Coordination of Electric Vehicle Charging through Multiagent Reinforcement Learning

Felipe Leno Da Silva, Cyntia E. H. Nishida, Diederik M. Roijers, and Anna H. Reali Costa, *Member, IEEE*

***Abstract*—The number of Electric Vehicle (EV) owners is expected to significantly increase in the near future, since EVs are regarded as valuable assets both for transportation and energy storage purposes. However, recharging a large fleet of EVs during peak hours may overload transformers in the distribution grid. Although several methods have been proposed to flatten peak-hour loads and recharge EVs as fairly as possible in the available time, these typically focus either on a single type of tariff or on making strong assumptions regarding the distribution grid. In this article, we propose the *MultiAgent Selfish-COllaborative architecture* (MASCO), a Multiagent Multiobjective Reinforcement Learning architecture that aims at simultaneously minimizing energy costs and avoiding transformer overloads, while allowing EV recharging. MASCO makes minimal assumptions regarding the distribution grid, works under any type of tariff, and can be configured to follow consumer preferences. We perform experiments with real energy prices, and empirically show that MASCO succeeds in balancing energy costs and transformer load.**

***Index Terms*—Electric Vehicles, Congestion Management, Smart Grid, Multiagent Systems, Multiobjective Reinforcement Learning.**

Nomenclature

A	Action space
B	Battery level
C	Collaboration weight
CC	Collaboration criteria
CO	Collaboration optimization
$\boldsymbol{D}$	Joint observation space
EV	Electric Vehicle
G	Set of friendly agents
k	Decision step
n	Number of agents
P	Transition function
$Q\backslash W\backslash V$	Utility function
R	Reward function
S	State space
SO	Selfish optimization
$\boldsymbol{U}$	Joint action space
$\boldsymbol{w}$	Preference vector
α	Learning rate
γ	Discount factor
ε	Non-greedy probability
π	Control Policy
Θ	Current time
ζ	Transformer load

I. Introduction

Electric Vehicles (EVs) have been regarded as the future of the automobile industry, as they offer a clean and noiseless mean of transportation [1]. Additionally, EVs may be used as energy storage devices in Smart Grid appliances [2]. However, it is generally known that the power grid could be greatly affected by the energy demand introduced by EVs [3]. Without any charging control, EVs automatically charge when plugged-in, which may cause grid congestion.

Different approaches have been proposed to allow EV battery recharge while avoiding grid overload. Some of these approaches are based on multi-tariff charging [4]. However, time-of-use tariffs alone have been shown to merely shift the peak load [5], which does not solve the congestion problem. This happens due to the lack of coordination, as the EV owners, that primarily care about charging costs, recharge in the beginning of the cheapest period, causing congestion in that period.

Another approach is to regard each EV as an intelligent control system that chooses its own charging schedule. This type of approach generally models EVs as agents in Multiagent Systems (MAS). Even though MAS-based technologies are suitable to the coordinated charging problem, often proposals are either: *(i) centralized* [6], [7], which causes scalability issues; *(ii) restricted* [5], [8], usually assuming that the distribution station follows a certain control algorithm and interacts with EVs, or that all EVs follow the same control algorithm; or *(iii) solely based on dynamic pricing interactions* [4], which may take a long time to implement in countries with fixed price plans [1], [9] due to regulations or structural costs.

We propose a MAS control architecture to coordinate EV charging by minimizing energy costs, while avoiding grid overloads and maintaining an adequate battery level. Balancing between those objectives requires reasoning about the relative risk of applying each action for all objectives leading to different trade-offs, as well as the utilities that users would derive from these available trade-offs. For this reason, we solve this problem following an explicitly multiobjective approach. We consider a low-voltage urban network in a radial layout, in which EVs must avoid overloads in the distribution transformer they are connected to.

This study was financed in part by São Paulo Research Foundation (FAPESP), grants 2015/16310-4 and 2018/00344-5, Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brazil (CAPES) - Finance Code 001, and CNPq (grants 425860/2016-7 and 307027/2017-1).

