ approaches to 0 as $K \to \infty$ (or $\delta \to 0$). From the method we used in constructing the MDP , both $\mathbf{x}_{\text{MDP}}(t+\eta)$ and $\mathbf{x}(t+\eta)$ should be bounded by $\mathbf{x}_1(t+\eta)$ and $\mathbf{x}_2(t+\eta)$, which leads to $\|\mathbf{x}(t+\eta) - \mathbf{x}_{\text{MDP}}(t+\eta)\|$ approaching 0.

3. For a special case: If the system is asymptotically stable at 0 (regardless of the linearization), then $\mathbf{x}_{\text{MDP}}(t+\eta)$ approaches to $\mathbf{x}(t+\eta)$ as $K \to \infty$.

The proof for this statement is relatively simpler. For any discretization threshold δ, we can guarantee that the state $\mathbf{x}(t)$ will fall inside the region $[-\delta, \delta]$ at some finite time T, and remain in $[-\delta, \delta]$ $\forall t > T$. This fact implies that with discretization, the MDP will have an absorbing state specified by the region $[-\delta, \delta]$. In addition, regardless of the starting state $\mathbf{x}(0)$ and $\mathbf{x}_{\text{dis}}(0)$, there must be a path toward the absorbing state/region. Therefore, the MDP will eventually bring $\mathbf{x}_{\text{dis}}(t)$ to the absorbing state after some finite time L. Thus, after $\max(T, L)$, both $\mathbf{x}_{\text{dis}}(t)$ and $\mathbf{x}(t)$ will stay inside $[-\delta, \delta]$. Therefore, $\|\mathbf{x}(t) - \mathbf{x}_{\text{MDP}}(t)\| \leqslant \delta$ as $t \to \infty$.

In Figure 2 and Figure 3, we show some toy examples in the one-dimensional system to demonstrate the first necessary condition. In these figures, $\mathbf{x}_{\text{MDP}}$ is computed from the MDP with sampling method in [45]. The left side is the result of the system

$$x(t+1) = 0.1\sin(x(t)) + e^{(-(x(t))^2)} \quad (28)$$

and the right side is the result of the system

$$x(t+1) = 1.1\sin(x(t)) + e^{(-(x(t))^2)} \quad (29)$$

The state space in both of these systems is [-1.5, 1.5]; the initial $\mathbf{x}(0)$ is 0.5 for both of them; and we discretize the entire state space into $K = 100$ regions. The derivative matrices (21) for systems (28) and (29) are one-dimensional functions $0.1\cos(x) - 2xe^{(-x^2)}$and $1.1\cos(x) - 2xe^{(-x^2)}$, correspondingly. As in Figure 3, where we plot the derivative of (28) and (29) in the domain [-1.5. 1.5], system (28) satisfies the first necessary condition; while system (29) does not. We observe that $\mathbf{x}$ and $\mathbf{x}_{\text{MDP}}$ approach closely to each other in system (29) but not in system (29).

4.2. The non-autonomous system

When we linearize the general system (1) using Taylor series expansion at any point <**x**, **u**> = [**p**, **q**], we have

$$f(\mathbf{x}) \approx f(\mathbf{p}, \mathbf{q}) + \mathbf{M}_{\mathbf{p}}(\mathbf{x} - \mathbf{p}) + \mathbf{M}_{\mathbf{q}}(\mathbf{u} - \mathbf{q}) \quad (30)$$

where $\mathbf{M}_{\mathbf{p}}$ and $\mathbf{M}_{\mathbf{q}}$ are the partial derivative of f at [**p**, **q**]

$$\mathbf{M}_{\mathbf{p}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}\Big|_{\substack{\mathbf{x}=\mathbf{p}\\ \mathbf{u}=\mathbf{q}}} & \frac{\partial f_1}{\partial x_2}\Big|_{\substack{\mathbf{x}=\mathbf{p}\\ \mathbf{u}=\mathbf{q}}} & \cdots & \frac{\partial f_1}{\partial x_n}\Big|_{\substack{\mathbf{x}=\mathbf{p}\\ \mathbf{u}=\mathbf{q}}} \\ \frac{\partial f_2}{\partial x_1}\Big|_{\substack{\mathbf{x}=\mathbf{p}\\ \mathbf{u}=\mathbf{q}}} & \frac{\partial f_2}{\partial x_2}\Big|_{\substack{\mathbf{x}=\mathbf{p}\\ \mathbf{u}=\mathbf{q}}} & \cdots & \frac{\partial f_2}{\partial x_n}\Big|_{\substack{\mathbf{x}=\mathbf{p}\\ \mathbf{u}=\mathbf{q}}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1}\Big|_{\substack{\mathbf{x}=\mathbf{p}\\ \mathbf{u}=\mathbf{q}}} & \frac{\partial f_n}{\partial x_2}\Big|_{\substack{\mathbf{x}=\mathbf{p}\\ \mathbf{u}=\mathbf{q}}} & \cdots & \frac{\partial f_n}{\partial x_n}\Big|_{\substack{\mathbf{x}=\mathbf{p}\\ \mathbf{u}=\mathbf{q}}} \end{bmatrix} \quad (31)$$

and

$$\mathbf{M}_{\mathbf{q}} = \begin{bmatrix} \frac{\partial f_1}{\partial u_1}\Big|_{\substack{\mathbf{x}=\mathbf{p}\\ \mathbf{u}=\mathbf{q}}} & \frac{\partial f_1}{\partial u_2}\Big|_{\substack{\mathbf{x}=\mathbf{p}\\ \mathbf{u}=\mathbf{q}}} & \cdots & \frac{\partial f_1}{\partial u_m}\Big|_{\substack{\mathbf{x}=\mathbf{p}\\ \mathbf{u}=\mathbf{q}}} \\ \frac{\partial f_2}{\partial u_1}\Big|_{\substack{\mathbf{x}=\mathbf{p}\\ \mathbf{u}=\mathbf{q}}} & \frac{\partial f_2}{\partial u_2}\Big|_{\substack{\mathbf{x}=\mathbf{p}\\ \mathbf{u}=\mathbf{q}}} & \cdots & \frac{\partial f_2}{\partial u_m}\Big|_{\substack{\mathbf{x}=\mathbf{p}\\ \mathbf{u}=\mathbf{q}}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial u_1}\Big|_{\substack{\mathbf{x}=\mathbf{p}\\ \mathbf{u}=\mathbf{q}}} & \frac{\partial f_n}{\partial u_2}\Big|_{\substack{\mathbf{x}=\mathbf{p}\\ \mathbf{u}=\mathbf{q}}} & \cdots & \frac{\partial f_n}{\partial u_m}\Big|_{\substack{\mathbf{x}=\mathbf{p}\\ \mathbf{u}=\mathbf{q}}} \end{bmatrix} \quad (32)$$

