Multiobjective Reinforcement Learning: A Comprehensive Overview

Chunming Liu, Xin Xu, *Senior Member, IEEE*, and Dewen Hu, *Senior Member, IEEE*

***Abstract*—Reinforcement learning (RL) is a powerful paradigm for sequential decision-making under uncertainties, and most RL algorithms aim to maximize some numerical value which represents only one long-term objective. However, multiple long-term objectives are exhibited in many real-world decision and control systems, so recently there has been growing interest in solving multiobjective reinforcement learning (MORL) problems where there are multiple conflicting objectives. The aim of this paper is to present a comprehensive overview of MORL. The basic architecture, research topics, and naïve solutions of MORL are introduced at first. Then, several representative MORL approaches and some important directions of recent research are comprehensively reviewed. The relationships between MORL and other related research are also discussed, which include multiobjective optimization, hierarchical RL, and multiagent RL. Moreover, research challenges and open problems of MORL techniques are suggested.**

***Index Terms*—Markov decision process (MDP), multiobjective reinforcement learning (MORL), Pareto front, reinforcement learning (RL), sequential decision-making.**

I. Introduction

REINFORCEMENT learning (RL) was originally studied from the perspective of animal learning behaviors [1], and it has become a major class of machine learning methods [2] to solve sequential decision making problems under uncertainties [3], [4]. In an RL system, a learning agent aims to learn an optimal action policy via interactions with an uncertain environment. At each step, the learning agent is not provided explicitly what action to take, and instead it must determine the best action to maximize long-term rewards and execute it. Then the selected action makes the current state of the environment to transit into its successive state, and the agent receives a scalar reward signal that evaluates the effect of this state transition, as shown in Fig. 1. Thus, there is a feedback architecture in a learning system based on RL and the interaction between the learning agent and its environment can be described by a sequence of states, actions, and rewards. This sequential decision process is usually modeled as a Markov decision process (MDP). The rule or strategy for action selection is called a policy. In RL, the agent learns optimal or near-optimal action policies from such interactions in order to maximize some notion of long-term objectives.

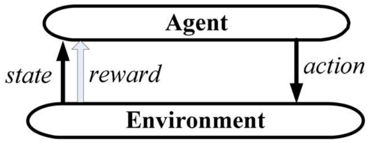

Fig. 1. Basic RL scenario.

In the past decades, there has been a large number of works on RL theory and algorithms [5]–[8]. By focusing on the computational efforts along state transition trajectories and using function approximation techniques for estimating value functions or policies, RL algorithms have produced good results in some challenging real-world problems [9], [10]. However, despite of many advances in RL theory and algorithms, one remained challenge is to scale up to larger and more complex problems. The scaling problem for sequential decision-making mainly includes the following aspects [11]. A problem that has a very large or continuous state or action space, a problem that is best described as a set of hierarchically organized tasks and sub-tasks, and a problem that needs to solve several tasks with different rewards simultaneously. An RL problem in the last aspect is called a multiobjective RL (MORL) problem, which refers to the sequential decision making problem with multiple objectives.

MORL has been regarded as an important research topic, due to the multiobjective characteristics of many practical sequential decision-making and adaptive optimal control problems in the real world. Compared with conventional RL problems, MORL problems require a learning agent to obtain action policies that can optimize two or more objectives at the same time. In MORL, each objective has its own associated reward signal, so the reward is not a scalar value but a vector. When all the objectives are directly related, a single objective can be derived by combining the multiple objectives together. If all the objectives are completely unrelated, they can be optimized separately and we can find a combined policy to optimize all of them. However, if there are conflicting objectives, any policy can only maximize one of the objectives, or realize a trade-off among the conflicting objectives [12]. Therefore, MORL can be viewed as the combination

Manuscript received February 16, 2014; revised June 11, 2014; accepted August 15, 2014. Date of publication October 8, 2014; date of current version February 12, 2015. This work was supported in part by the Program for New Century Excellent Talents in Universities under Grant NCET-10-0901 and in part by the National Fundamental Research Program of China under Grant 2013CB329401. This paper was recommended by Associate Editor A. H. Tan.

C. Liu is with the College of Mechatronics and Automation, National University of Defense Technology, Changsha 410073, China.

X. Xu is with the Institute of Unmanned Systems, College of Mechatronics and Automation, National University of Defense Technology, Changsha 410073, China (e-mail: xinxu@nudt.edu.cn).

D. Hu is with the Department of Automatic Control, College of Mechatronics and Automation, National University of Defense Technology, Changsha 410073, China (e-mail: dwhu@nudt.edu.cn).

Digital Object Identifier 10.1109/TSMC.2014.2358639

of multiobjective optimization (MOO) and RL techniques to solve the sequential decision making problems with multiple conflicting objectives.

In the MOO domain, there are two common strategies [13]: one is the multiobjective to single-objective strategy and the other is the Pareto strategy. The former strategy is to optimize a scalar value, including the weighted sum method [14], the constraint method [15], the sequential method [16], and the max-min method, etc. [17]. In these methods, a scalar value is computed from the multiple objectives for the utility of an action decision, and the conventional single-objective optimization (SOO) techniques can be used. The latter strategy is to use the vector-valued utilities. In this case, it is difficult to order the candidate solutions completely, and the Pareto optimality concept is usually used. The Pareto optimal solutions are defined as noninferior and alternative solutions among the candidate solutions, and they represent the optimal solutions for some possible trade-offs among the multiple conflicting objectives [12]. All Pareto optimal solutions constitute the Pareto front and one major research issue of MOO is to find or approximate the Pareto front.

Similar to the MOO domain, MORL algorithms can be divided into two classes based on the number of learned policies. One class comprises single-policy MORL approaches and the other, multiple-policy MORL approaches [12], [18]. Single-policy approaches aim to find the best single policy which represents the preferences among the multiple objectives as specified by a user or derived from the problem domain. The major difference among single-policy approaches is the way for determining and expressing these preferences. The aim of multiple-policy MORL approaches is to find a set of policies that approximate the Pareto front. The main difference among multiple-policy approaches is the approximation scheme for the Pareto front. Major approaches to MORL will be further discussed in Section IV.

Although there have been some recent advances in different MORL algorithms, many research challenges still remain in developing MORL theory and algorithms for real-world problems. In addition, according to the authors' knowledge, there is only one related survey paper that has been published in the literature. However, it covers the much broader topic of multiobjective sequential decision-making [18]. Therefore, the aim of this paper is to provide a comprehensive review of MORL principles, algorithms and some open problems. A representative set of MORL approaches are selected to show the overall framework of the field, to present a summary of major achievements, and to suggest some open problems for future research.

The remainder of this paper is organized as follows. In Section II, the background of MORL is briefly introduced, including MDP, RL, and MOO. The basic architecture, research topics, and naïve solutions of MORL are described in Section III. A representative set of approaches to MORL are reviewed in Section IV. In Section V, some important directions of recent research on MORL are discussed in detail. The related works are introduced in Section VI. Section VII analyzes the challenges and open problems of MORL. Section VIII concludes this comprehensive overview.

II. Background

In this section, the necessary backgrounds on MDP models, RL techniques, and MOO problems are introduced. Firstly, an MDP is characterized as the formulation of a sequential decision-making problem. Then, some basic RL techniques are introduced, where the discussion is restricted to finite state and action spaces, since most MORL results up to now are given for finite spaces. At last, the MOO problem is introduced, as well as the concept of Pareto optimality.

A. MDP Models

A sequential decision-making problem can be formulated as an MDP which is defined as a 4-tuple $\{S, A, R, P\}$. In this 4-tuple, S is the state space of a finite set of states, A is the action space of a finite set of actions, R is the reward function and P is the matrix of state transition probability. After a state transition from state s to state s’ when taking action a, $p(s, a, s')$ and $r(s, a, s')$ represent the probability and the reward of the state transition, respectively. An action policy of the MDP is defined as a function $\pi : S \rightarrow Pr(A)$, where $Pr(A)$ is a probability distribution in A.

Due to the different influences of future rewards on the present value, there are two different objective functions of an MDP. One is the discounted reward criteria, which is to estimate the optimal policy π^* satisfying the following equation:

$$J_{\pi^*} = \max_{\pi} J_{\pi} = \max_{\pi} E_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right] \tag{1}$$

where $\gamma(1 > \gamma > 0)$ is the discount factor and $r_t = r(x_t, a_t)$ is the reward at time step t, $E_\pi[\,\cdot\,]$ stands for the expectation with respect to the policy π and the probability matrix P, and J_π is the expected total reward. The other one is called the average reward criteria, which is to estimate the optimal policy π^* satisfying the following equation:

$$\rho_{\pi^*} = \max_{\pi} \rho_{\pi} = \max_{\pi}\left\{\lim_{n\rightarrow\infty} \frac{1}{n}\sum_{t=0}^{n-1} E_{\pi}\left[r_t\right]\right\} \tag{2}$$

where ρ_π is the average reward per time step for the policy π.

For the discounted reward criteria, the state value function and the state-action value function for a policy π are defined by

$$V^{\pi}(s) = E_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r_t \middle| s_0 = s\right] \tag{3}$$

$$Q^{\pi}(s, a) = E_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r_t \middle| s_0 = s, a_0 = a\right]. \tag{4}$$

According to the theory of dynamic programming (DP) [20], the following Bellman equations are satisfied:

$$V^{\pi}(s) = E_{\pi}\left[r(s, a) + \gamma V^{\pi}\left(s'\right)\right] \tag{5}$$

$$Q^{\pi}(s, a) = R(s, a) + \gamma E_{\pi}\left[Q^{\pi}\left(s', a'\right)\right] \tag{6}$$

where $R(s,\ a)$ is the expected reward received after taking action a in state s, s' is the successive state of s, and $\pi(s, a)$ represents the probability of action a taken by policy π in state s.