F. L. Silva, C. E. H. Nishida, and A. H. R. Costa are with the Intelligent Techniques Laboratory (LTI), Universidade de São Paulo, São Paulo, Brazil. E-mail: {f.leno,cyntiaeico,anna.reali}@usp.br. D. M. Roijers is with the Inst. of ICT, HU University of Applied Sciences Utrecht, The Netherlands, and the AI laboratory, Vrije Universiteit Brussel, Belgium. E-mail: diederik.yamamoto-roijers@hu.nl.

We propose the *MultiAgent Selfish-COllaborative architecture* (MASCO), which is based on *Multiagent Multiobjective Reinforcement Learning* and defines locally when an EV should recharge its battery or wait for a more appropriate time. EVs are agents in a multiagent architecture and act based on locally observable information and communication only. Local information comprises EV's own battery level and information regarding its own environment, that is, the transformer load and energy price. An EV gets information about other EVs through communication. MASCO agents decide whether to behave selfishly or collaboratively based on a heuristic, at a given time. We contribute MASCO as an approximated algorithm to solve the coordinated charging problem in a practical and realistic manner. Moreover, MASCO agents can work concurrently with agents following any type of policy or learning algorithm and do not require manual inputs in regard to daily energy requirements. We empirically show that MASCO learns how to coordinate charging both in time-of-use and dynamic pricing environments.

The remainder of this article is organized as follows: Section II defines the Coordinated Charging Problem. Section III presents related work and Section IV shows a high level description of our proposal; Section V provides a review on *Smart Grid* and *Reinforcement Learning*; Section VI gives a detailed description of MASCO; Section VII presents our experiments, along with discussions. Finally, Section VIII concludes the article and discusses future work.

II. Problem Statement

We assume a typical situation in a residential area, in which the local distribution grid is connected to a number of residential households. Each household has an EV that a consumer uses for personal transportation. In this article, the consumer is who uses the car, regardless of whether it is an individual or a group of people. If too many EVs are recharging at the same time, this causes an overload in the transformer. Therefore, as each EV may have a communication link with other cars, EVs should coordinate to prevent transformer overload.

When an EV is plugged-in, we defer control to an autonomous agent. At each time step k, all agents observe their local information, receive the transformer load at time step $k-1$, may communicate with neighboring agents, and observe the energy price at time step k. Then, all agents choose and apply one action, to *charge* or *not charge*, which lasts for a predetermined time interval. Our proposed architecture is agnostic to the actual implementation of the distribution grid, which can vary from region to region. It is noteworthy that the energy price can be both dynamic or fixed.

We define the following requirements for any approach to tackle this problem:

Requirement 1: Distributed Solution – All agents must define their policy reasoning over local information and data received from communication.

Requirement 2: Self-interested agents – We assume that the agents are not necessarily homogeneous or benevolent (completely collaborative), since "dumb"[1] models may exist or some agents may refuse to cooperate.

Requirement 3: Unpredictable consumer behavior – We assume that the consumer is unwilling to manually configure her daily journey. In practice, the user EV-use profile usually roughly follows a distribution that can be approximated. However, the time and energy spent cannot be exactly anticipated.

Because of these requirements for realistic EV charging problems, finding optimal solutions is typically intractable for all but the smallest instances. Therefore, in this article, we propose a heuristic method. Although MASCO cannot be proved to find optimal policies, we show in our experiments that it works well under realistic scenarios.

III. Related Work

The Coordinated Charging problem is a relevant research topic that has been extensively studied and various solutions have been proposed. Below, we present and discuss the most recent and similar work to our proposal.

Cao *et al.* [1] proposed a distributed approach that models EV coordination as an optimization problem and finds an optimal policy to reduce energy costs. However, their method only works for a time-of-use tariff, and agents book in advance a share of energy from the transformer. It is not clear how their proposed method deals with heterogeneous agents.

Yu Yang *et al.* [10] formulate the charging problem as a Markov decision process, and developed a distributed simulation-based policy improvement method. However, since they consider total observability, their approach has scalability problems.