Similar to the autonomous system, for the close region $([\mathbf{x}_{\text{MDP}}(t), \mathbf{u}_{\text{MDP}}(t)])$ (11), including the boundary, containing [$\mathbf{x}(t)$, $\mathbf{u}(t)$], let C_η be the set of all $\mathbf{x}(t+\eta)$ computed by tracking all points in $([\mathbf{x}_{\text{MDP}}(t), \mathbf{u}_{\text{MDP}}(t)])$ on f after η time points. On the region C_η containing all possible $\mathbf{x}(t+\eta)$, there exists two points $\mathbf{x}_1(t+\eta)$ and $\mathbf{x}_2(t+\eta)$such that $\|\mathbf{x}_1(t+\eta) - \mathbf{x}_2(t+\eta)\|$ is the maximum for all pairs of points in C_η. There must exist two chains: $[\mathbf{x}_1(t), \mathbf{u}_1(t)], [\mathbf{x}_1(t+1), \mathbf{u}_1(t+1)], ..., [\mathbf{x}_1(t+\eta), \mathbf{u}_1(t+\eta)]$ and $[\mathbf{x}_2(t), \mathbf{u}_2(t)], [\mathbf{x}_2(t+1), \mathbf{u}_2(t+1)], ..., [\mathbf{x}_2(t+\eta), \mathbf{u}_2(t+\eta)]$ such that $\mathbf{u}_1(t+\eta) = f(\mathbf{x}_1(t+\eta-1), \mathbf{u}_1(t+\eta-1)) = f(f(\mathbf{x}_1(t+\eta-2), \mathbf{u}_1(t+\eta-2))) = ... = f^\eta(\mathbf{x}_1(t), \mathbf{u}_1(t))$ and $\mathbf{u}_2(t+\eta) = f(\mathbf{x}_2(t+\eta-1), \mathbf{u}_2(t+\eta-1)) = f(f(\mathbf{x}_2(t+\eta-2), \mathbf{u}_2(t+\eta-2))) = ... = f^\eta(\mathbf{x}_2(t), \mathbf{u}_2(t))$. Applying the Taylor series expansion, we have

$$\begin{aligned} &\mathbf{x}_1(t+\eta) - \mathbf{x}_2(t+\eta) = \frac{\partial f}{\partial(\mathbf{x}_2(t+\eta-1)), \mathbf{u}_2(t+\eta-1))} \times \\ &([\mathbf{x}_1(t+\eta-1), \mathbf{u}_1(t+\eta-1)] - [\mathbf{x}_2(t+\eta-1), \mathbf{u}_2(t+\eta-1)]) + O(\delta^2) \\ &\quad = \mathbf{M}_{\mathbf{p},\mathbf{x}_2(t+\eta-1)}(\mathbf{x}_1(t+\eta-1) - \mathbf{x}_2(t+\eta-1)) + \mathbf{M}_{\mathbf{q},\mathbf{u}_2(t+\eta-1)}(\mathbf{u}_1(t+\eta-1) - \mathbf{u}_2(t+\eta-1)) \quad (33) \end{aligned}$$

where $\mathbf{M}_{\mathbf{p},\mathbf{x}_2}$ and $\mathbf{M}_{\mathbf{q},\mathbf{u}_2}$ are the $\mathbf{M}_{\mathbf{p}}$ (31) and $\mathbf{M}_{\mathbf{q}}$ (32) at $[\mathbf{x}_2, \mathbf{u}_2]$, respectively.

Suppose that we have an arbitrary control law $\mathbf{u} = k(\mathbf{x})$. Taking the derivative of the control rule, we have $\Delta\mathbf{u} = \mathbf{M}_k \Delta\mathbf{x}$ such that

$$\mathbf{M}_k = \begin{bmatrix} \frac{\partial k_1}{\partial x_1}\Big|_{\mathbf{x}=\mathbf{p}} & \frac{\partial k_1}{\partial x_2}\Big|_{\mathbf{x}=\mathbf{p}} & \cdots & \frac{\partial k_1}{\partial x_n}\Big|_{\mathbf{x}=\mathbf{p}} \\ \frac{\partial k_2}{\partial x_1}\Big|_{\mathbf{x}=\mathbf{p}} & \frac{\partial k_2}{\partial x_2}\Big|_{\mathbf{x}=\mathbf{p}} & \cdots & \frac{\partial k_2}{\partial x_n}\Big|_{\mathbf{x}=\mathbf{p}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial k_m}{\partial x_1}\Big|_{\mathbf{x}=\mathbf{p}} & \frac{\partial k_m}{\partial x_2}\Big|_{\mathbf{x}=\mathbf{p}} & \cdots & \frac{\partial k_m}{\partial x_n}\Big|_{\mathbf{x}=\mathbf{p}} \end{bmatrix} \quad (34)$$

For any state $\mathbf{x}$, we denote $\mathbf{M}_{k\mathbf{x}}$ as the specific $\mathbf{M}_k$ matrix at state $\mathbf{x}$. Substitute (34) to (33), we have

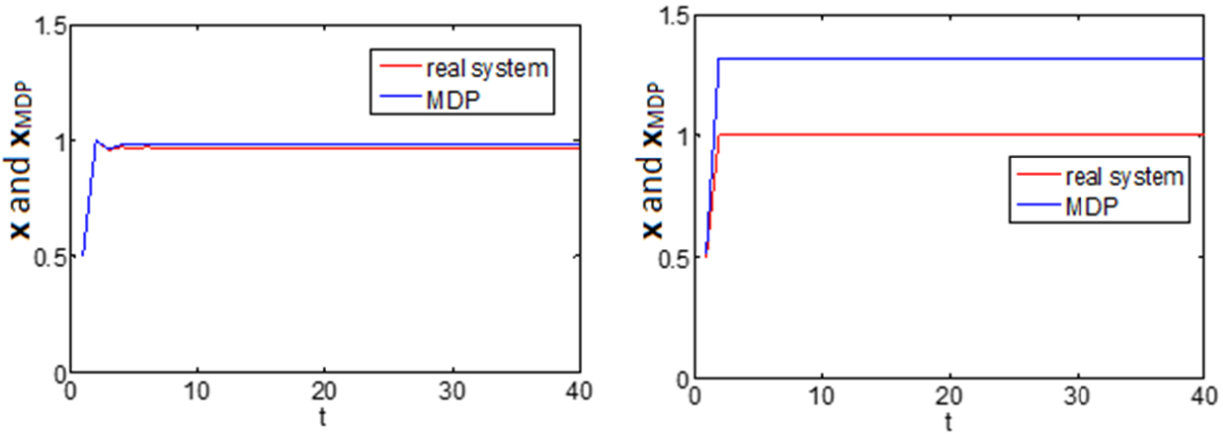

Fig. 2. The closeness between $\mathbf{x}$(real system) and $\mathbf{x}_{\text{MDP}}$ (MDP). The left figure corresponds to system (28). The right figure corresponds to system (29)

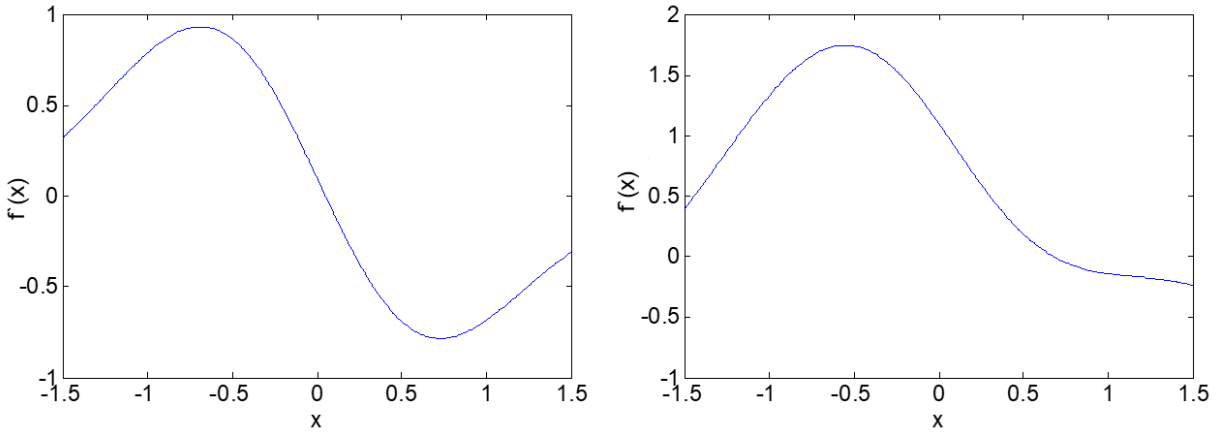

Fig. 3. Derivative $\partial f/\partial x$ in system (28) on the left and system (29) on the right.