The optimal state-action value function is defined as

$$Q^*(s, a) = \max_{\pi} Q^{\pi}(s, a).$$

When $Q^*(s,a)$ is obtained, the optimal policy π^* can be computed by

$$\pi^*(s) = \arg\max_{a} Q^*(s, a)$$

where the optimal policy π^* is a deterministic policy, and it is a projection from S to A.

For the average reward criteria, let ρ_π [see (2)] be the average reward per time step for a policy π. The relative state value function and the relative state-action value function are defined as [4]

$$\overline{V}^{\pi}(s) = \sum_{t=0}^{\infty} E_{\pi}\,[r_t - \rho_\pi \,|\, s_0 = s]$$

$$\overline{Q}^{\pi}(s, a) = \sum_{t=0}^{\infty} E_{\pi}\,[r_t - \rho_\pi \,|\, s_0 = s, a_0 = a]$$

and the following Bellman equations are satisfied:

$$\begin{aligned} \overline{V}^{\pi}(s) &= E_{\pi}\left[r_t + \overline{V}^{\pi}\left(s'\right)\right] - \rho_\pi \\ \overline{Q}^{\pi}(s, a) &= R(s, a) - \rho_\pi + E_{\pi}\left[\overline{Q}^{\pi}\left(s', a'\right)\right]. \end{aligned} \tag{7}$$

The optimal relative state-action value function for average reward settings satisfies

$$\overline{Q}^{*}(s, a) + \rho_{\pi^*} = \max_{\pi}\left\{\overline{\mathrm{Q}}^{\pi}(\mathrm{s, a}) + \rho_\pi\right\}$$

and the optimal policy π^* can also be obtained by

$$\pi^*(s) = \arg\max_{a} \overline{Q}^{*}(s, a).$$

B. Basic RL Algorithms

The earlier approach to solve MDPs with model information is to use the dynamic programming (DP) techniques, which compute the optimal policies by estimating the optimal state-action value functions. However, traditional DP algorithms commonly require full model information and large amounts of computation are needed for large state and action spaces. Different from DP, a RL agent learns an optimal or near-optimal policy by interacting with the environment whose dynamic model is assumed to be unknown [4], [19]. As indicated in [4], based on the observed state transition data of MDPs, RL algorithms integrate the techniques of Monte Carlo, stochastic approximation, and function approximation to obtain approximate solutions of MDPs. As a central mechanism of RL, temporal-difference (TD) learning [5] can be viewed as a combination of Monte Carlo and DP. On one hand, like Monte Carlo methods, TD algorithms can learn the value functions using state transition data without model information. On the other hand, similar to DP, before a final outcome is obtained, TD methods can update the current estimation of value functions partially based on previous learned results [4], [5].

Algorithm 1 Q-Learning Algorithm [4], [6]

```
\\N: The maximum number of episodes
Initialize $Q(s, a)$ arbitrarily;
repeat (for each episode $i$)
  Initialize $s$;
  repeat (for each step of episode)
    Choose $a$ from $s$ using policy derived from $Q(s, a)$;
    Take action $a$, observe $r$, $s'$;
    $Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s, a)]$;
    $s \leftarrow s'$;
  until $s$ is terminal
until $i = N$
```

Algorithm 2 R-Learning Algorithm [4], [22]

```
\\$\rho$: The average reward
\\N: The maximum number of episodes
Initialize $Q(s, a)$ and $\rho$ arbitrarily;
repeat (for each episode $i$)
  $s \leftarrow$ current state;
  Select $a$ from $s$ using policy derived from $Q(s, a)$;
  Take action $a$, observe $r$, $s'$;
  $Q(s, a) \leftarrow Q(s, a) + \alpha[r - \rho + \gamma \max_{a'} Q(s', a') -$
  $Q(s, a)]$;
  if $Q(s, a) = \max_a Q(s, a)$ then
    $\rho \leftarrow \rho + \beta[r - \rho + \max_{a'} Q(s', a') - \max_a Q(s, a)]$;
  end if
until $i = N$
```

For the discounted reward criteria, Q-learning and Sarsa are the most widely used tabular RL algorithms. The Q-learning algorithm is shown in Algorithm 1, where α is the learning rate parameter, and r is the immediate reward. If in the limit the Q values of all admissible state-action pairs are updated infinitely often, and α decays in a way satisfying the usual stochastic approximation conditions, then the Q values will converge to the optimal value Q^* with probability 1 [20]. For the Sarsa algorithm, if each action is executed infinitely often in every state that is visited infinitely often, the action is greedy with respect to the current Q value in the limit, and the learning rate decays appropriately, then the estimated Q values will also converge to the optimal value Q^* with probability 1 [21].

For the average reward criteria, R-learning [22] is the most widely studied RL algorithm based on TD. The major steps of the R-learning algorithm are illustrated in Algorithm 2.

C. MOO Problems

The MOO problem can be formulated as follows [23], [24]:

$$\begin{aligned} &\max\ F(X) = \left[f_1(X), f_2(X), \ldots, f_{m_f}(X)\right] \\ &\text{s.t. } g_i(X) \le 0, i = 1, \ldots, m_g \end{aligned}$$

where the "max" operator for a vector is defined either in the sense of Pareto optimality or in the sense of maximizing a weighted scalar of all the elements, $X = [x_1, x_2, \ldots x_N]^T \in R^N$ is the vector of variables to be optimized, functions $g_i(X)$ $(i = 1, 2, \ldots, m_g)$ are the constraint functions of this problem, and $f_i(X)$ $(i = 1, 2, \ldots, m_f)$ are the objective functions.

The optimal solutions of an MOO problem can be described by two concepts. One is the concept of multiobjective to single

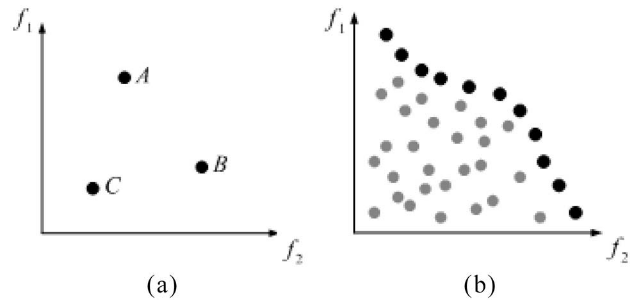

Fig. 2. Concepts of Pareto dominance and Pareto front [25]. (a) Pareto dominance. (b) Pareto front.

objective, in which a synthetic objective function is derived, and the optimal solution of this MOO problem can be obtained by solving a SOO problem. The other one is the concept of Pareto dominance and Pareto front [25]. When a solution A is better than another solution C for at least one objective, and A is also superior or at least equal to C for all the other objectives, the solution A is said to dominate C. In MOO, it is preferable to find all the dominating solutions instead of the dominated ones [25].

In the case of $m_f = 2$, as shown in Fig. 2(a), solution C is dominated by A and B, and it is hard to compare the overall performance between A and B. As indicated in [25], the Pareto front can be generated by deleting all the dominated solutions from the set of all possible solutions. From Fig. 2(b), it can be seen that the Pareto front is the set of all the black points, and the solutions corresponding to the gray points are dominated by at least one element of the Pareto front. Since it is difficult to obtain the complete Pareto front for any real-world MOO problem, a simplified way for MOO is to find a set of solutions that approximates the real Pareto front [26].

III. MORL Problem

Before providing insights into the current state of the art, and determining some important directions for future research, it is necessary to characterize the basic architecture, main research topics, and naïve solutions of MORL in advance.

A. Basic Architecture

MORL is different from traditional RL in that there are two or more objectives to be optimized simultaneously by the learning agent, where a reward vector is provided for the learning agent at each step. Fig. 3 shows the basic architecture of MORL, where there are N objectives, and r_i $(N \geq i \geq 1)$ is the ith feedback signal of the agent's current reward vector which is provided by the environment. Obviously this basic architecture illustrates the case of a single agent that has to optimize its action policies for a set of different objectives simultaneously.

For each objective i $(N \geq i \geq 1)$ and a stationary policy π, there is a corresponding state-action value function $Q_i^\pi(s,a)$, which satisfies the Bellman equation (6) or (7).

Let

$$MQ^\pi(s,a) = \left[\, Q_1^\pi(s,a),\ Q_2^\pi(s,a),\ \ldots\ Q_N^\pi(s,a) \,\right]^{\mathrm{T}}$$

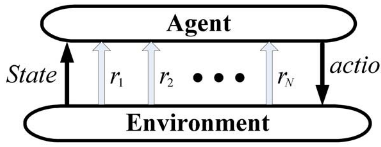

Fig. 3. Basic architecture of MORL.

where $MQ^\pi(s,a)$ is the vectored state-action value function, and it also satisfies the Bellman equation. The optimal vectored state-action value function is defined as

$$MQ^*(s,a) = \max_\pi MQ^\pi(s,a) \tag{8}$$

and the optimal policy $\pi*$ can also be obtained easily by

$$\pi^*(s) = \operatorname*{argmax}_{\mathrm{a}} MQ^*(s,a). \tag{9}$$

In this basic architecture, the optimization problems of (8) and (9) are both MOO problems.

B. Major Research Topics

MORL is a highly interdisciplinary field and it refers to the integration of MOO methods and RL techniques to solve sequential decision making problems with multiple conflicting objectives. The related disciplines of MORL include artificial intelligence, decision and optimization theory, operations research, control theory, and so on. Research topics of MORL are interplayed by MOO and RL, mainly including the preferences among different objectives (the preferences may vary with time), appropriate representations of preferences, the approximation of the Pareto front and the design of efficient algorithms for specific MORL problems. Therefore, one important task of MORL is to suitably represent the designer's preferences or ensure the optimization priority with some policies in the Pareto front. After appropriately expressing the preferences, the remained task is to design efficient MORL algorithms that can solve the sequential decision making problems based on observed state transition data.

C. Naïve Solutions

Like MOO problems, MORL approaches can be divided into two groups based on the number of the policies to be learned [12]: single-policy approaches and multiple-policy approaches.