Karfopoulos *et al.* [5], Hu *et al.* [8], and Cao and Chen [11] proposed non-cooperative game architectures that work as follows: EV owners select their daily desired charging based on the predicted price per period, minimizing the energy costs. Then, one agent related to the distribution infrastructure defines new tariffs to eliminate congestion in the cheapest periods. This procedure is iterated until an equilibrium is achieved. Although their approach is very efficient, an actual implementation would incur in structural costs and require a change in regulations to locations that do not use dynamic pricing. Also, it is not clear how their proposals work concomitantly with EVs that follow other control policies.

Vasinari *et al.* [12] propose a method that does not require that users manually set their energy usage. In their proposal, EVs recharge in a shared station[2]. Ghosh and Aggarwal [13] proposed a method in which EVs recharge in a shared station that offers different contracts with a certain amount of energy at a given price over a deadline of time. However, both methods needs to take into account all other agents in the station, which is not a scalable procedure.

Dusparic *et al.* [14] propose a distributed Multiagent Reinforcement Learning approach similar to ours. However, agents need to know consumer's desired battery level. This information is used both to process an internal feedback related to the

[1] EVs that always charge whenever plugged.

[2] Household charging spots can be seen as part of a big "charging station" , that allocates the available energy among all agents similarly to a bidding procedure.

state of charge and to execute a load prediction procedure to find the most appropriate time slot to recharge. Unlike MASCO, their proposal requires the consumer's desired battery level and has a prediction agent that predicts price and load for the next 24 hours based on current load, load historical data and weather forecast.

In summary, existing proposals have some issues in regard to: (i) *Other Agents Behavior*: many approaches do not clearly describe how they cope with heterogeneous agents; (ii) *Distribution Infrastructure*: some proposals model interactions between the EVs and the distribution infrastructure, and are either unusable or hard to apply if the existing infrastructure cannot be promptly changed; (iii) *Energy Tariff Type*: most of the proposals can be applied exclusively for one tariff type; (iv) *Precise Information about Consumer Energy Requirements*: some proposals assume that the consumer is willing and able to manually set her precise energy needs every day. We argue that this is unrealistic.

IV. Proposal

MASCO differs from other approaches because it makes minimal assumptions about the distribution infrastructure as agents only need to be able to observe the transformer load and the energy price. MASCO is also able to cope with heterogeneous agents and to handle any type of tariff.

Our proposal is a distributed Multiagent Multiobjective Reinforcement Learning System that aims to simultaneously optimize three conflicting objectives:

- **Battery Level** – Consumers need a high battery level before daily travel, therefore agents aim to maximize it.
- **Price Paid** – Agents aim to minimize the total cost of energy for the consumers.
- **Transformer Overload** – Agents explicitly aim to minimize the number of transformer overloads.

A good trade-off between the objectives must be found, and this trade-off should follow the costumer's preferences. The solution of a multiobjective problem is a set of policies with different trade-offs between objectives. Specifically, for each possible utility function that corresponds to a possible user's preferences, the solution set has at least one optimal policy. This is called *coverage set* [15]. Given a utility function[3] f, typically parameterized by a vector $\boldsymbol{w}$, it is possible to select a single optimal solution from the coverage set that maximizes user utility. It is noteworthy that the manufacturer or consumer sets her preferences and it can be dynamic; in this case, the agent should adapt its actuation according to the user's preferences.

Using reinforcement learning (RL) provides many advantages. RL can be used to train autonomous agents without an explicit environment model, which could be difficult to build due to large-scale and complexity of a smart grid. RL is adaptive, it can work with different implementations of distribution systems and adapt when conditions change. For example, RL can be used with any type of tariff without requiring customization and MASCO agents adapt to a change in usual load when new EVs plug-in. RL deals with partial observability, in a large-scale smart grid it is not expected that an EV can communicate with uncooperative EVs from different brands.

In MASCO, the agents first try to optimize multiple objectives through a *Selfish Optimization* procedure that outputs a selfish policy π_S. Also, each agent has a communication link with a set of friendly agents with which it can collaborate. The local agent learns how its actions affect other agents and it builds a collaborative policy π_C, which intends to help other agents through a *Collaborative Optimization* procedure. Finally a *Cooperation Criterion* chooses when the agent should execute π_S or π_C. For example, when the agent is very low on battery it may choose to recharge even if there is a risk that this course of action may result in an overload. The group of friendly agents may be assembled, for example, by including all EVs of the same brand in a neighborhood. All the actions of other agents are not observed and are taken as stochastic effects of the environment. Although this is an approximate solution, in practice MASCO achieves good results. Before we further detail our proposal (Section VI), we first introduce the fundamental concepts underlying MASCO.