$$\begin{aligned}\|\mathbf{x}_1(t+\eta)-\mathbf{x}_2(t+\eta)\| &= \|\mathbf{M}_{\mathbf{p},\mathbf{x}_2(t+\eta-1)}(\mathbf{x}_1(t+\eta-1)-\mathbf{x}_2(t+\eta-1))+\mathbf{M}_{\mathbf{q},\mathbf{u}_2(t+\eta-1)}(\mathbf{u}_1(t+\eta-1)-\mathbf{u}_2(t+\eta-1))\| \\ &\leqslant \|(\mathbf{M}_{\mathbf{p},\mathbf{x}_2(t+\eta-1)}+\mathbf{M}_{\mathbf{q},\mathbf{u}_2(t+\eta-1)}\mathbf{M}_{k,\mathbf{x}_2(t+\eta-1)})(\mathbf{x}_1(t+\eta-1)-\mathbf{x}_2(t+\eta-1))\| \quad (35)\end{aligned}$$

Recursively applying the derivative chain rule on $(\mathbf{x}_1(t+\eta-1)-\mathbf{x}_2(t+\eta-1))$ until $[\mathbf{x}(t),\mathbf{u}(t)]$, with the same argument from (33) to (35), we have

$$\begin{aligned}&\|\mathbf{x}_1(t+\eta)-\mathbf{x}_2(t+\eta)\| \leqslant \\ &\|(\mathbf{M}_{\mathbf{p},\mathbf{x}_2(t+\eta-1)}+\mathbf{M}_{\mathbf{q},\mathbf{u}_2(t+\eta-1)}\mathbf{M}_{k,\mathbf{x}_2(t+\eta-1)})\times \\ &(\mathbf{M}_{\mathbf{p},\mathbf{x}_2(t+\eta-2)}+\mathbf{M}_{\mathbf{q},\mathbf{u}_2(t+\eta-2)}\mathbf{M}_{k,\mathbf{x}_2(t+\eta-2)})\times\ldots \\ &\times(\mathbf{M}_{\mathbf{p},\mathbf{x}_2(t+1)}+\mathbf{M}_{\mathbf{q},\mathbf{u}_2(t+1)}\mathbf{M}_{k,\mathbf{x}_2(t+1)}) \\ &(\mathbf{x}_1(t)-\mathbf{x}_2(t))\| \quad (36)\end{aligned}$$

From this point, similar to the autonomous system, we have the necessary conditions for the $\mathbf{x}_{\text{MDP}}(t+\eta)$ approaches to $\mathbf{x}(t+\eta)$.

1. If the matrices $\mathbf{M}_\mathbf{p}+\mathbf{M}_\mathbf{q}\mathbf{M}_k$ generated by (31), (32) and (34) have no eigenvalue outside the unit circle on the complex plane, then $\mathbf{x}_{\text{MDP}}(t+\eta)$ approaches to $\mathbf{x}(t+\eta)$ as $\delta\to 0$ with any η.

2. If the system (1) has an asymptotic equilibrium point p such that the linearized matrix $\mathbf{M}_\mathbf{p}+\mathbf{M}_\mathbf{q}\mathbf{M}_k$ at the equilibrium point has all eigenvalues inside the unit circle of the complex plane, then $\mathbf{x}_{\text{MDP}}(t+\eta)$ approaches to $\mathbf{x}(t+\eta)$ as $\delta\to 0$ with any η.

We omit the proof for these two statements since the proof is almost similar to the proof we already showed in the autonomous system section.

In Figure 4, we show some toy examples in one-dimensional system to demonstrate the first necessary condition. Similar to the autonomous system examples, in this figures, $\mathbf{x}_{\text{MDP}}$ is computed from the MDP with sampling method in [45]. The left side is the result of the system

$$x(t+1) = \sin(x(t)) + u(t) \text{ and control law } u(t) = -0.5x(t) \quad (37)$$

and the right side is the result of the system

$$x(t+1) = \sin(x(t)) + u(t) \text{ and control law } u(t) = -2x(t) \quad (38)$$

The state space in both of these systems is [-1, 1]; the initial $\mathbf{x}(0)$ is 0.5 for both of them; and we discretize the entire state space into K = 100 regions. In (37), $\mathbf{M_p} + \mathbf{M_q}\mathbf{M}_k = \cos(x(t)) - 0.5$, which is within [0.0403, 0.5]. Therefore, (37) meets the first necessary condition. In (38), $\mathbf{M_p} + \mathbf{M_q}\mathbf{M}_k = \cos(x(t)) - 2$, which is between [-1.5403, -1]. Therefore, (37) does not meet the necessary condition. As in Figure 4, $\mathbf{x}_{\text{MDP}}(t)$ converges to $\mathbf{x}(t)$ in system (37), but not in system (38).

4.3. The existence of the MDP solution as to near-optimally stabilize the system

In this section, we show the existence of the MDP solution when the system (1) is stable at the equilibrium point. The stability definition is defined as follow: there exist a positive small number ε such that if $\|\mathbf{x}\| < \varepsilon$ then $\|f(\mathbf{x}, \mathbf{0})\| < \varepsilon$. With this assumption, when we choose K such that $\chi/K < \varepsilon$, the MDP will have a special state $\mathbf{x}^*_{\text{MDP}} = \mathbf{0}$ with the following properties:

- The MDP's optimal policy at $\mathbf{x}^*_{\text{MDP}}$ is $\mathbf{u}^*_{\text{MDP}} = \mathbf{0}$.
- The later states in the MDP are also $\mathbf{x}^*_{\text{MDP}}$. The proof of these properties is relatively simple due to the properties of the state and action reward functions in (2) and (3), where the optima are at 0. From this stability assumption of f, we prove the following statements.

1. If the system (1) is stable and the HJB equation (1)-(1) has a finite solution as $\gamma \to 1$, then in the MDP, $\mathbf{x}^*_{\text{MDP}}(t) = 0$ as $t \to \infty$.

The proof of this statement is as follow. If the HJB equation (1)-(5) has a finite solution as $\gamma \to 1$, then the control function $\mathbf{u}(t)$ has to be able to bring $\mathbf{x}(t)$ to 0 in finite time. Otherwise, the state and action rewards are always negative and will approach infinite as $\gamma \to 1$. Since $\mathbf{x}(t)$ is 0 in finite time, there must exist a path in the MDP that can reach $\mathbf{x}^*_{\text{MDP}}$ with positive probability. Obviously one of these paths is the discretization of the HJB's solution $\mathbf{u}(t)$. Since the policy iteration in MDP has been proven to converge to the optimal policy [46], this policy cannot be worse than the policy induced by discretizing the HJB equation's solution. Therefore, in the MDP's optimal policy, there must exist a path from any state to $\mathbf{x}^*_{\text{MDP}}$ with positive probability $\phi > 0$. With infinite number of visit t $\to \infty$, the maximum probability for **not** reaching $\mathbf{x}^*_{\text{MDP}}$ is $(1-\phi)^\infty = 0$.

2. If all $\mathbf{M_p}$ matrices (31) have the most prominent eigenvalues within the unit circle $\forall \mathbf{x}, \mathbf{u}$ and $\mathbf{x}_{\text{MDP}}(t) = \mathbf{0}$ as $t \to \infty$ in the MDP solution for all starting $\mathbf{x}(0)$, then by applying the MDP's control unit $\mathbf{x}_{\text{MDP}}(t)$ on $\mathbf{x}(t)$, $||\mathbf{x}(t)|| \leqslant \delta\sqrt{N}$.