The aim of single-policy approaches is to obtain the best policy which simultaneously satisfies the preferences among the multiple objectives as assigned by a user or defined by the application domain. In this case, the naïve approach to solve a MORL problem is to design a synthetic objective function $TQ(s,a)$, which can suitably represent the overall preferences. Similar to the Q-learning algorithm, a naïve solution of single-policy approaches to MORL is shown in Algorithm 3.

The Q-value update rule for each objective can be expressed as

$$Q_i(s,a) = (1-\alpha)Q_i(s,a) + \alpha\left(r_i + \max_{a'} Q_i\left(s',a'\right)\right)$$

where $N \geq i \geq 1$, and α is the learning rate parameter. The overall single policy can be determined based on $TQ(s,a)$,

Algorithm 3 Naïve Solution of Single-Policy Approaches to MORL

```
\\K: The maximum number of episodes
\\N: The number of objectives
Initialize $TQ(s, a)$ arbitrarily;
repeat (for each episode $j$)
    Initialize $s$;
    repeat (for each step of episode)
        Choose $a$ from $s$ using policy derived from $TQ(s, a)$;
        Take action $a$, observe $r_1$, $r_2$,..., $s'$;
        for $i = 1, 2, \ldots, N$ do
            $Q_i(s, a) \leftarrow Q_i(s, a) + \alpha[r_i + \gamma \max_{a'} Q(s', a') - Q_i(s, a)]$;
        end for
        Compute $TQ(s, a)$;
        $s \leftarrow s'$;
    until $s$ is terminal
until $j = K$
```

which can be derived using the Q-values for all the objectives. The major difference among single-policy approaches is the way in which these preferences are expressed. By making use of the synthetic objective function, the Q-values of every objective can be utilized to fairly distribute control actions.

In order to ensure the diversity in the policy space for different optimization objectives, multiple-policy approaches have been studied to obtain a set of policies that can approximate the Pareto front. The major difference among multiple-policy approaches is the manner in which the Pareto front is approximated. However, it is hard to approximate the Pareto front directly in many real-world applications. One naïve solution of multiple-policy approaches is to find policies in the Pareto front by using different synthetic objective functions. Obviously, if a set of parameters can be specified in a synthetic objective function, the optimal policy can be learned for this set of parameters. In [27], it was illustrated that by running the scalar Q-learning algorithm independently for different parameter settings, the MORL problem can be solved in a multiple-policy way.

IV. Representative Approaches to MORL

According to the different representations of preferences, several typical approaches to MORL have been developed. In this section, seven representative MORL approaches are reviewed and discussed. Among these seven representative approaches, the weighted sum approach, W-learning, the analytic hierarchy process (AHP) approach, the ranking approach, and the geometric approach are single-policy approaches. The convex hull approach and the varying parameter approach belong to multiple-policy approaches.

A. Weighted Sum Approach

In [28], an algorithm based on "greatest mass" was studied to estimate the combined Q-function. For Q-learning based on the strategy of greatest mass, the synthetic objective function is generated by summing the Q-values for all the objectives

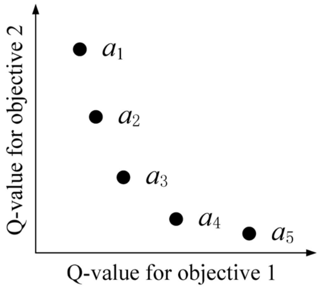

Fig. 4. Concave region of the weighted sum approach.

$$TQ(s, a) = \sum_{i=1}^{N} Q_i(s, a). \tag{10}$$

Based on the above synthetic objective function, the action with the maximal summed value is then chosen to be executed. Since Sarsa(0) is an on-policy (the samples used for weight update are generated from the current action policy) RL algorithm and it does not have the problem of positive bias, GM-Sarsa(0) was proposed for MORL in [11]. The positive bias may be caused by some off-policy RL methods which only use the estimates of greedy actions for learning updates. An advantage of GM-Sarsa(0) is as follows: since the updates are based on the actually selected actions rather than the best action determined by the value function, GM-Sarsa(0) is expected to have smaller errors between the estimated Q-values and the true Q-values.

A natural extension of the GM-Sarsa(0) approach is the weighted sum approach, which computes a linearly weighted sum of Q-values for all the objectives

$$TQ(s, a) = \sum_{i=1}^{N} w_i Q_i(s, a).$$

The weights can make the user have the ability to put more or less emphasis on each objective. In [29], the weighted sum approach was employed to combine seven vehicle overtaking objectives, and there are three navigation modes that were used to tune the weights. In [30], a similar approach was used for the combination of three objectives, which represent the degree of the crowd in an elevator, the waiting time, the number of start-ends, respectively.

Although the weighted sum approach is very simple to be implemented, the actions in concave regions of the Pareto front may not be chosen so that the Pareto front cannot be well approximated during the learning process [25]. For example, as shown in Fig. 4, if there are two objectives and five candidate actions, it can be seen that actions a_2, a_3, and a_4 are in the concave region of the Pareto front, while actions a_1 and a_5 are vertices. For all candidate actions for any positive weights $\{w_i\}$ $(N \geq i \geq 1)$, the linear weighted sums of Q-values of actions a_2, a_3, and a_4 are not the maximum. Thus, actions a_2, a_3 and a_4 will never be selected for greedy policies. Instead, one of the two actions a_1 and a_5 will be frequently chosen according to the preset weights. In order to overcome this drawback, some nonlinear functions for weighting Q-values may be used for specific problem domains [31].

B. W-Learning Approach

In order to ensure that the selected action is optimal for at least one objective, several winner-take-all methods for MORL were studied in [32]. When the current state is s, a value $W_i(s)$ is computed for each objective. Then, the selected action is based on the objective with the maximal W value.

One simple method to compute W values is called Top-Q [11], [32], which assigns the W value as the highest Q-value in the current state

$$W_i(s) = \max_a Q_i(s,a) \quad 1 \le i \le N.$$

The largest W value can be obtained as

$$W_{\max}(s) = \max_i W_i(s) = \max_i \left\{ \max_a Q_i(s,a) \right\}$$

$$W_{\max}(s) = \max_a \left\{ \max_i Q_i(s,a) \right\} \quad 1 \le i \le N \tag{11}$$

therefore the selected action is

$$\tilde{a} = \arg\max_a \left\{ \max_i Q_i(s,a) \right\} \quad 1 \le i \le N. \tag{12}$$

The synthetic objective function for the Top-Q approach can be written as

$$TQ(s,a) = \max_i Q_i(s,a) \quad 1 \le i \le N. \tag{13}$$

In Top-Q, the selected action is guaranteed to be optimal for at least one objective. However, one drawback of this approach is that the objective with the highest Q-value may have similar priorities for different actions, while other objectives cannot be satisfied due to their low action values. In addition, since the Q-values depend on the scaling of reward functions, a change in reward scaling may influence the results of the winner-take-all contest in (11)–(13). Therefore, although the Top-Q approach may obtain good performance in some cases, its behavior will be greatly influenced by the design of reward functions [11].

In order to overcome the above drawback, W-Learning was studied in [32] to compute the W values based on the following rule:

$$W_i(s) = (1-\alpha) W_i(s) + \alpha P_i(s)$$

$$P_i(s) = \max_a Q_i(s,a) - \left(r_i + \gamma \max_{a'} Q_i\left(s', a'\right) \right) \tag{14}$$

where $N \ge i \ge 1$, s' is the successive state after action a is executed. At each step, after selecting and executing the action with the highest W value, all the W values, except the highest W value (the winner or the leader), are updated according to the above rule in (14). Humphrys [32] pointed out that it may be not necessary to learn the W values, and instead they can be computed directly from the Q-values in a process called Negotiated W-learning, as shown in Algorithm 4. The negotiated W-learning algorithm can explicitly find that if an objective is not preferred to determine the next action, it may be expected to lose the most long-term reward.

Algorithm 4 Negotiated W-Learning [32]

```
\\N: The number of objectives
1:  Initialize leader $l$ with a random integer between 1 and $N$;
2:  Observe state $s$;
3:  $W_l$ =0;
4:  $a_l = \arg\max_a Q_l(s,a)$;
5:  loop:
6:    for all objectives $i$ except $l$ do
7:      $W_i = \max_a Q_i(s,a) - Q_i(s,a_l)$;
8:    end for
9:    if $\max_i W_i > W_l$ then
10:     $W_l = \max_i W_i$;
11:     $a_l = \arg\max_a Q_i(s,a)$;
12:     $l$=$i$;
13:     go to 5;
14:   else
15:     terminate the loop;
16:   end if
17: Return $a_l$;
```

C. AHP Approach [34]

Generally, the designer of MORL algorithms may not have enough prior knowledge about the optimization problem. In order to express the information of preferences, some qualitative rules are usually employed, such as "objective B is less important than objective A." The qualitative rules specify the relative importance between two objectives but do not provide a precise mathematical description. Thus, MORL algorithms can make use of the method of AHP to obtain a quantified description of the synthetic objective function $TQ(s,a)$. Compared with the original AHP method in [33], the MORL method proposed in [34] can solve sequential decision-making problems with variable number of objectives.

Based on the designer's prior knowledge of the problem, the degree of relative importance between two objectives can be quantified by L grades, and a scalar value is defined for each grade. For example, in [34], L is set to be 6, and the evaluation of the importance of objective i relative to objective j is supposed to be c_{ij}, where $N \ge i \ge 1$ and $N \ge j \ge 1$. After determining the value of c_{ij}, the relative importance matrix for all objectives $C = (c_{ij})_{N \times N}$ can be obtained.

With matrix C, the importance factor I_i (for objective i) can be calculated [34]

$$I_i = \frac{SL_i}{\sum_{j=1}^{N} SL_j}$$

where

$$SL_i = \sum_{j=1, j \ne i}^{N} c_{i,j}$$

is the importance of objective i relative to all other objectives. Then, for each objective, a fuzzy inference system can be constructed. To compare two candidate actions a_p and a_q $(a_p, a_q \in A)$, both the importance factor I_i and the value of improvement $D_i(a_p, a_q) = Q_i(s, a_p) - Q_i(s, a_q)$ are used as the inputs of the fuzzy system, and the output of the fuzzy system is the "goodness" of a_p relative to a_q.