V. Foundations

In MASCO, we use Reinforcement Learning (RL) [16]. RL is an extensively applied technique that has been successfully applied in many problems is often used in sequential decision making problems. In each decision step k, the agent observes the current state of the environment s_k and applies an action a_k. The agent observes the new state s_{k+1}, and receives a reward r_k. The agent goal is to learn a policy $\pi : S \rightarrow A$, that maps each state to the most appropriate action.

Q-Learning [17] is a popular and effective algorithm to learn how to solve sequential decision problems. Q-Learning iteratively learns a Q-table, i.e., a function that aims to estimate the long-term discounted reward associated to each state-action pair: $Q : S \times A \rightarrow \mathbb{R}$. At each decision step, Q is updated following:

$$Q_{k+1}(s_k, a_k) \leftarrow (1-\alpha)Q_k(s_k, a_k)+\alpha[r_k+\gamma \max_a Q_k(s_{k+1}, a)], \quad (1)$$

where $r_k = R(s_k, a_k, s_{k+1})$ is the observed reward, $0 < \alpha \leq 1$ is the learning rate that determines how much the newly acquired information replaces the previous one, γ is the discount factor that encodes the horizon in which rewards are relevant and Q_k is the current estimate of the Q function.

Q-Learning converges to the optimal Q function: $Q^*(s, a) = E\left[\sum_{i=0}^{\infty} \gamma^i r_i\right]$, and an optimal policy can be derived: $\pi^*(s) = \arg\max_a Q^*(s, a)$. It is noteworthy that the standard RL approach only takes into account single-agent and single-objective domains. However, many real-world decision problems are more naturally defined by multiple and conflicting objectives, such as minimizing energy price while avoiding transformer overloads. In such domains, it is often undesirable, infeasible, or even impossible to cast the decision problem to an equivalent single-objective problem. This results in RL problems in which the task completion is described by a set

[3] Also known as a scalarization function.

of o reward functions $R : S \times A \times S \rightarrow \mathbb{R}^o$, rather than a single one.

When the utility function, f, that represents the utility of the user as a function of the value-vectors is fixed throughout learning and execution, and known a priori, it can be possible to collapse multiple rewards to a single one. However, this is only possible if f is linear: $V_w = f(\mathbf{V}, \boldsymbol{w}) = \boldsymbol{w} \cdot \mathbf{V}$, where $\boldsymbol{w}$ is a weight vector that encodes the relative importance of objectives. On the other hand, there are other situations where the weights are not previously known or are expected to change. In our setting for example, consumer preferences can change over time. Hence, we need to explicitly take multiple objectives into account, and adjust the policies accordingly when the consumer's preferences change [18].

Even though Multiobjective RL (MORL) has been gaining attention due to the multiobjective nature of many real-world domains [19], most multiobjective approaches are applicable only to the single-agent case [15]. On the other hand, multiagent approaches can be extended to MAS, modeled as *Stochastic Games (SGs)*, where each agent has its own local state features, action set, and reward function [20], [21].

Agents might use only their local observations, which reduces the number of required Q-table entries. However, it also introduces an approximation error, as agents must coordinate without full observability of the state variables. This error can be alleviated by increasing the frequency of communication to disambiguate non-observable state information, but communication is expensive and not always reliable in the real world. In order to model the coordinated charging problem under a general theoretical framework, we extend SGs to a *MultiObjective Partially Observable Stochastic Game* (MOPOSG), which is composed of $< S, \boldsymbol{U}, \boldsymbol{D}, P, R_1^{o_1}, ..., R_n^{o_n} >$, where:

- n is the number of agents.
- $S = S_1 \times \cdots \times S_n$ is the state space composed of the local state space of each agent.
- $\boldsymbol{U} = A_1 \times \cdots \times A_n$ is the joint action space composed of the action space of each agent.
- $\boldsymbol{D} = Z_1 \times \cdots \times Z_n$ is the joint observation space containing all possible combinations of agent observations.
- P is the state and observation transition function, where $P(s_k, \boldsymbol{d_k}, \boldsymbol{u_k}, s_{k+1})$ denotes the probability of achieving state $s_{k+1} \in S$ and joint observation $\boldsymbol{d_k} \in D$ after executing the joint action $\boldsymbol{u_k} \in U$ in s_k.
- $R_i^{o_i} : S \times \boldsymbol{U} \times S \rightarrow \mathbb{R}^{o_i}$ is the reward function of agent Ag_i that describes a vector of o_i rewards, one for each objective.