The proof of this statement is as follow. Since we apply $\mathbf{u}_{\text{MDP}}(t)$ for all $\mathbf{x}(t)$ in ($\mathbf{x}_{\text{MDP}}(t)$) region, the difference of the control unit cancels. Thus, the equation (30) becomes

$$f(\mathbf{x}) \approx f(\mathbf{p}, \mathbf{q}) + \mathbf{M_p}(\mathbf{x} - \mathbf{p}) (39) \quad (39)$$

Following the same argument from (31) to (34), we have

$$\|\mathbf{x}_1(t+\eta) - \mathbf{x}_2(t+\eta)\| \leqslant \|(\mathbf{M}_{\mathbf{p},\mathbf{x}_2(t+\eta-1)}) \times (\mathbf{M}_{\mathbf{p},\mathbf{x}_2(t+\eta-2)}) \times ... \times (\mathbf{M}_{\mathbf{p},\mathbf{x}_2(t+1)})(\mathbf{x}_1(t) - \mathbf{x}_2(t))\| \quad (40)$$

Because the most prominent eigenvalues of $\mathbf{M_p}$ are within unit circle, from (40), we have

$$\|\mathbf{x}(t) - \mathbf{x}_{MDP}(t)\| \leqslant \|\mathbf{x}_1(t+\eta) - \mathbf{x}_2(t+\eta)\| \leqslant \|\mathbf{x}_1(t) - \mathbf{x}_2(t)\| \leqslant \delta\sqrt{N} \quad (41)$$

Therefore, if $\mathbf{x}_{\text{MDP}}(t) \to 0$, then$||\mathbf{x}(t)|| \leqslant \delta\sqrt{N}$.

5. Learning control system with selective decentralization approach

5.1. Statement of selective decentralization

Let us rewrite system (1) as

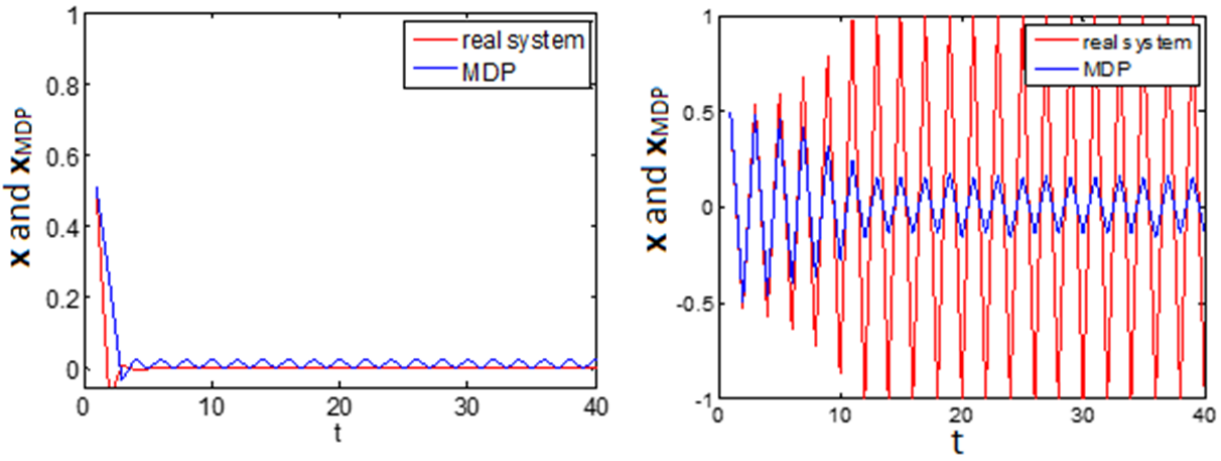

Fig. 4. The closeness between $\mathbf{x}$(real system) and $\mathbf{x}_{\text{MDP}}$ (MDP). The left figure corresponds to system (37). The right figure corresponds to system (38).

$$\Sigma : \boldsymbol{x}(t+1) = f[\boldsymbol{x}(t), \boldsymbol{u}(t), \theta] \tag{42}$$

where θ is an unknown parameter vector in $\Re^N$. In the identification phrase, the intermediate objective is to estimate θ using measurements of the overall system. In the problem of interest to us, the system is assumed to consist of r subsystems of low dimension which are interconnected. However, how these subsystems interconnect is unknown. If the state vectors of the subsystems $\Sigma_1, \Sigma_2, ..., \Sigma_r$ are respectively $\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_r$, it is assumed that each subsystem can be described by the difference equation

$$\Sigma_i : \boldsymbol{x}_i(k+1) = f_i[\boldsymbol{x}_i(k), u_i(k), \theta_i] + \sigma_i[z_i(k)] \tag{43}$$

where the parameter σ_i is assumed to be small, and $[\boldsymbol{x}_i, z_i] = \boldsymbol{x}^T$ (i.e., the elements of z_i are state variables not contained in $\boldsymbol{x}_i$). A decentralized approximated model can be set up as

$$\hat{\boldsymbol{x}}_i(t+1) = \hat{f}_i[\boldsymbol{x}_i(t), z_i(t), \boldsymbol{u}(t), \theta(t)] \tag{44}$$

To be more specific, for the linear system, the decentralized model has the form

$$\hat{\mathbf{A}} = \begin{bmatrix} \widehat{\mathbf{A}_1} & \widehat{\mathbf{a}_{1,2}} & \cdots & \widehat{\mathbf{a}_{1,r}} \\ \widehat{\mathbf{a}_{2,1}} & \widehat{\mathbf{A}_2} & \cdots & \widehat{\mathbf{a}_{2,r}} \\ \vdots & \cdot & \ddots & \vdots \\ \widehat{\mathbf{a}_{r,1}} & \widehat{\mathbf{a}_{r,2}} & \cdots & \widehat{\mathbf{A}_r} \end{bmatrix} \tag{45}$$

where the lower-case $\widehat{\mathbf{a}}$ stands for the estimated communication among the subsystems, which is expected to be minor. The nonlinear decentralized model has the form

$$\widehat{\mathbf{x}}(t+1) = \begin{bmatrix} \widehat{\mathbf{x}_1}(t+1) \\ \widehat{\mathbf{x}_r}(t+1) \\ \vdots \\ \widehat{\mathbf{x}_r}(t+1) \end{bmatrix} = \hat{f}(\mathbf{x}(t), \mathbf{u}(t)) = \begin{bmatrix} \hat{f}_1(\mathbf{x}_1(t), \mathbf{u}_1(t)) \\ \hat{f}_2(\mathbf{x}_2(t), \mathbf{u}_2(t)) \\ \vdots \\ \hat{f}_r(\mathbf{x}_r(t), \mathbf{u}_r(t)) \end{bmatrix} \tag{46}$$

At this stage, the knowledge that each subsystem has about the components of z that affect it becomes important. Here, we assume the unknown decentralization structure: every subsystem Σ_i knows the small set of variables in z_i that might affect its outputs, but does not know exactly which variables do affect them.

Selective decentralization policy: The number of possible decentralization structures for r subsystems is $B(r)$ (the r^{th} Bell's number), which grows super-exponentially. We set up a separate identification model for each such decentralization structure and adaptively switch among the models implementing the different decentralization policies to determine the best model.

Complete decentralization policy: The subsystems perform identification and calculate their local control using their own state and control subspace without any communication. In this work, we mention this naive

approach to compare the control performance with the selective decentralization approach.

In addition, in this paper, we refer *centralized control*, or centralization, as considering the whole system as one component. In this case, $r = 1$ and $B(r) = 1$. The other formulation is the same to decentralization.

5.2. Selective decentralized control framework

Figure 5 shows the design of the learning control system in this work with two phases: identification and control. In the identification phase, we train the neural networks to acquire the functional approximators $\hat{f}$ from using <$\mathbf{x}(t)$, $\mathbf{u}(t)$> as the input tuples and $\mathbf{x}(t+1)$ as the outputs. The details of system identification is omitted in this paper since we have already presented them in [47]. In the control phase, to compute the near-optimal control, we use (9-10) for the linear system, and policy iteration algorithm for the nonlinear system after setting up the corresponding MDP [29]. Here, the window size parameter Ω decides how frequently we call the identification phase. In other words, Ω decides the number of <$\mathbf{x}(t)$, $\mathbf{u}(t)$, $\mathbf{x}(t+1)$> tuples to train $\hat{f}$.