By incorporating the fuzzy subsets and fuzzy inference rules, an action selection strategy was constructed to solve the MORL problem [34]. The main drawback of this approach is that it requires a lot of prior knowledge of the problem domain.

D. Ranking Approach

The ranking approach, also called the sequential approach, or the threshold approach, aims to solve multiobjective decision problems via an ordering or a preference relation among multiple criteria. The idea of using ordinal relations in optimal decision making was studied in the early research work by Mitten [35] and Sobel [36]. The synthetic objective function $TQ(s, a)$ was expressed in terms of "partial policies." To ensure the effectiveness of the subordinate objective (the less important objective), multiple solutions need be obtained for the optimization problem of the main objective.

Inspired by the idea of the ranking approach, an ordering of multiple objectives was established in [37] for MORL where threshold values were specified for some objectives in order to put the constraints on the objectives. One example for this kind of situations is that an unmanned vehicle performs navigation tasks in an environment while avoiding its fuel level from being empty. The MORL approach in [37] optimizes one objective while putting constraints on other objectives. The actions are chosen based on the thresholds and a lexicographic ordering (the last objective is maximized at first) [12], [37].

Let

$$CQ_i(s, a) = \min\{Q_i(s, a), C_i\}$$

where $N \geq i \geq 1$, C_i is the threshold value (the maximum allowable value) for objective i. Since objective N is assumed to be unconstrained, $C_N = +\infty$. In the ranking approach for MORL [12], [37], given a partial ordering of all objectives and their threshold values, actions a and a' can be compared by the action comparison mechanism shown in Algorithm 5, where there is a sub-function *Superior*() which was recursively defined in [12]. Based on the action comparison and selection mechanism, the MORL problem can be solved by combining this mechanism with some standard RL algorithms such as Q-learning.

The performance of the ranking-based MORL approach is mainly dependent on the ordering of the objectives as well as the threshold values. The design of an appropriate lexicographic ordering of all the objectives and their threshold values still requires some prior knowledge of the problem domain, which exhibits the designer's preferences [105]. Geibel [38] employed this idea to balance the expected return and risk, where the risk is smaller than some specified threshold, and the problem was formulated as a constrained MDP. In [39], the ranking-based MORL approach was applied to the routing problem in cognitive radio networks to address the challenges of randomness, uncertainty, and multiple metrics.

Algorithm 5 Action Comparison Mechanism of the Ranking Approach [12]

```
Superior($CQ(s_i, a')$, $CQ(s_i, a)$, $i$);
if $CQ(s_i, a') > CQ(s_i, a)$ then
    Return true;
else if $CQ(s_i, a') = CQ(s_i, a)$ then
    if $i = N$ then
        Return true;
    else
        Return Superior($CQ(s_{i+1}, a')$, $CQ(s_{i+1}, a)$, $i + 1$);
    end if
else
    Return false;
end if
```

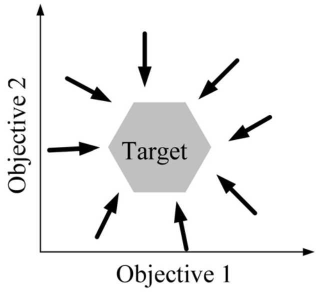

Fig. 5. Predicted target set (two objectives).

E. Geometric Approach

To deal with dynamic unknown Markovian environments with long-term average reward vectors, Mannor and Shimkin [40] proposed a geometric approach to MORL. It is assumed that the actions of other agents may influence the dynamics of the environment. Sufficient conditions for state recurrence, i.e., the game is irreducible or ergodic, are also assumed to be satisfied. In [40], using the proposed geometric-based idea, two MORL algorithms, called multiple directions RL (MDRL) and single direction RL (SDRL), were presented to approximate a desired target set in a multidimensional objective space, as shown in Fig. 5. This target set can be viewed as the synthetic objective function $TQ(s, a)$, which satisfies some geometric conditions of a dynamic game model. The MDRL and SDRL algorithms are based on the reachability theory for stochastic games, and the main mechanism behind these two algorithms is to steer the average reward vector to the target set.

When a nested class of target sets is prescribed, Mannor and Shimkin [40] also provided an extension of the geometric MORL algorithm, where the goal of the learning agent was to approach the target set with the smallest size. A particular example of this case is to solve constrained MDPs with average rewards. In addition, the geometric algorithms also need some prior knowledge of the problem domain to define the target set.

Algorithm 6 Convex Hull Algorithm [42]

```
Initialize $\hat{Q}(s, a)$, for all $s$, $a$, arbitrarily;
while not converged do
  for all $s \in S$, $a \in A$ do
    $\hat{Q}(s, a) \leftarrow E\left[\vec{r}(s, a) + \gamma\, hull \bigcup_d \hat{Q}(s', a') | s, a\right]$;
  end for
end while
Return $\hat{Q}$;
```

F. Convex Hull Approach

Barrett and Narayanan [42] presented a multiple-policy algorithm to MORL, which can simultaneously learn optimal policies for all linear preference assignments in the objective space. Two operations on convex hulls are defined as follows.

Definition 1 [42]: Translation and scaling operations

$$\vec{u} + bV \equiv \{\vec{u} + b\vec{v} | \, \vec{v} \in V\}. \tag{15}$$

Definition 2 [42]: Summing two convex hulls

$$U + V \equiv \text{hull}\,\{\vec{u} + \vec{v} | \, \vec{u} \in U, \vec{v} \in V\} \tag{16}$$

where $\vec{u}$ and $\vec{v}$ are vectors in convex hulls U and V, respectively. Equation (15) indicates that the convex hull of a transformed (translation and scaling) convex hull is just itself and (16) indicates that the convex hull of two summed convex hulls requires more computation. Based on the definitions, the convex hull algorithm is illustrated Algorithm 6, where $\hat{Q}$ is the vertices of the convex hull of all possible Q-value vectors.

This algorithm can be viewed as an extension of standard RL algorithms where the expected rewards used to update the value function are optimal for some linear preferences. Barrett and Narayanan [42] presented this algorithm and proved that the solution can find the optimal policy for any linear preference function. This solution can be simplified as the standard value iteration algorithm when a special weight vector is used.

This convex-hull technique can be integrated with other RL algorithms. Since multiple policies are learned at once, the integrated RL algorithms should be off-policy algorithms.

G. Varying Parameter Approach

Generally, a multiple-policy approach can be realized by performing multiple runs with different parameters, objective thresholds, and orderings in any single-policy algorithm. For example, as indicated in [12] and [27], scalarized Q-learning can be used in a multiple-policy manner by executing repeated runs of the Q-learning algorithm using different parameters.

Shelton [43] applied policy gradient methods and the idea of varying parameters to the MORL domain. In the proposed approach, after estimating multiple policy gradients for each objective, a weighted gradient is obtained and multiple policies can be found by varying the weights of the objective gradients.

TABLE I
REPRESENTATIVE APPROACHES TO MORL

MORL Approaches		Basic Principle
Single-policy approaches	The weighted sum approach	A linear weighted sum of Q-values is computed as the synthetic objective function.
	The W-learning approach	Each objective has its own recommendation for action selection and the final decision is based on the objective with the largest value.
	The AHP approach	The analytic hierarchy process (AHP) is employed to derive a synthetic objective function.
	The ranking approach	"Partial policies" are used as the synthetic objective function.
	The geometric approach	A target set satisfying certain geometric conditions in multi-dimensional objective space is used as the synthetic objective function.
Multiple-policy approaches	The convex hull approach	Learn optimal value functions or policies for all linear preference settings in the objective space.
	The varying parameter approach	Performing any single-policy algorithm for multiple runs with different parameters, objective threshold values and orderings.

In summary, the representative approaches mentioned above to MORL can be briefly described in Table I. In addition to these approaches, there are some other MORL approaches proposed recently [44], [45]. The algorithm proposed in [44] is an extension of multiobjective fitted Q-iteration (FQI) [54] that can find control policies for all the linear combinations of preferences assigned to the objectives in a single training procedure. By performing single-objective FQI in the state-action space, as well as the weight space, the multiobjective FQI (MOFQI) algorithm realizes function approximation and generalization of the action-value function with vectored rewards. In [45], different immediate rewards were given for the visited states by comparing the objective vector of the current state with those of the Pareto optimal solutions that have been computed. After keeping track of the nondominated solutions and constructing the Pareto front at the end of the optimization process, the Pareto optimal solutions can be memorized in an elite list.

V. IMPORTANT DIRECTIONS OF RECENT RESEARCH ON MORL

Although the MORL approaches summarized in Section IV are very promising for further applications, there are several important directions in which MORL approaches have been improved recently.

A. Further Development of MORL Approaches

In order to obtain suitable representations of the preferences and improve the efficiency of MORL algorithms, a lot of efforts have been made recently. Estimation of distribution algorithms (EDA) was used by Handa [46] to solve MORL problems. By incorporating the notions in evolutionary MOO, the proposed method was able to acquire various strategies by a single run. Studley and Bull [47] investigated the performance of a learning classifier system for MORL, and the results demonstrated that the choice of action-selection policies can greatly affect the performance of the learning system. Kei *et al.* [48] selected effective action candidates by the α-domination strategy and used a goal-directed bias based on the achievement level of each evaluation. Zhang *et al.* [49] and Zhao and Zhang [50] used a parallel genetic algorithm (PGA) to evolve a neuro-controller, and perturbation stochastic approximation (SPSA) was used to improve the convergence of the proposed algorithm. The notion of adaptive margins [51] was adopted to improve the performance of MORL algorithms [52]. To obtain Pareto optimal policies in large or continuous spaces, some recent efforts include: Khamis and Gomaa [53] developed a multiagent framework for MORL in traffic signal control, Castelletti *et al.* [54] made use of the FQI for approximating the Pareto front, and Wiering and de Jong [55] proposed a new value iteration algorithm, called consistency multi objective dynamic programming.