In learning problems, P and $R_i^{o_i}$ are unknown to the agent that can only observe observations and reward returns. Solving MOPOSG optimally would be both intractable and impracticable in real-world scenarios. Hence, MASCO is a solution to find a reasonable policy without optimality guarantees.

The closest learning algorithm to MASCO is Distributed W-Learning (DWL) [22], where each agent has a set of *neighbors* with which it can cooperate. All agents outside its neighborhood are ignored. DWL does not require a single reward function and can be applied to domains with heterogeneous agents. However, Silva and Costa [23] noted that DWL cannot provide a policy that favors an objective over another according to user preferences (or constraints) in coordinated charging problems [14]. Thus, we propose MASCO as an architecture similar to DWL to solve the coordinated charging problem in a customized, distributed, and scalable way. We further detail our proposal in the next section.

VI. MASCO: Solving the Coordinated Charging problem with RL

We model the EV charging problem as a MOPOSG, in which each agent has one reward function per objective and a local observation function. As MASCO is modeled as a MOPOSG, it takes into account partial observability and stochasticity introduced by self-interested agents. Unpredictable consumer behavior also increases the environment stochasticity. MASCO is also a distributed solution as DWL. Therefore, MASCO fulfills the three requirements we defined in Section II. Because exact and even bounded approximate solutions are computationally infeasible, we focus on devising a heuristic method that works well in practice.

MASCO learns multiple policies in a distributed manner. An agent learns a policy π_S^i for each reward function r^i, maximizing it in a selfish manner. Also, for each agent j with whom the learning agent i wants to collaborate, one policy $\pi_C^{j,i}$ is learned maximizing each reward function $r^{j,i}$ in a collaborative manner. Figure 1 visually describe the MASCO architecture. Note that *Selfish Optimization* and *Collaborative Optimization* are both MORL optimization problems, in which a single policy is derived from multiple objectives. After the resulting selfish and collaborative policies are defined, a *Cooperation Criterion* chooses which of the two policies is obeyed in time step k, and the resulting action a_k is applied.

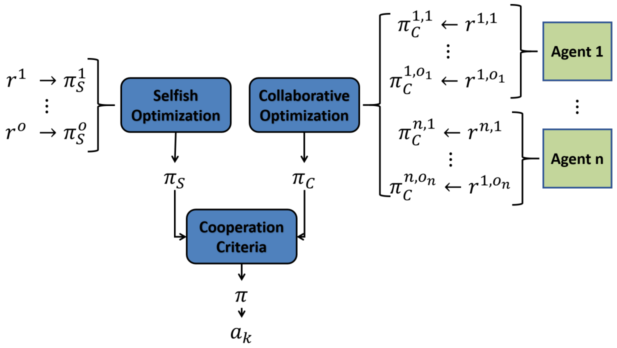
$r^1 \rightarrow \pi_S^1$
⋮
$r^o \rightarrow \pi_S^o$
Selfish Optimization
Collaborative Optimization
$\pi_C^{1,1} \leftarrow r^{1,1}$
⋮
$\pi_C^{1,o_1} \leftarrow r^{1,o_1}$
Agent 1
⋮
$\pi_C^{n,1} \leftarrow r^{n,1}$
⋮
$\pi_C^{n,o_n} \leftarrow r^{1,o_n}$
Agent n
π_S
π_C
Cooperation Criteria
π
a_k

Fig. 1. A graphical representation of MASCO. A selfish policy π_S^i is learned for each local reward r^i and a collaborative policy $\pi_C^{j,i}$ is learned for each reward $r^{j,i}$ of agent j. In each decision step, the selfish and collaborative optimization algorithms define π_S and π_C to be executed. A final Cooperation Criteria decides whether the agent should be selfish or collaborative in k, defining the resulting action to be executed, a_k.