The selective decentralized control examines all of the $B(r)$ connection schemes among the subsystem and uses the scheme with lowest identification error to apply the control algorithm. For example, with $r = 3$, we have $B(r) = 5$ possible decentralization schemes: {{1, 2, 3}}, {{1, 2}, {3}}, {{1, 3}, {2}}, {{1}, {2, 3}} and {{1}, {2}, {3}}, in which each scheme has 1, 2, 2, 2 and 3 subsystem(s), correspondingly. A subsystem only uses its state and control variable to compute its own approximator. For example, in the linear system, with scheme {{1, 2}, {3}}, we have the format $\hat{\mathbf{A}} = \begin{bmatrix} \widehat{\mathbf{A}_{1,2}} & \\ & \widehat{\mathbf{A}_3} \end{bmatrix}$. In this example, $\widehat{\mathbf{A}_{1,2}}(t)$ is computed only using $\mathbf{x}_1(t-1), \mathbf{x}_2(t-1), \mathbf{u}_1(t-1)$ and $\mathbf{u}_2(t-1)$, meanwhile $\widehat{\mathbf{A}_3}(t)$ is computed only using $\mathbf{x}_3(t-1)$ and $\mathbf{u}_3(t-1)$. If scheme {{1, 2}, {3}} returns the lowest identification error, then from (10), we compute the next control $[\mathbf{u}_1(t), \mathbf{u}_2(t)]$ using only $\widehat{\mathbf{A}_{1,2}}(t)$ and $\mathbf{u}_3(t)$ using only $\widehat{\mathbf{A}_3}(t)$.

Let w be the window index. Then the window w covers the discrete time index from $t = (w-1)\Omega + 1$ to $t = w\Omega$. Let $E(w)$ be the window-identification error at window w, which is the average of $e(t)$ from $t = (w-1)\Omega + 1$ to $t = w\Omega$. Let ρ_1 and ρ_2 be two small numbers for thresholding. The pseudo code for selective decentralization is as follow:

initialize b: the best decentralization scheme
  need_iden = **true** // whether or not execute identification
for w from 1 to the maximum window index
  calculate control policy using b
    (using (9-10) for linear system)
    (using (11-18) for nonlinear system)
  if *need_iden* = **true**
    Train approximator and compute $E(w)$ for $B(r)$ decentralization schemes
  end if
  Select the scheme with the lowest $E(w)$ as b
  if $w > 1$ and ($E(w) < \rho_1$
    or $|E(w) - E(w-1)|/|E(w)| < \rho_2$)
    if *need_iden* = **true**
  end if
end for

6. Simulation results

6.1. Linear system

In this simulation, we setup a system of 8-dimension state and control variables with $r = 4$. The unknown transitional block matrix A is setup with real subsystem components {{1,2}, {3, 4},{5, 6}, {7,8}} as follow. With raw matrix $\tilde{\mathbf{A}}$ matrix as

$$\tilde{\mathbf{A}} = \begin{bmatrix} 0.7 & 0.3 & & & & & & \\ 0.2 & 0.8 & & & & & & \\ & & 0.23 & 0.77 & & & \psi & \\ & & 0.4 & 0.6 & & & & \\ & \psi & & & 0.5 & 0.5 & & \\ & & & & 0.35 & 0.65 & & \\ & & & & & & 0.9 & 0.1 \\ & & & & & & 0.15 & 0.85 \end{bmatrix} \tag{47}$$

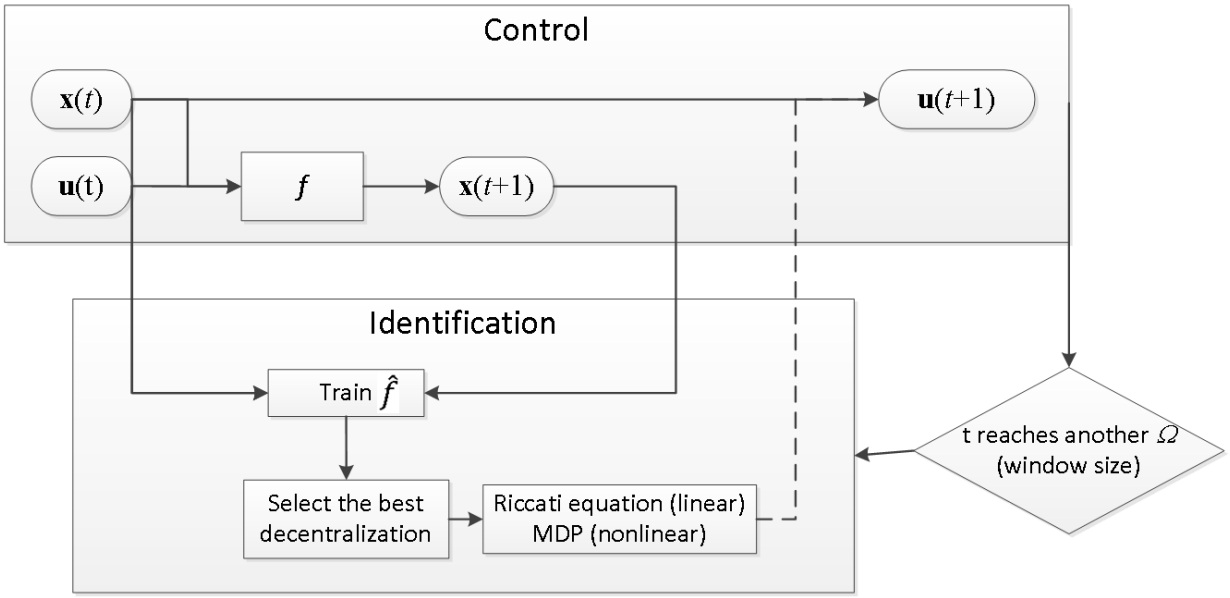

Fig. 5. The learning design for selective decentralized control.

where the non-block entries of $\tilde{\mathbf{A}}$ are a random numbers between 0 and ψ. To avoid numerical overflowing, we normalize $\tilde{\mathbf{A}}$ into $\mathbf{A}$ such that $\mathbf{A}$ is a Markov matrix. The reward functions are $p(\mathbf{x}) = -\mathbf{x}^T\mathbf{x}$ and $q(u) = -\mathbf{u}^T\mathbf{u}$. The discount reward factor in (5) is γ = 0.9. The control algorithm could be referred to (9) and (10). As shown in (47), ψ decides the interconnection strength among system components. We call ψ coupling parameter. We setup the completely decoupled system by setting ψ = 0 and the strongly coupled system by ψ = 0.1. We set $\mathbf{B}$ as the identity matrix. For identification, we set the learning rate in [36] as 0.5. At the starting point, we set $\mathbf{x}(0)$ and $\mathbf{u}(0)$ as vectors of 1. Because calculating $\hat{\mathbf{A}}$ [36, 47] is relatively simple, we set the window size Ω = 1. We run the experiment for at most 1000 iteration. We repeat this setup 100 times since A and $\mathbf{x}(0)$ contains random parameters and report the mean statistics.

In Figures 6 and 7, we observe that the selectively decentralized system shows better control performance than the completely decentralized system and the centralized system. In these figures, we draw the y-axis in log scale since $\mathbf{x}$ converges to 0 so quickly that the linear-scale plot could not show the difference. We use norm(x) to denote the second-norm, or trajectory, of $\mathbf{x}$. Clearly, after more than 30 iterations, $\mathbf{x}$ in the selectively decentralized system converge to 0 faster than they are in the completely decentralized system and the centralized system. At the first few iterations, the selectively decentralized system shows slightly poorer

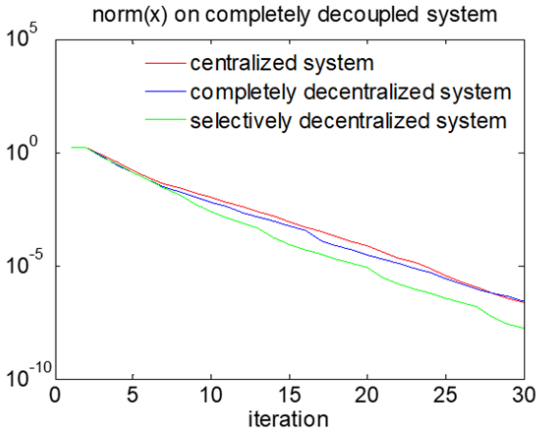

Fig. 6. Comparison of control performance among the centralized system, the completely decentralized system and the selectively decentralized system when the system is linear and completely decoupled.

control performance. This may due to the complexity of the selectively decentralized system in identifying unknown $\mathbf{A}$. In the other hands, as the systems are more coupled, we see that the performance gap between the decentralized systems and the centralized system is less.