Most of the MORL algorithms mentioned above belong to single-policy approaches and they were studied and tested independently. The main limitation is that the small scale of previous MORL problems may not verify the algorithm's performance in dealing with a wide range of different problem settings, and the algorithm implementations always require much prior knowledge about the problem domain.

B. Dynamic Preferences in MORL

Since many real-world optimization and control problems are not stationary, there are growing interests in solving dynamic optimization problems in recent years. Similarly, the policies obtained by MORL approaches usually rely on the preferences on the reward vectors, so it is necessary to develop MORL algorithms that take dynamic preferences into consideration.

In optimal control with multiple objectives, the designer can use the fixed-weight approach [56] to determine the optimization direction. However, if there is no exact background knowledge of the problem domain, the fixed-weight approach may find unsatisfactory solutions. For MOO, the random-weight approach [57] and the adaptive approach [58] were studied to compute the weights based on the objective data so that the manual selection of objective weights can be simplified. Nevertheless, it is hard for these two approaches to express the preference of the designer.

An important problem is to automatically derive the weights when the designer is unable to assign the weights. This problem is usually called preference elicitation (PE), and its major related work is inverse RL [59], [60]. In order to predict the future decisions of an agent from its previous decisions, a Bayesian approach was proposed [61] to learn the utility function. In [62], an apprenticeship learning algorithm was presented in which observed behaviors were used to learn the objective weights of the designer.

Aiming at the time-varying preferences among multiple objectives, Natarajan and Tadepalli [63] proposed an MORL method that can find and keep a finite number of policies which can be appropriately selected for varying weight vectors. This algorithm is based on the average reward criteria, and its schema is shown in Algorithm 7, where δ is a tunable parameter. The motivation for this algorithm is that despite of infinitely many weight vectors, the set of all optimal policies may be well represented by a small number of optimal policies.

Algorithm 7 Algorithm Schema for Dynamic MORL [63]

```
Obtain the current weight vector $\vec{w}_{\text{new}}$, set $\delta$ as a
  threshold value;
$\pi_{\text{init}} = \arg\max_{\pi\in\Pi}(\vec{w}_{\text{new}} \cdot \vec{\rho}_{\pi})$
Compute the value function vectors of $\pi_{\text{init}}$
Compute the average reward vector of $\pi_{\text{init}}$
Learn the new policy $\pi'$ through vector-based RL;
If $(\vec{w}_{\text{new}} \cdot \vec{\rho}_{\pi'} - \vec{w}_{\text{new}} \cdot \vec{\rho}_{\pi_{\text{init}}}) > \delta$, add $\pi'$ to the set
  of stored policies.
```

C. Evaluation of MORL Approaches

As a relatively new field of study, research on MORL has mainly focused on various principles and algorithms to deal with multiple objectives in sequential decision-making problems. Although it is desirable to develop standard benchmark problems and methods for the evaluation of MORL algorithms, there has been little work on this topic except a recent work by Vamplew *et al.* [12]. In previous studies on MORL, the algorithms were usually evaluated on different learning problems so that it is necessary to define some standard test problems with certain characteristics for a rigorous performance testing of MORL algorithms. For example, many MORL algorithms were tested separately on some ad hoc problems, including traffic tasks [64]–[67], medical tasks [68], robot tasks [69]–[71], network routing tasks [72], grids tasks [73], and so on. As indicated in [12], it is difficult to make comparisons among these algorithms due to the lack of benchmark test problems and methodologies. Furthermore, for ad hoc application problems, the Pareto fronts are usually unknown and it is hard to find absolute performance measures for MORL algorithms. Therefore, a suite of benchmark problems with known Pareto fronts were proposed in [12], together with a standard method for performance evaluation which can serve as a basis for future comparative studies. In addition, two classes of MORL algorithms were evaluated in [12] based on different evaluation metrics. In particular, single policy MORL algorithms were tested via online learning while offline learning performance was tested for multipolicy approaches.

VI. Related Fields of Study

MORL is a highly interdisciplinary field, and its related fields mainly include MOO, hierarchical RL (HRL) and multiagent RL (MARL). Their relations to MORL and recent progress are discussed in this section. In addition, by using a keyword-based search in Google scholar, the average number of MORL-related publications was about 90 per year from

1999 to 2001, while this average number becomes 700 per year from 2011 to 2013. Thus, it can be observed that during the past 10 years, there is a significant increase in the number of publications related to MORL.

A. MOO

MOO problems usually have no unique, optimal solution, which is different from SOO that has a single best solution. For MOO and MORL, a set of noninferior, alternative solutions called the Pareto optimal solutions can be defined instead of a single optimal solution. Nevertheless, the aim of MOO is to solve a parameter optimization problem with multiple objectives, while MORL is to solve sequential decision making problems with multiple objectives.

There are two common goals for MOO algorithms: one is to find a set of solutions that is close to the Pareto optimal front, and the other is to obtain a diverse set of solutions representing the whole Pareto optimal front. These two goals have also been studied in MORL. In order to achieve these two goals, a variety of algorithms have been presented to solve MOO problems, such as multiobjective particle swarm optimizer (MOPSO) [74], multiobjective genetic algorithms (MOGA) [14], [75], etc. These MOO algorithms have improved mathematical characteristics for solving various MOO problems. Nevertheless, when the dimension of MOO problems is high, many of these algorithms usually have decreased performance due to the difficulty of finding a wide range of alternative solutions. Moreover, it is hard for MOGA and MOPSO to solve MOO problems with concave Pareto fronts which are popularly encountered in real-world applications.

Most MOO problems have different kinds of constraints and they are also called constrained MOO problems. Until now, various constraint handling approaches [76]–[79] have been proposed to improve the performance of multi objective evolutionary algorithm and MOPSO.

B. HRL

Hierarchical RL (HRL) makes use of a divide-and-conquer strategy to solve complex tasks with large state or decision spaces. Unlike conventional RL, HRL aims to solve sequential decision-making problem that can be best described as a set of hierarchically organized tasks and sub-tasks. MORL differs from HRL in that the MORL problem requires the learning agent to solve several tasks with different objectives at once. The most outstanding advantage of HRL is that it can scale to large and complex problems [80]. One common feature of HRL and MORL is that there are multiple tasks that need to be solved for the learning agent.

Earlier HRL algorithms need prior knowledge about the high-level structure of complex MDPs. There have been some HRL approaches or formulations to incorporate prior knowledge: HAMs [81], MAXQ [82], options [83], and ALisp [84]. The usage of the prior knowledge can simplify the problem decomposition and accelerate the learning process for good policies. Currently, most HRL algorithms are based on the semi-MDP (SMDP) model. In [85], the SMDP framework was extended to concurrent activities, multiagent domains, and partially observable states. As discussed in [103], although value function approximators can be integrated with HRL, few successful results have been reported in the literature for applying existing HRL approaches to MDPs with large or continuous spaces. Furthermore, to automatically decompose the state space of MDPs or construct options is still a difficult task.

Recently some new HRL algorithms were proposed. In [86], an HRL approach was presented where the state space can be partitioned by critical state. A hierarchical approximate policy iteration (HAPI) algorithm with binary-tree state space decomposition was presented in [87]. In the HAPI approach, after decomposing the original MDP into multiple sub-MDPs with smaller state spaces, better near-optimal local policies can be found and the final global policy can be derived by combining the local policies in each sub-MDP. For MORL problems, a hierarchical learning architecture with multiple objectives was proposed in [88]. The major idea of this approach is to make use of a reference network so that an internal reinforcement representation can be generated during the operation of the learning process. Furthermore, internal reinforcement signals from different levels can be provided to represent multilevel objectives for the learning system.

C. MARL

In MORL, the learning agent aims to solve sequential decision problems with reward vectors, and multiple policies may be obtained separately through decomposition. For multiple policy approaches, the MORL problem can be solved in a distributed manner which is closely related to multiagent RL (MARL) [89]. In MARL, each agent may also have its own objective and there may be multiple objectives in the learning system. However, most of the efforts in designing MARL systems have been focused on the communication and interaction (cooperation, competition, and mixed strategies) among agents. In MORL, usually there is no explicit sharing of information among objectives. The main task of an MARL system is that autonomous multiple agents explicitly consider other agents and coordinate their action policies so that a coherent joint behavior can be realized.

In recent years, the research results on MARL can be viewed as a combination of temporal-difference learning, game theory, and direct policy search techniques. There are several major difficulties in the research of MARL algorithms, which are different from single-agent settings. The first difficulty is the existence of multiple equilibriums. In Markov games with a single equilibrium value, the optimal policy can be well defined. But when there are multiple equilibriums, MARL algorithms have to ensure the agents to coordinate their policies for selecting appropriate equilibriums. From an empirical viewpoint, we should carefully consider the influence of multiple equilibriums on MARL algorithms, and undesirable equilibriums may be easily reached due to certain game properties [90]. In the design of MARL algorithms, one major aim is to make the agents' policies converge to desirable equilibriums. There have been many heuristic exploration strategies proposed in the literature so that the probability for reaching the optimal equilibriums can be improved in identical interest games [89]–[92].

VII. Challenges and Open Problems

In addition to the three important aspects for further development in Section V, there are several challenges and open problems in MORL, which will be discussed in the sequel.

A. State/Feature Representation in MORL

The problem of selecting an efficient state or feature representation in a real-world problem has been an important topic in RL [4], [8]. In the earlier research on RL, there has been some important progress in RL theory and algorithms for discrete-state MDPs. However, most real-world problems have large or continuous state spaces, thus, the huge computational costs will make earlier tabular RL algorithms be impractical for real applications. In the past decade, approximate value functions or policies with feature representations have been widely studied. But one main obstacle is that many RL algorithms and theories with good convergence properties usually rely on manually selected feature representations. Thus, it is difficult to accurately approximate the optimal value functions or policies without carefully selected features. Some recent advances in automatic feature representation include kernel methods and graph Laplacian approaches for RL [93], [94]. In MORL, there are multiple objectives to be achieved and multiple value functions may need to be approximated simultaneously. The feature representation problem in MORL is more complicated due to the existence of multiple objectives or value functions. Therefore, state or feature representation in MORL is still a big challenge for further research.