Let o be the number of reward functions to be optimized for the agent, G the set of agents with whom the local agent wants to cooperate and are connected to the same distribution transformer (*friendly agents*), and $\boldsymbol{r} = [r^1, \ldots, r^o]^T$ the reward vector for all local objectives. $\boldsymbol{Q}^S = [Q^1, \ldots, Q^o]^T$ is the set of Q-tables related to local rewards, where Q^i is related to objective i. $SO(\boldsymbol{Q}^S, s_k)$ is the *Selfish Optimization Function* that returns an action a^S defined by the selfish-based objective optimization for state s_k. For example, if the agent wants to minimize costs and avoid overloads, a possible interpretation of this problem in the MASCO architecture is to store both a Q-table Q^1, from which a policy π_S^1 that minimizes costs can be extracted, and a Q-table Q^2, that can be used to avoid overloads through a policy π_S^2. Even though π_S^1 and π_S^2 optimize only one objective, SO can estimate which of the two objectives is more important at a given time step and alternate between the policies according to their relative importance, leading to an approximate non-stationary policy.

Now, let o_j, $j \in G$ be the number of reward functions of a given friendly agent j, $\boldsymbol{Q}^C = [Q^{1,1}, \dots, Q^{1,o_1}, Q^{2,1}, \dots, Q^{n,o_n}]^T$ is the set of Q-tables related to other agents objectives, where $Q^{j,i}$ is related to objective i of agent j. $CO(\boldsymbol{Q}^C, s_k)$ is the *Collaborative Optimization Function* that returns an action a^C for s_k aiming to collaborate with other agents. In the above mentioned example, the agent stores Q-tables $Q^{j,1}$ and $Q^{j,2}$ for each friendly agent j, and each table learns how the local actions affect the agent j. Then, CO defines the agent that needs help the most and collaborates with it to optimize one objective. For example, the local agent detects that j almost ran out of battery and thus needs to recharge. Then, the local agent chooses the *not charge* action to prevent overload.

Finally, $CC(a_k^S, a_k^C)$ is a *Cooperation Criterion* which chooses if the agent should collaborate or act in a selfish manner at step k. SO provides the best action for a local agent, CO for a friendly agent and CC balances both sides. Applying a selfish action may be beneficial if the transformer does not overload, which results in an increase of reward intake. CO and SO are implemented in this article with *Linear Scalarization Functions*, i.e.

$$SO(\boldsymbol{Q}^S, s) = CO(\boldsymbol{Q}^C, s) = \arg\max_{a \in A} f_{\boldsymbol{Q}}(a), \tag{2}$$

where f is a scalarization function, defined in this article as $f_{\boldsymbol{Q}}(a) = \sum_j \sum_{i=1}^{o} w_i Q^{j,i}(s,a)$, where w is the vector of user preferences. Note that j corresponds to the own agent in the selfish case.

A linear scalarization is a simple and common function to model utility (though it cannot provide solutions that could be optimal under non-linear utility models on the non-convex portions of a Pareto frontier [24]). Nonlinear scalarization functions could also be used, however f must be very carefully chosen, as the use of nonlinear scalarization functions may preclude convergence [15]. Thus, we left the use of nonlinear utility functions for this domain as an open problem for further studies. The $\boldsymbol{w}$ vector may be different for each agent to reflect the consumer's personal preferences.

In order to define CC, a W-function is computed as in W-Learning [25] and DWL. A W-table indicates which objective is expected to lose more of the long-term reward if its policy is not obeyed. W-values are updated only when the best action for a given objective is not executed, following the equation:

$$\begin{aligned} W^i(s_k) \leftarrow (1-\alpha)W^i(s_k) + \alpha(Q^i(s_k, a_k) - \\ (r_k^i + \gamma \max_a Q^i(s_{k+1}, a))), \end{aligned} \tag{3}$$

where r_k^i is the reward observed for objective i, and Q^i is the Q-table associated to objective i. The winning action a^S or a^C is defined according to:

$$W_{win} = \max(W_S, C \times W_C), \tag{4}$$

where parameter C weights collaboration, W_S is the W-value associated with the objective that has the highest Q-value in Q^S, and W_C is the W-value associated with Q^C. If $W_{win} = W_S$, $a_k = a_k^S$; otherwise, $a_k = a_k^C$. The selfish and collaborative W-values for all objectives are updated with Eq. 3, except the one associated with W_{win}.