6.2. *Nonlinear system*

In this example, we choose the system

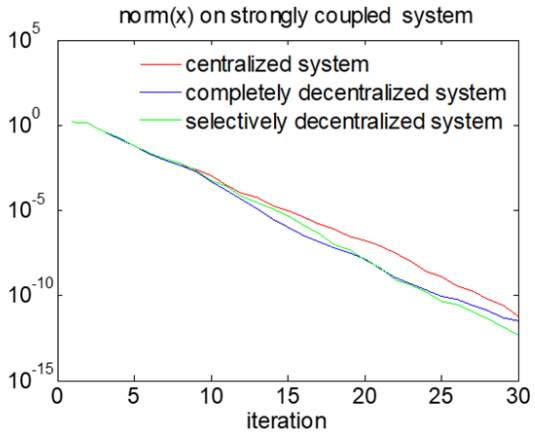

Fig. 7. Comparison of control performance among the centralized system, the completely decentralized system and the selectively decentralized system when the system is linear and strongly coupled.

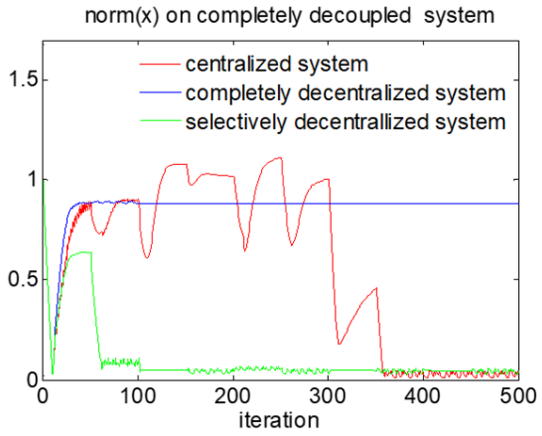

Fig. 8. Comparison of control performance among the centralized system, the completely decentralized system and the selectively decentralized system when the system (48) is completely decoupled.

$$\mathbf{x}(t+1) = sin(\mathbf{A}\mathbf{x}(t) + \mathbf{u}(t)) \tag{48}$$

where $\mathbf{x}, \mathbf{u} \in \Re^4$, matrix $\mathbf{A}$ is defined by normalizing $\tilde{\mathbf{A}}$ into a Markov matrix where

$$\tilde{\mathbf{A}} = \begin{bmatrix} 0.7 & 0.3 & & \psi \\ 0.2 & 0.8 & & \\ & & 1 & \\ \psi & & & 1 \end{bmatrix} \tag{49}$$

and the *sin* function is defined as

$$sin(\mathbf{x}) = \begin{bmatrix} \sin(x_1) \\ \sin(x_2) \\ \vdots \\ \sin(x_n) \end{bmatrix} \tag{50}$$

and $\mathbf{x}(0) = 0.2$. Here, we assume that the boundary of $\mathbf{x}$ an $\mathbf{u}$ is known as $-0.2 \leqslant x_i, u_i \leqslant 0.2 \forall i \in [1,4]$ and the real subsystem component in (1) is {{1,2}, {3}, {4}}. The reward functions are $p(\mathbf{x}) = -\mathbf{x}^T\mathbf{x}$ and $q(u) = -uTu$. The discount reward factor in (5) is $\gamma = 0.9$.

For system approximation, we use a three-layer neural network with 30 hidden units, sigmoid activation function, and backpropagation to train the neural network for $\hat{f}$. For each training step, we pass the training sample set $< \mathbf{x}(t), \mathbf{u}(t) > 2000$ times. We set window size ïĂăΩ = 50 (figure 1). In addition, we run the experiment for at most 10000 iteration. Similar to the linear system case study, we setup the completely decoupled system by setting $\psi = 0$ and the strongly coupled system by $\psi = 0.1$. In each state and control vector dimension, we divide the dimension into $K = 8$ regions, which makes the resolution threshold (13) 0.05.

In Figures 8 and 9, we observe that the selectively decentralized system shows better control performance than the completely decentralized system and the centralized system. Similar to figures 2 and 3, we use norm(x) to denote the second-norm of $\mathbf{x}$. For the ease of visualization, we only draw the result up to the 500^{th} iteration, when the selective decentralization is showed to converge. Here, we observe that when the system is completely decoupled, the centralized system converges to 0 significantly slower than the selectively decentralized system does. Surprisingly, the completely decentralized system does not converge within the maximum number of iterations in our experiment. In addition, when the system is strongly coupled, both the completely decentralized system and the centralized system fail to control within the maximum number of iterations in our experiment.

6.3. *Comparison between the discretized-MDP approach and TD-learning*

In figure 10, we compare the learning performance among the Discretized-MDP, Adaptive Dynamic Programming (ADP) [48–50] and Q-learning [51]. Q-learning is one of the most well-known techniques in reinforcement learning, following the temporal-difference (TD) principles [52]. ADP, which is one of the most promising approaches aiming for online learning, has been proven to stabilize the nonlinear system in feedback linearization form. The example used in this section is

$$\mathbf{x}(t+1) = sin(\mathbf{A}\mathbf{x}(t)) + \mathbf{u}(t) \tag{51}$$

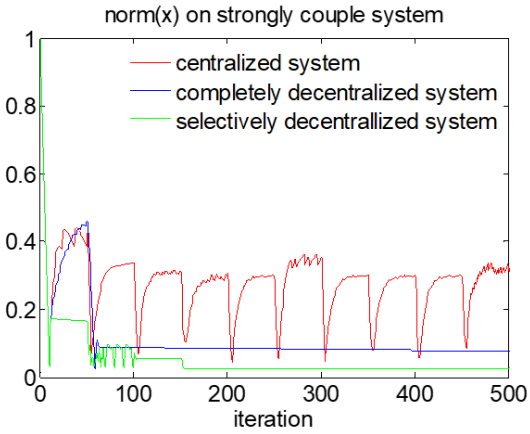

Fig. 9. Comparison of control performance among the centralized system, the completely decentralized system and the selectively decentralized system when the system (48) is strongly coupled.

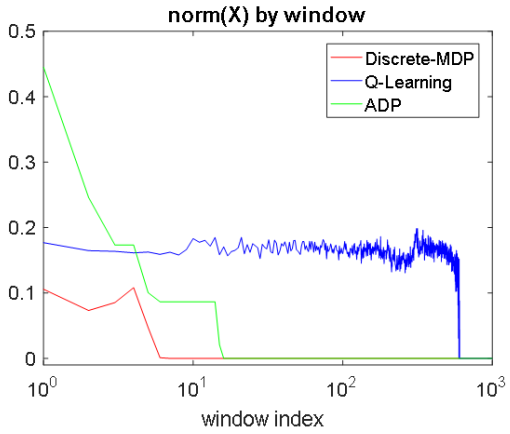

Fig. 10. Comparison of control performance among the Discretized-MDP approach, the ADP approach and the Q-learning approach

where $\mathbf{x}, \mathbf{u} \in \Re^3$ and matrix $\mathbf{A}$ is

$$\mathbf{A} = \begin{bmatrix} 0.6 & 0.2 & 0.2 \\ 0.2 & 0.6 & 0.2 \\ 0.2 & 0.2 & 0.6 \end{bmatrix} \tag{52}$$

In addition, we experimented these approaches without any decentralization. The starting state $\mathbf{x}(0)$ is [0.5, 0.5, 0.5] for all experiments. We show the implementation details for Q-learning as in [53]. The implementation for ADP is accordant to [48]. For discretization of both the discretized-MDP and the Q-learning approaches, we make the resolution threshold (13) 0.05. We observe that these techniques could stabilize the system; however, the ADP and discretized-MDP approaches are significantly superior to the Q-learning approach. The discretized-MDP approach also stabilizes the system faster than the ADP approach.