B. Value Function Approximation in MORL

As discussed above, to solve MDPs with large or continuous state spaces, value function approximation (VFA) is a key technique to realize generalization and improve learning efficiency [95]. According to the basic properties of function approximators, there are two different kinds of VFA methods, i.e., linear VFA [96], [97] and nonlinear VFA [9], [98], [99]. In some real-world applications, multilayer neural networks were commonly employed as the nonlinear approximators for VFA. However, the empirical results of successful RL applications using nonlinear VFA commonly lack rigorous theoretical analysis. Some negative results concerning divergence were reported for Q-learning and TD learning based on direct gradient rules [4], [20]. Hence, many RL approaches with VFA require significant design efforts or problem insights, and it is hard to find a basis function set that is both sufficiently simple and sufficiently reliable. To solve MDPs with large or continuous state spaces, MORL algorithms also require VFA to improve generalization ability and reduce computational costs. The additional representation of the preferences among different objectives is more difficult for developing VFA techniques, especially when the MORL problem has dynamic preferences. Hence, VFA becomes a greater challenge for MORL than that for standard RL.

C. Convergence Analysis of MORL Algorithms

Both the Q-learning algorithm and the Sarsa algorithm have some attractive qualities as basic approaches to RL. The major advantage is the fact that they are guaranteed to converge to the optimal solution for a single MDP with discrete state and action spaces. Suppose there are N objectives in an MORL problem, then this problem can be considered as N sub-MDPs (a sub-MDP is an MDP with one single objective). Hence, the convergence results of the algorithm to solve this MORL problem not only depend on the convergence of all the algorithms to solve these sub-MDPs but also depend on the representations of the preferences among all the objectives. The convergence of single-policy approaches to MORL can be analyzed based on the results of the RL algorithms used to solve sub-MDPs. However, the convergence of multiple-policy approaches has to consider the way how the Pareto front is approximated. In short, for stationary preferences, the convergence analysis of MORL algorithms is mainly dependent on the properties of the learning algorithms to solve MDPs and the representations of the preferences. For dynamic preferences, the dynamic characteristics must be considered additionally. So far, the convergence of MORL algorithms commonly lacks rigorous theoretical results.

D. MORL in Multiagent Systems

A multiagent system (MAS) is a system that has multiple interacting autonomous agents, and there are increasing numbers of application domains that are more suitable to be solved by a multiagent system instead of a centralized single agent [88], [100]. MORL in multiagent systems is a very important research topic, due to the multiobjective nature of many practical multiagent systems. How to extend the algorithms and theories of MORL in single-agent systems to MORL in multiagent systems is an open problem. In particular, to achieve several objectives at once by the cooperation of multiple agents in multiagent systems is a difficult problem to be solved. If there are competitions in multiagent systems, the MORL problem will become more challenging.

E. Applications of MORL

In recent years, RL has been applied in a variety of fields [101]–[104], but the research on MORL algorithms is a relatively new field of study, and as such there are few real-world applications so far. The following difficult problems should be studied before MORL can be successfully applied in real-time complex systems. One is to develop efficient MORL algorithms that can solve MDPs with large or continuous state spaces. The other is to evaluate the performance of different MORL algorithms in practice and find the gap between theoretical analysis and real performance. The third one is to consider the constraints from real-world applications when developing theoretical models for MORL and to investigate how theoretical results can promote the development of new algorithms and mechanisms.

Since in many real-world applications, there are multiple conflict objectives to be optimized simultaneously, it can be expected that MORL will find more application domains with the development of new theory and algorithms.

VIII. Conclusion

In this paper, the background, basic architectures, major research topics, and naïve solutions of MORL were introduced at first, then several representative approaches were reviewed and some important directions of recent research were discussed in detail. There are two main advantages of MORL. One is that MORL is very useful to improve the performance of single objective RL by generating highly diverse Pareto-optimal models for constructing policy ensembles in domains with multiple objectives. The second is that MORL algorithms can realize a trade-off between accuracy and interpretability for sequential decision-making tasks.

For single-policy approaches, the weighted sum and W-learning approaches are very simple to implement, but they cannot express exactly the preferences of the designer. The AHP, ranking, and geometric approaches may express the preferences more exactly, but they need more prior knowledge of the problem domain. For multiple-policy approaches, the convex hull algorithm can learn optimal policies for all linear preference assignments over the objective space at once. The varying parameter approach can be easily implemented by performing multiple runs with different parameters, objective threshold values, and orderings in any single-policy algorithm.

MORL approaches have been improved recently in three important aspects: enhancing their solution qualities, adapting dynamic preferences and constructing evaluation systems. The main challenges and open problems in MORL include value function approximation, feature representation, convergence analysis of algorithms and the application of MORL to multiagent systems and real world problems. It can be expected that there will be more and more research progress toward these directions.

Acknowledgment

The authors would like to thank the Associate Editors and anonymous reviewers for their valuable comments and suggestions, which greatly improved the quality of this paper.

References

[1] E. L. Thorndike, *Animal Intelligence*. Darien, CT, USA: Halfner, 1911.

[2] A. M. Turing, "Computing machinery and intelligence," *Mind*, vol. 59, pp. 433–460, Oct. 1950.

[3] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *J. Artif. Intell. Res.*, vol. 4, no. 1, pp. 237–285, 1996.

[4] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 1998.

[5] R. S. Sutton, "Learning to predict by the method of temporal differences," *Mach. Learn.*, vol. 3, no. 1, pp. 9–44, 1988.

[6] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, no. 3, pp. 279–292, 1992.

[7] F. Y. Wang, H. Zhang, and D. Liu, "Adaptive dynamic programming: An introduction," *IEEE Comput. Intell. Mag.*, vol. 4, no. 2, pp. 39–47, May 2009.

[8] X. Xu, D. W. Hu, and X. C. Lu, "Kernel based least-squares policy iteration for reinforcement learning," *IEEE Trans. Neural Netw.*, vol. 18, no. 4, pp. 973–992, Jul. 2007.

[9] R. H. Crites and A. G. Barto, "Elevator group control using multiple reinforcement learning agents," *Mach. Learn.*, vol. 33, nos. 2–3, pp. 235–262, 1998.

[10] G. J. Tesauro, "Practical issues in temporal difference learning," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 257–277, 1992.

[11] N. Sprague and D. Ballard, "Multiple-goal reinforcement learning with modular Sarsa(0)," in *Proc. 18th Int. Joint Conf. Artif. Intell.*, 2003, pp. 1445–1447.

[12] P. Vamplew, R. Dazeley, A. Berry, R. Issabekov, and E. Dekker, "Empirical evaluation methods for multiobjective reinforcement learning algorithms," *Mach. Learn.*, vol. 84, nos. 1–2, pp. 51–80, 2011.

[13] G. Mitsuo and C. Runwei, *Genetic Algorithms and Engineering Optimization*. Beijing, China: Tsinghua Univ. Press, 2003.

[14] I. Y. Kim and O. L. de Weck, "Adaptive weighted sum method for multiobjective optimization: A new method for Pareto front generation," *Struct. Multidiscipl. Optim.*, vol. 31, no. 2, pp. 105–116, 2006.

[15] A. Konaka, D. W. Coitb, and A. E. Smith, "Multi-objective optimization using genetic algorithms: A tutorial," *Reliab. Eng. Syst. Safety*, vol. 91, no. 9, pp. 992–1007, Sep. 2006.

[16] M. Yoon, Y. Yun, and H. Nakayama, *Sequential Approximate Multiobjective Optimization Using Computational Intelligence*. Berlin, Germany: Springer, 2009.

[17] J. G. Lin, "On min-norm and min-max methods of multi-objective optimization," *Math. Program.*, vol. 103, no. 1, pp. 1–33, 2005.

[18] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley, "A survey of multi-objective sequential decision-making," *J. Artif. Intell. Res.*, vol. 48, no. 1, pp. 67–113, Oct. 2013.

[19] J. Si, A. Barto, W. Powell, and D. Wunsch, *Handbook of Learning and Approximate Dynamic Programming*. Hoboken, NJ, USA: Wiley-IEEE Press, 2004.

[20] T. Jaakkola, M. I. Jordan, and S. P. Singh, "On the convergence of stochastic iterative dynamic programming algorithms," *Neural Comput.*, vol. 6, no. 6, pp. 1185–1201, Nov. 1994.

[21] S. P. Singh, T. Jaakkola, M. L. Littman, and C. Szepesvari, "Convergence results for single-step on-policy reinforcement learning algorithms," *Mach. Learn.*, vol. 38, no. 3, pp. 287–308, Mar. 2000.

[22] A. Schwartz, "A reinforcement learning method for maximizing undiscounted rewards," in *Proc. 10th Int. Conf. Mach. Learn.*, 1993, pp. 298–305.

[23] H. L. Liao, Q. H. Wu, and L. Jiang, "Multi-objective optimization by reinforcement learning for power system dispatch and voltage stability," in *Proc. Innov. Smart Grid Technol. Conf. Eur.*, Gothenburg, Sweden, 2010, pp. 1–8.

[24] K. Sindhya, "Hybrid evolutionary multi-objective optimization with enhanced convergence and diversity," Ph.D. dissertation, Dept. Math. Inf. Tech., Univ. Jyvaskyla, Jyvaskyla, Finland, 2011.

[25] P. Vamplew, J. Yearwood, R. Dazeley, and A. Berry, "On the limitations of scalarisation for multi-objective reinforcement learning of Pareto fronts," in *Proc. 21st Aust. Joint Conf. Artif. Intell.*, vol. 5360. 2008, pp. 372–378.