CC is a heuristic that provides an approximate solution by simplifying how MASCO chooses between the two policies based only on its local observations and communications. Since communications do not provide information about friendly agents' preferences or any kind of information for unfriendly agents, the local agent can only partially observe the environment. As mentioned before, a decentralized model introduces approximation errors, therefore this heuristic cannot provide a global optimal solution.

Another approximation error occurs in the selfish and collaborative division. When an agent is selfish, it can increase its local reward intake without decreasing the global value, which happens when the transformer does not overload. However, when an agent is selfish and it causes an overload, all agents suffer a penalization and the global value decreases. It is noteworthy that MASCO is not guaranteed to converge, as it solves a relaxed formulation of the problem. MASCO is a framework to find a reasonable policy that works well under realistic scenarios, as we show in our experiments. It is expected that the computational and space complexity grow with $|G|$, as more friendly agents mean more Q-tables and W-tables that need to be stored and updated. Although updating all tables and selecting an action require simple calculations, a large G requires too much space and more samples to compute a good policy. It can be solved by limiting $|G|$, which, in turn, decreases observability.

MASCO is fully described by Algorithm 1. First (steps 1 – 4) the agent arbitrarily initiates one (Q-table, W-vector) pair for each local objective (Q^i, W^i) and one pair ($Q^{Ag,i}$, $W^{Ag,i}$) for each objective of agents in G. Then, the agent observes its own current state and communicates with other agents to receive their states (step 5). a_k^S and a_k^C are then defined (steps 7 – 8) by maximizing the weighted Q-values. Action a_k is chosen according to W_{win}, calculated by (4) (step 9) and executed following an ε-greedy strategy[4] (step 10). After the execution of all actions, the agent then updates all Q-tables according to the observed local rewards and rewards received by communication (related to other agents). W-vectors are all updated except W_{win} (steps 12 – 28). Finally, the agent can observe the next state and execute the same procedure again.

Since the reward functions may have different scales, which can lead to problems in the resulting policy [23], we perform a normalization, $Q_N(s,a) = \frac{Q(s,a)}{\sum_{a' \in A} |Q(s,a')|}$, before action selection [26] (Lines 7 and 8 of Algorithm 1).

[4] With probability $1-\varepsilon$ the action a_k will be executed; with probability ε a random action will be performed.

Algorithm 1 MASCO learning algorithm

```
Require: Set of friendly agents $G$, set of actions $A$, local objectives $o$, local state space $S$, friendly agents objectives $o_{Ag}$ and state space $S_{Ag}$, weight vector $\boldsymbol{w}$, cooperation rate $C$, discount rate $\gamma$, learning rate $\alpha$.
Initialize $Q^i(s,a)$ and $W^i(s)$, for all objective $i$.
Initialize $Q^{Ag,j}(s^{Ag},a)$ and $W^{Ag,j}(s^{Ag})$ for all objective $j$ and agent $Ag \in G$.
$\boldsymbol{Q}^S = [Q^1, \dots, Q^o]^T$
$\boldsymbol{Q}^C = [Q^{1,1}, \dots, Q^{1,o_1}, Q^{2,1}, \dots, Q^{|G|,o_{|G|}}]^T$
Observe $s_k$ and receive $s_k^{Ag}, \forall Ag \in G$.
for each decision step $k$ do
    $a_k^S = \arg\max_{a \in A} \sum_{Q^i \in \boldsymbol{Q}^S} w_i Q^i(s_k, a)$ (Eq. 2).
    $a_k^C = \arg\max_{a \in A} \sum_{Q^{Ag,i} \in \boldsymbol{Q}^C} w_i Q^{Ag,i}(s_k^{Ag}, a)$ (Eq. 2).
    Define $a_k$ according to $W_{win}$ (Eq. 4).
    All agents execute their actions.
    Observe $s_{k+1}$ and receive $s_{k+1}^{Ag}, \forall Ag \in G$.
    for each objective $i \in o$ do
        Observe $r^i$.
        Update $Q^i$ with $s_k, a_k, r^i, s_{k+1}$ (Eq. 1).
        if $W^i \neq W_{win}$ then
            Update $W^i$ with $s_k, r^i, Q^i$ (Eq. 3).
        end if
    end for
    for each friendly agent $Ag \in G$ do
        for each objective $i \in o_{Ag}$ do
            Receive $r^{Ag,i}$.
            Update $Q^{Ag,i}$ with $s_k^{Ag}, a_k, r^{Ag,i}, s_{k+1}^{Ag}$ (Eq. 1).
            if $W^{Ag,i} \neq W_{win}$ then
                Update $W^{Ag,i}$ with $s_k^{Ag}, r^{Ag,i}, Q^{Ag,i}$ (Eq. 3).
            end if
        end for
    end for
    $s_k \leftarrow s_{k+1}$, $s_k^{Ag} \leftarrow s_{k+1}^{Ag}$.
end for
```