There are several points to note in figure 10. First, since the difference of converging time in these approaches could be exponential, we draw the x-axis, which stands for converging time measured by the number of windows, in log scale. Therefore, the state trajectory (second norm of $\mathbf{x}$) may neither be smooth nor seem differentiable. Second, since the Q-learning performance in [53] is measured by the average state trajectory over a window, the x-axis unit figure 10 is the window index, with window size $\Omega = 50$. Therefore, the lines in figure 10 show the average of state-trajectory over each window.

7. Conclusions

In this paper, we show that selective decentralization can improve the learning performance in both linear and nonlinear systems with several levels of interconnection among subsystems. Here, we measure the performance on the number of iterations, or samples, needed in learning. This measurement of performance is useful for problems in which the number of training samples is limited. In addition, we show that the discrete-MDP technique could help in learning nonlinear control problem in general form.

Compared to adaptive dynamic programming (ADP) [49, 54–59], which is one of the most interested approaches in reinforcement learning and adaptive control in the recent years, our discrete-MDP approach is more limited in utilizing the capability of neural networks. In the ADP approach, the neural networks are used to approximate both the control function $\mathbf{u} = k(\mathbf{x})$ and the state utility function $J(\mathbf{x})$. In our approach, we only use the neural networks in system identification. From our point of view, when the system is completely unknown, it is difficult to initialize the admissible control [60] for the action neural networks, which is the necessary condition for convergence in ADP. Furthermore, the initialization of state utility for the critique neural networks is another challenge in ADP for controlling unknown system. Although [48, 49] show techniques to initialize the state utility by arbitrary positive-definite functions, the necessary condition is that the state utility is non-negative, which is different from the state utility assumption in our paper. In the other hand, as we have shown that the discrete-

MDP approach could approximate an admissible control for the system given some mild prerequisites, it is possible to use the result of the discrete-MDP approach as the initialization of the ADP's action network.

In addition, this work handles the learning problem such that the identification and control could be executed consecutively and repeatedly. In most of the theoretical reinforcement learning AI work, especially the ADP [49, 54–59], to tackle the unknown nature of the problem, the learning agent initially executes random actions to acquire enough number of samples for one-time identification. The number of random actions could be between thousands and millions, depending on the system. This work shows that the learning agent may not need to execute any random actions: acting 'optimally' according to the most updated approximation of the system, even if the approximation may not be precise, could stabilize the system. In Figure 10, we show that ADP could be executed in this manner, although the discrete-MDP shows faster learning speed.

There are several limitations in this paper. First, the discretization thresholds need the distribution of the next state assuming that the current state and control vectors are uniformly distributed and may require a number of ad-hoc steps. Second, in selective decentralization, we still explore all possible decoupling scheme $B(k)$, which grows exponentially. However, since the selectively decentralized system converges faster than the centralized system in most of the cases, we believe that the heavily computational model-switching phase in the selective decentralized system will be relatively short. Therefore, the selectively decentralized system may be more computationally efficient than the centralized system, which must run the learning algorithm in high dimensional data for long term.

8. Acknowledgment

The research presented in this paper was supported by the United States National Science Foundation grant No. ECCS-1407925.

References

[1] Ioannou, P.A., Decentralized adaptive control of interconnected systems, *IEEE Transactions on Automatic Control*, **31** (4), 1986, pp. 291-298.

[2] Shi, L., and Singh, S.K., Decentralized adaptive controller design for large-scale systems with higher order interconnections, *IEEE Transactions on Automatic Control*, **37** (8), 1992, pp. 1106-1118.

[3] Busoniu, L., Babuska, R., and De Schutter, B., A comprehensive survey of multiagent reinforcement learning, *IEEE Transactions on Systems, Man, And Cybernetics-Part C: Applications and Reviews*, **38** (2), 2008, pp. 156-172.

[4] Gavel, D.T., and Siljak, D., Decentralized adaptive control: structural conditions for stability, *IEEE Transactions on Automatic Control*, **34** (4), 1989, pp. 413-426

[5] Rota, G.-C., The number of partitions of a set, *The American Mathematical Monthly*, **71** (5), 1964, pp. 498-504.

[6] Narendra, K., Oleng, N., and Mukhopadhyay, S., Decentralised adaptive control with partial communication, *IEEE Proceedings-Control Theory and Applications*, **153** (5), 2006, pp. 546-555.

[7] Arslan, G., and Yksel, S., *Decentralized Q-Learning for Weakly Acyclic Stochastic Dynamic Games*, in IEEE Conference on Decision and Control, 2015, pp. 6743-6748.

[8] Arslan, G., and Yksel, S., Decentralized Q-Learning for Stochastic Teams and Games, *IEEE Transactions on Automatic Control*, **62** (4), 2017, pp. 1545-1558.

[9] Teacy, W.L., Chalkiadakis, G., Farinelli, A., Rogers, A., Jennings, N.R., McClean, S., and Parr, G.: *Decentralized Bayesian reinforcement learning for online agent collaboration*, in International Foundation for Autonomous Agents and Multiagent Systems, 2012, edn., pp. 417-424.

[10] Shah, P., and Parrilo, P.A., H2-Optimal Decentralized Control Over Posets: A State-Space Solution for State-Feedback, *IEEE Transactions on Automatic Control*, **58** (12), 2013, pp. 3084-3096.

[11] Hua, C., and Ding, S.X., Decentralized networked control system design using TâĂŞS fuzzy approach, *IEEE Transactions on fuzzy systems*, **20** (1), 2012, pp. 9-21.

[12] Ranjbar-Sahraei, B., Shabaninia, F., Nemati, A., and Stan, S.-D., A novel robust decentralized adaptive fuzzy control for swarm formation of multiagent systems, *IEEE Transactions on Industrial Electronics*, **59** (8), 2012, pp. 3124-3134.

[13] Mahajan, A., Optimal decentralized control of coupled subsystems with control sharing, *IEEE Transactions on Automatic Control*, **58** (9), 2013, pp. 2377-2382.

[14] Han, Z., and Narendra, K.S., New concepts in adaptive control using multiple models, *IEEE Transactions on Automatic Control*, **57** (1), 2012, pp. 78-89.

[15] Narendra, K.S., and Balakrishnan, J., Improving transient response of adaptive control systems using multiple models and switching, IEEE Transactions on Automatic Control, **39** (9), 1994, pp. 1861-1866.

[16] Narendra, K.S., and Mukhopadhyay, S., *To communicate or not to communicate: A decision-theoretic approach to decentralized adaptive control*, in Advances in Computing and Communications, 2010, pp. 6369-6376.

[17] Battistelli, G., Hespanha, J.P., Mosca, E., and Tesi, P., Model-free adaptive switching control of time-varying plants, *IEEE Transactions on Automatic Control*, **58** (5), 2013, pp. 1208-1220.

[18] Liu, W., Gu, W., Sheng, W., Meng, X., Wu, Z., and Chen, W., Decentralized multi-agent system-based cooperative frequency control for autonomous microgrids with communication constraints, *IEEE Transactions on Sustainable Energy*, **5** (2), 2014, pp. 446-456.