[26] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca, "Performance assessment of multiobjective optimizers: An analysis and review," *IEEE Trans. Evol. Comput.*, vol. 7, no. 2, pp. 117–132, Apr. 2003.

[27] A. Castelletti, G. Corani, A. Rizzolli, R. Soncinie-Sessa, and E. Weber, "Reinforcement learning in the operational management of a water system," in *Proc. IFAC Workshop Model. Control Environ. Issues,* Yokohama, Japan, 2002, pp. 325–330.

[28] J. Karlsson, "Learning to solve multiple goals," Ph.D. dissertation, Dept. Comput. Sci., Univ. Rochester, Rochester, NY, USA, 1997.

[29] D. C. K. Ngai and N. H. C. Yung, "A multiple goal reinforcement learning method for complex vehicle overtaking maneuvers," *IEEE Trans.* Intell. Transp. Syst., vol. 12, no. 2, pp. 509–522, Jun. 2011.

[30] F. Zeng, Q. Zong, Z. Sun, and L. Dou, "Self-adaptive multi-objective optimization method design based on agent reinforcement learning for elevator group control systems," in *Proc. 8th World Congr. Int. Control Autom.*, Jinan, China, 2010, pp. 2577–2582.

[31] G. Tesauro *et al.*, "Managing power consumption and performance of computing systems using reinforcement learning," in *Advances in Neural Information Processing Systems.* Cambridge, MA, USA: MIT Press, 2007, pp. 1497–1504.

[32] M. Humphrys, "Action selection methods using reinforcement learning," in *From Animals to Animats 4*, P. Maes, M. Mataric, J.-A. Meyer, J. Pollack, and S. W. Wilson, Eds. Cambridge, MA, USA: MIT Press, 1996, pp. 134–144.

[33] X. N. Shen, Y. Guo, Q. W. Chen, and W. L. Hu, "A multi-objective optimization genetic algorithm incorporating preference information," *Inf. Control*, vol. 36, no. 6, pp. 746–753, 2007.

[34] Y. Zhao, Q. W. Chen, and W. L. Hu, "Multi-objective reinforcement learning algorithm for MOSDMP in unknown environment," in *Proc. 8th World Congr. Int. Control Autom.,* 2010, pp. 3190–3194.

[35] L. G. Mitten, "Composition principles for synthesis of optimum multistage processes," *Oper. Res.*, vol. 12, pp. 610–619, Aug. 1964.

[36] M. J. Sobel, "Ordinal dynamic programming," *Manage. Sci.*, vol. 21, pp. 967–975, May 1975.

[37] Z. Gabor, Z. Kalmar, and C. Szepesvari, "Multi-criteria reinforcement learning," in *Proc. 15th Int. Conf. Mach. Learn.*, 1998, pp. 197–205.

[38] P. Geibel, "Reinforcement learning with bounded risk," in *Proc. 18th Int. Conf. Mach. Learn.*, 2001, pp. 162–169.

[39] K. Zheng, H. Li, R. C. Qiu, and S. Gong, "Multi-objective reinforcement learning based routing in cognitive radio networks: Walking in a random maze," in *Proc. Int. Conf. Comput. Netw. Commun.*, 2012, pp. 359–363.

[40] S. Mannor and N. Shimkin, "A geometric approach to multi-criterion reinforcement learning," *J. Mach. Learn. Res.*, vol. 5, pp. 325–360, Jan. 2004.

[41] S. Mannor and N. Shimkin, "The steering approach for multi-criteria reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2001, pp. 1563–1570.

[42] L. Barrett and S. Narayanan, "Learning all optimal policies with multiple criteria," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 41–47.

[43] C. R. Shelton, "Balancing multiple sources of reward in reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2000, pp. 1082–1088.

[44] A. Castelletti, F. Pianosi, and M. Restelli, "Tree-based fitted Q-iteration for multi-objective Markov decision problems," in *Proc. Int. Joint Conf. Neural Netw.*, 2012, pp. 1–8.

[45] H. L. Liu and Q. H. Wu, "Multi-objective optimization by reinforcement learning," in *Proc. IEEE Congr. Evol. Comput.*, 2010, pp. 1–8.

[46] H. Handa, "Solving multi-objective reinforcement learning problems by EDA-RL-acquisition of various strategies," in *Proc. 9th Int. Conf. Int. Syst. Design Appl.*, 2009, pp. 426–431.

[47] M. Studley and L. Bull, "Using the XCS classifier system for multi-objective reinforcement learning," *Artif. Life*, vol. 13, no. 1, pp. 69–86, 2007.

[48] A. Kei, S. Jun, A. Takanobu, I. Kokoro, and K. Shigenobu, "Multi-criteria reinforcement learning based on goal-directed exploration and its application to bipedal walking robot," *Trans. Inst. Syst. Control Inf. Eng.*, vol. 18, no. 10, pp. 352–360, 2005.

[49] H. J. Zhang, J. Zhao, R. Wang, and T. Ma, "Multi-objective reinforcement learning algorithm and its application in drive system," in *Proc. 34th Annu. IEEE Conf. Ind. Electron.*, Orlando, FL, USA, 2008, pp. 274–279.

[50] J. Zhao and H. J. Zhang, "Multi-objective reinforcement learning algorithm and its improved convergency method," in *Proc. 6th IEEE Conf. Ind. Electron. Appl.*, Beijing, China, 2011, pp. 2438–2445.

[51] K. Hiraoka, M. Yoshida, and T. Mishima, "Parallel reinforcement learning for weighted multi-criteria model with adaptive margin," *Cogn. Neurodyn.*, vol. 3, pp. 17–24, Mar. 2009.

[52] X. L. Chen, X. C. Hao, H. W. Lin, and T. Murata, "Rule driven multi objective dynamic scheduling by data envelopment analysis and reinforcement learning," in *Proc. IEEE Int. Conf. Autom. Logist.,* Hong Kong, 2010, pp. 396–401.

[53] M. A. Khamis and W. Gomaa, "Adaptive multi-objective reinforcement learning with hybrid exploration for traffic signal control based on cooperative multi-agent framework," *Eng. Appl. Artif. Intell.*, vol. 29, pp. 134–151, Mar. 2014.

[54] A. Castelletti, F. Pianosi, and M. Restelli, "Multi-objective fitted Q-iteration: Pareto frontier approximation in one single run," in *Proc. IEEE Int. Conf. Netw. Sens. Control*, Delft, The Netherlands, 2011, pp. 260–265.

[55] M. A. Wiering and E. D. de Jong, "Computing optimal stationary policies for multi-objective Markov decision processes," in *Proc. IEEE Int. Symp. Approx. Dyn. Program. Reinf. Learn.*, Honolulu, HI, USA, 2007, pp. 158–165.

[56] M. Oubbati, P. Levi, and M. Schanz, "A fixed-weight RNN dynamic controller for multiple mobile robots," in *Proc. 24th IASTED Int. Conf. Model. Identif. Control,* 2005, pp. 457–462.

[57] C. Q. Zhang, J. J. Zhang, and X. H. Gu, "The application of hybrid genetic particle swarm optimization algorithm in the distribution network reconfigurations multi-objective optimization," in *Proc. 3rd Int. Conf. Nat. Comput.*, vol. 2. 2007, pp. 455–459.

[58] D. Zheng, M. Gen, and R. Cheng, "Multiobjective optimization using genetic algorithms," *Eng. Val. Cot Anal.*, vol. 2, pp. 303–310, 1999.

[59] A. Y. Ng and S. Russell, "Algorithms for inverse reinforcement learning," in *Proc. 17th Int. Conf. Mach. Learn.*, 2000, pp. 663–670.

[60] C. Boutilier, "A POMDP formulation of preference elicitation problems," in *Proc. 18th Nat. Conf. Artif. Intell.*, 2002, pp. 239–246.

[61] U. Chajewska, D. Koller, and D. Ormoneit, "Learning an agent's utility function by observing behavior," in *Proc. 18th Int. Conf. Mach. Learn.*, 2001, pp. 35–42.

[62] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proc. 21st Int. Conf. Mach. Learn.*, 2004, pp. 1–8.

[63] S. Natarajan and P. Tadepalli, "Dynamic preferences in multi-criteria reinforcement learning," in *Proc. 22nd Int. Conf. Mach. Learn.,* 2005, pp. 601–608.

[64] M. A. Khamis, W. Gomaaa, "Adaptive multi-objective reinforcement learning with hybrid exploration for traffic signal control based on cooperative multi-agent framework," *Eng. Appl. Artif. Intell.*, vol. 29, pp. 134–151, 2014.

[65] Z. H. Yang and K. G. Wen, "Multi-objective optimization of freeway traffic flow via a fuzzy reinforcement learning method," in *Proc. 3rd Int. Conf. Adv. Comput. Theory Eng.*, vol. 5, 2010, pp. 530–534.

[66] K. G. Wen, W. G. Yang, and S. R. Qu, "Efficiency and equity based freeway traffic network flow control," in *Proc. 2nd Int. Conf. Comput. Autom. Eng.*, vol. 5. 2010, pp. 453–456.

[67] H. L. Duan, Z. H. Li, and Y. Zhang, "Multi-objective reinforcement learning for traffic signal control using vehicular ad hoc network," *EURASIP J. Adv. Signal Process.*, vol. 2010, pp. 1–7, Mar. 2010, Art. ID 724035.

[68] R. S. H. Istepanian, N. Y. Philip, and M. G. Martini, "Medical QoS provision based on reinforcement learning in ultrasound streaming over 3.5G wireless systems," *IEEE J. Select. Areas Commun.*, vol. 27, no. 4, pp. 566–574, May 2009.

[69] Y. Nojima, F. Kojima, and N. Kubota, "Local episode-based learning of multi-objective behavior coordination for a mobile robot in dynamic environments," in *Proc. 12th IEEE Int. Conf. Fuzzy Syst.*, vol. 1. 2003, pp. 307–312.