[19] Bian, T., Jiang, Y., and Jiang, Z.-P., Decentralized adaptive optimal control of large-scale systems with application to power

systems, *IEEE Transactions on Industrial Electronics*, **62** (4), 2015, pp. 2439-2447.

[20] Gao, J., Dou, L., and Su, P., *Multi-model switching control of hypersonic vehicle with variable scramjet inlet based on adaptive neural network*, in World Congress on Intelligent Control and Automation, 2016, pp. 1714-1719.

[21] Lewis, F.L., Vrabie, D., and Vamvoudakis, K.G., Reinforcement Learning and Feedback Control: Using Natural Decision Methods to Design Optimal Adaptive Controllers, *IEEE Control Systems Magazine*, **2**(6), pp. 76-105.

[22] Bellon, J., *Riccati Equations in Optimal Control Theory*, 2008, available at scholarworks.gsu.edu/cgi/viewcontent.cgi?article=1045&context=math_theses.

[23] Abu-Khalaf, M., and Lewis, F.L., Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network HJB approach, *Automatica*, **41** (5), 2005, pp. 779-791.

[24] Saridis, G.N., and Lee, C.-S.G., An approximation theory of optimal control for trainable manipulators, *IEEE Transactions on Systems, Man and Cybernetics*, **9** (3), 1979, pp. 152-159.

[25] Beard, R.W., Saridis, G.N., and Wen, J.T., *Galerkin approximations of the generalized Hamilton-Jacobi-Bellman equation*, *Automatica*, **33** (12), 1997, pp. 2159-2177.

[26] Huang, C.-S., Wang, S., and Teo, K., Solving HamiltonâĂŤJacobiâĂŤBellman equations by a modified method of characteristics, *Nonlinear Analysis: Theory, Methods & Applications*, **40** (1), 2000, pp. 279-293.

[27] Lewis, F.L., and Syrmos, V.L.: *Optimal control*, John Wiley & Sons, 1995.

[28] Chui, C.K., and Chen, G., *Linear Systems and optimal control*, Springer Science & Business Media, 2012.

[29] Russell;, S., and Norvig, P., *Artificial Intelligence A Modern Approach*, 3rd edition, Prentice Hall, 2010, pp.656.

[30] Munos, R., and Moore, A.W., *Variable resolution discretization for high-accuracy solutions of optimal control problems*, in International Joint Conference on Artificial Intelligence, 1999, pp. 256.

[31] Munos, R., and Moore, A., Variable resolution discretization in optimal control,*Machine learning*, **49** (2-3), 2002, pp. 291-323.

[32] Kharroubi, I., Langren, N., and Pham, H., A numerical algorithm for fully nonlinear HJB equations: an approach by control randomization, *Monte Carlo Methods and Applications*, **20** (2), pp. 145-165.

[33] Chang, H.S.: Decentralized learning in finite Markov chains: revisited, *IEEE Transactions on Automatic Control*, **54** (7), 2009, pp. 1648-1653.

[34] Vrancx, P., Verbeeck, K., and Now, A., Decentralized learning in markov games, *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, **38** (4), 2008, pp. 976-981.

[35] Pillonetto, G., Dinuzzo, F., Chen, T., De Nicolao, G., and Ljung, L., Kernel methods in system identification, machine learning and function estimation: A survey, *Automatica*, **50** (3), 2014, pp. 657-682

[36] Keesman, K.J., *System Identification: an Introduction*, Springer-Verlag, 2011. pp. 94-97.

[37] Funahashi, K.-I., On the approximate realization of continuous mappings by neural networks, *Neural networks*, **2** (3), 1989, pp. 183-192

[38] Miller, W.T., Werbos, P.J., and Sutton, R.S., *Neural networks for control*, MIT press, 1995.

[39] Russell, S., and Norvig, P., *Artificial Intelligence: A Modern Approach*, 3rd Edition, Pearson, 2010, pp. 830-841.

[40] Lancaster, P., and Rodman, L., *Algebraic riccati equations*, Clarendon press, 1995.

[41] Arnold III, W.F., and Laub, A.J., *Generalized eigenproblem algorithms and software for algebraic Riccati equations*, in Proceedings of the IEEE, **72** (12), 1984, pp. 1746-1754.

[42] Yang, X., Liu, D., and Wang, D., Reinforcement learning for adaptive optimal control of unknown continuous-time nonlinear systems with input constraints, *International Journal of Control*, **87** (3), 2014, pp. 553-566.

[43] Bishop, C.M., Pattern Recognition, *Machine Learning*, Springer, 2006, pp. 537-541.

[44] Bellman, R., *On the theory of dynamic programming*, Proceedings of the National Academy of Sciences, **38** (8), 1952, pp. 716-719

[45] Chang, H.S., Hu, J., Fu, M.C., and Marcus, S.I., *Simulation-based algorithms for Markov decision processes*, Springer Science & Business Media, 2013.

[46] Bertsekas, D.P., *Dynamic programming and optimal control 3rd edition*, volume II, Belmont, MA: Athena Scientific, 2011.

[47] Nguyen, T., and Mukhopadhyay, S., *Identification and Optimal Control of Large-scale System Using Selective Decentralization*, in Proc. IEEE International Conference on Systems. Man and Cybernetics, 2016 pp. pp.

[48] Zhang, H., Liu, D., Luo, Y., and Wang, D., *Adaptive Dynamic Programming for Control: Algorithms and Stability*, Springer, 2013.

[49] Wei, Q., Liu, D., Lin, Q., and Song, R., Adaptive dynamic programming for discrete-time zero-sum games, *IEEE Transactions on Neural Networks and Learning Systems*, (99), 2017, pp. 1-13.

[50] Wang, F.-Y., Zhang, H., and Liu, D., Adaptive dynamic programming: an introduction, *Computational Intelligence Magazine*, IEEE, **4** (2), 2009, pp. 39-47.

[51] Watkins, C.J., and Dayan, P., Q-learning, *Machine learning*, **8**, (3-4), 1992, pp. 279-292.

[52] Barto, A.G., Temporal difference learning, *Scholarpedia*, **2** (11), 2007, pp. 1604.

[53] Nguyen, T., and Mukhopadhyay, S., *Selectively Decentralized Q-Learning*. in Proc. IEEE International Conference on Systems, Man, and Cybernetics, 2017, pp. 328-333.

[54] Wei, Q., Liu, D., and Lin, H., Value iteration adaptive dynamic programming for optimal control of discrete-time nonlinear systems, *IEEE Transactions on cybernetics*, **46** (3), 2016, pp. 840-853.

[55] Wei, Q., Song, R., and Yan, P., Data-driven zero-sum neuro-optimal control for a class of continuous-time unknown nonlinear systems with disturbance using ADP, *IEEE transactions on neural networks and learning systems*, 27 (2), 2016, pp. 444-458.

[56] Wei, Q., Lewis, F.L., Liu, D., Song, R., and Lin, H., Discrete-time local value iteration adaptive dynamic programming: Convergence analysis, *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2017, pp. 1-17.

[57] Wei, Q., Lewis, F.L., Sun, Q., Yan, P., and Song, R., Discrete-time deterministic Q-learning: A novel convergence analysis, I*EEE transactions on cybernetics*, 47 (5), 2017, pp. 1224-1237

[58] Wei, Q., Liu, D., and Lin, Q., Discrete-time local iterative adaptive dynamic programming: Terminations and admissibility analysis, *IEEE Transactions on Neural Networks and Learning Systems*, 2016.

[59] Wei, Q., Liu, D., Lin, Q., and Song, R., Discrete-time optimal control via local policy iteration adaptive dynamic programming, *IEEE transactions on cybernetics*, **47** (10), 2017, pp 3367-3379.

[60] Lewis, F.L., and Vrabie, D., Reinforcement learning and adaptive dynamic programming for feedback control, *IEEE Circuits and Systems Magazine*, **9** (3), 2009, pp. 32-50.