[70] D. C. K. Ngai and N. H. C. Yung, "Automated vehicle overtaking based on a multiple-goal reinforcement learning framework," in *Proc. IEEE Int. Conf. Control Appl.,* Seattle, WA, USA, 2010, pp. 530–535.

[71] T. Miyazaki, "An evaluation pattern generation scheme for electric components in hybrid electric vehicles," in *Proc. 5th IEEE Int. Conf. Int. Syst.*, Yokohama, Japan, 2010, pp. 321–326.

[72] A. Petrowski, F. Aissanou, I. Benyahia, and S. Houcke, "Multicriteria reinforcement learning based on a Russian doll method for network routing," in *Proc. 5th IEEE Int. Conf. Intell. Syst.*, 2010, pp. 321–326.

[73] J. Perez, C. Germain-Renaud, B. Kegl, and C. Loomis, "Multi-objective reinforcement learning for responsive grids," *J. Grid Comput.*, vol. 8, no. 3, pp. 473–492, 2010.

[74] L. V. S. Quintero, N. R. Santiago, and C. A. C. Coello, "Towards a more efficient multi-objective particle swarm optimizer," in *Multi-Objective Optimization in Computational Intelligence: Theory and Practice*. Hershey, PA, USA: IGI Global, 2008, pp. 76–105.

[75] H. Li and Q. F. Zhang, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.

[76] G. Tsaggouris and C. Zaroliagis, "Multiobjective optimization: Improved FPTAS for shortest paths and non-linear objectives with applications," *Theory Comput. Syst.*, vol. 45, no. 1, pp. 162–186, 2009.

[77] J. P. Dubus, C. Gonzales, and P. Perny, "Multiobjective optimization using GAI models," in *Proc. Int. Conf. Artif. Intell.*, 2009, pp. 1902–1907.

[78] P. Perny and O. Spanjaard, "Near admissible algorithms for multiobjective search," in *Proc. Eur. Conf. Artif. Intell.*, 2008, pp. 490–494.

[79] G. G. Yen and W. F. Leong, "A multiobjective particle swarm optimizer for constrained optimization," *Int. J. Swarm Intell. Res.*, vol. 2, no. 1, pp. 1–23, 2011.

[80] N. Dethlefs and H. Cuayahuitl, "Hierarchical reinforcement learning for adaptive text generation," in *Proc. 6th Int. Conf. Nat. Lang. Gener.*, 2010, pp. 37–45.

[81] R. Parr and S. Russell, "Reinforcement learning with hierarchies of machines," in *Advances in Neural Information Processing Systems*. Cambridge, MA, USA: MIT Press, 1997, pp. 1043–1049.

[82] T. Dietterich, "Hierarchical reinforcement learning with the MaxQ value function decomposition," *J. Artif. Intell. Res.*, vol. 13, no. 1, pp. 227–303, Aug. 2000.

[83] D. Precup and R. Sutton, "Multi-time models for temporally abstract planning," in *Advances in Neural Information Processing Systems*. Cambridge, MA, USA: MIT Press, 1998, pp. 1050–1056.

[84] D. Andre and S. Russell, "State abstraction for programmable reinforcement learning agents," in *Proc. 18th Nat. Conf. Artif. Intell.,* 2002, pp. 119–125.

[85] A. G. Barto and S. Mahadevan, "Recent advances in hierarchical reinforcement learning," *Discrete Event Dyn. Syst. Theory Appl.*, vol. 13, nos. 1–2, pp. 341–379, 2003.

[86] Z. Jin, W. Y. Liu, and J. Jin, "Partitioning the state space by critical states," in *Proc. 4th Int. Conf. Bio-Inspired Comput.*, 2009, pp. 1–7.

[87] X. Xu, C. Liu, S. Yang, and D. Hu, "Hierarchial approximate policy iteration with binary-tree state space decomposition," *IEEE Trans. Neural Netw.*, vol. 22, no. 12, pp. 1863–1877, Dec. 2011.

[88] H. B. He and B. Liu, "A hierarchical learning architecture with multiple-goal representations based on adaptive dynamic programming," in *Proc. Int. Conf. Netw. Sens. Control*, 2010, pp. 286–291.

[89] L. Busoniu, R. Babuska, and B. De Schutter, "A comprehensive survey of multi-agent reinforcement learning," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 38, no. 2, pp. 156–172, Mar. 2008.

[90] C. Claus and C. Boutilier, "The dynamics of reinforcement learning in cooperative multiagent systems," in *Proc. 15th Nat. Conf. Artif. Intell.*, 1998, pp. 746–752.

[91] S. Devlin and D. Kudenko, "Theoretical considerations of potential-based reward shaping for multi-agent systems," in *Proc. 10th Annu. Int. Conf. Auton. Agents Multiagent Syst.*, 2011, pp. 225–232.

[92] F. Leon, "Evolving equilibrium policies for a multiagent reinforcement learning problem with state attractors," in *Proc. Int. Conf. Comput. Collect. Intell.*, Gdynia, Poland, 2011, pp. 201–210.

[93] X. Xu, Z. Hou, C. Lian, and H. He, "Online learning control using adaptive critic designs with sparse kernel machines," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 5, pp. 762–775, May 2013.

[94] S. Mahadevan and M. Maggioni, "Proto-value functions: A Laplacian framework for learning representation and control in Markov decision processes," *J. Mach. Learn. Res.*, vol. 8, pp. 2169–2231, Jan. 2007.

[95] D. Liu, Y. Zhang, and H. Zhang, "A self-learning call admission control scheme for CDMA cellular networks," *IEEE Trans. Neural Netw.*, vol. 16, no. 5, pp. 1219–1228, Sep. 2005.

[96] X. Xu, H. G. He, and D. W. Hu, "Efficient reinforcement learning using recursive least-squares methods," *J. Artif. Intell. Res.*, vol. 16, pp. 259–292, Apr. 2002.

[97] J. Boyan, "Technical update: Least-squares temporal difference learning," *Mach. Learn.*, vol. 49, nos. 2–3, pp. 233–246, 2002.

[98] G. Tesauro, "TD-Gammon, a self-teaching backgammon program, achieves master-level play," *Neural Comput.*, vol. 6, no. 2, pp. 215–219, Mar. 1994.

[99] W. Zhang and T. Dietterich, "A reinforcement learning approach to job-shop scheduling," in *Proc. 14th Int. Joint Conf. Artif. Intell.*, 1995, pp. 1114–1120.

[100] J. Wu *et al.*, "A novel multi-agent reinforcement learning approach for job scheduling in grid computing," *Future Gen. Comput. Syst.*, vol. 27, no. 5, pp. 430–439, 2011.

[101] H. S. Ahn *et al.*, "An optimal satellite antenna profile using reinforcement learning," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 41, no. 3, pp. 393–406, May 2011.

[102] F. Bernardo, R. Agustí, J. Pérez-Romero, and O. Sallent, "Intercell interference management in OFDMA networks: A decentralized approach based on reinforcement learning," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 41, no. 6, pp. 968–976, Nov. 2011.

[103] S. Adam, L. Busoniu, and R. Babuska, "Experience replay for real-time reinforcement learning control," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 42, no. 2, pp. 201–212, Mar. 2012.

[104] F. Hernandez-del-Olmo, E. Gaudioso, and A. Nevado, "Autonomous adaptive and active tuning up of the dissolved oxygen setpoint in a wastewater treatment plant using reinforcement learning," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 42, no. 5, pp. 768–774, Sep. 2012.

[105] R. Issabekov and P. Vamplew, "An empirical comparison of two common multiobjective reinforcement learning algorithms," in *Proc. 25th Int. Australas. Joint Conf.*, Sydney, NSW Australia, 2012, pp. 626–636.

Chunming Liu received the B.Sc. and M.Sc. degrees from the National University of Defence Technology, Changsha, China, in 2004 and 2006, respectively, where he is currently pursuing the Ph.D. degree.

His current research interests include intelligent systems, machine learning, and autonomous land vehicles.

Xin Xu (M'07–SM'12) received the B.S. degree in electrical engineering from the National University of Defense Technology (NUDT), Changsha, China, in 1996, where he received the Ph.D. degree in control engineering from the College of Mechatronics and Automation, in 2002.

He is currently a Full Professor with the Institute of Unmanned Systems, College of Mechatronics and Automation, NUDT. He has been a Visiting Scientist for Cooperation Research with Hong Kong Polytechnic University, Hong Kong, University of Alberta, Edmonton, AB, Canada, University of Guelph, Guelph, ON, Canada, and University of Strathclyde, Glasgow, U.K. His current research interests include reinforcement learning, approximate dynamic programming, machine learning, robotics, and autonomous vehicles. He has authored or coauthored over 100 papers in international journals and conferences, and has coauthored four books. He currently serves as an Associate Editor for the *Information Sciences* journal, and a Guest Editor of the *International Journal of Adaptive Control and Signal Processing*.

Dr. Xu was the recipient of the 2nd Class National Natural Science Award of China in 2012 and the Fork Ying Tong Youth Teacher Fund of China in 2008. He is a Committee Member of the IEEE Technical Committee on Approximate Dynamic Programming and Reinforcement Learning and the IEEE Technical Committee on Robot Learning. He was a PC Member or a Session Chair of various international conferences.

Dewen Hu (SM'09) was born in Hunan, China, in 1963. He received the B.Sc. and M.Sc. degrees from Xi'an Jiaotong University, Xi'an, China, in 1983 and 1986, respectively, and the Ph.D. degree from the National University of Defense Technology, Changsha, China, in 1999.

From 1986, he was with the National University of Defense Technology. From 1995 to 1996, he was a Visiting Scholar with the University of Sheffield, Sheffield, U.K., and was promoted as a Professor in 1996. His current research interests include image processing, system identification and control, neural networks, and cognitive science.

Dr. Hu was the recipient of the 2nd Class National Natural Science Award of China in 2012. He is an Action Editor of *Neural Networks*.