It is noteworthy that MASCO assumes: (i) **Communication**: Agents must be able to communicate with all friendly agents $Ag \in G$ at all decision steps, as their local states and rewards are needed; (ii) **Knowledge of other friendly agents**: The local agent must know how many reward functions all agents in G have to initialize the correct number of Q-tables. As agents create all Q-tables in function of local actions, friendly agents may have different action sets and state spaces. Thus, MASCO can be used in heterogeneous MAS; (iii) **Friendly Agents Discovery**: Agents must be able to recognize the set of friendly agents G. In domains, such as in this article, where the spacial distance affects the agent communication abilities, G can be defined as a set of neighbors agents.

Coordinated Charging RL Modeling

Here, we define how to model the coordinated charging problem as an MOPOSG to solve with MASCO. All definitions are for a single local agent Ag^j and the overall space is composed of the space of each agent.

The local state is established according to the following information that is available to the agent in all time steps:

1) **Current Battery Level** – The agent's state of charge. This variable is discretized in slots of 20% of the full charge, i.e. $B = \{0-20\%, 20-40\%, \dots, 80-100\%\}$, and can be affected by a *charge* action, which recharges the battery, or a daily journey, which consumes energy.
2) **Current Time** – The current time of the day. The time is given in slots of t minutes, which was defined as $t = 15$ in this article. Thus $\Theta = \{0:0, 0:15, \dots, 23:45\}$.
3) **Transformer Load** – The transformer load in the last time step. The load is received through communication with the transformer in kW, and is discretized in: (i) LOW – up to 60% of the maximum desired load; (ii) $MEDIUM$ – between 60% and 80% of the maximum desired load; (iii) $HIGH$ – between 80% and 100% of the maximum desired load; and (iv) $OVER$ – any load greater than the previously defined intervals. Hence $\zeta = \{LOW, MEDIUM, HIGH, OVER\}$.
4) **Location** – An EV can be either *at home* or *traveling*. Thus $L = \{at\ home, traveling\}$.

According to these state variables, the complete local state space is defined as $S^j = B \times \Theta \times \zeta \times L$. Z^j is composed of the state space from the agents that Ag^j can observe, including itself. The discretization of the state space makes the learning process faster and less sensitive to noise [27]. Although a fine discretization may increase accuracy, it increases the number of states and consequently, it would require a lot of space to store all tables and much more learning trials to learn a policy. Local action space is composed of $A^j = \{\text{charge, not charge}\}$.

The transition function $P(s_k, \boldsymbol{d_k}, \boldsymbol{u_{k+1}}, s_{k+1})$ models how the environment reacts. As we are modeling a learning problem, agents do not know P and must learn how to actuate in an unknown environment. Reward functions should indicate the desirability of a state for a given objective [16]. In other words, desirable states should receive higher rewards and how much a state is desirable depends on prior knowledge. In this article, each objective is encoded by the following reward functions:

- **Battery Level** – This reward encodes the user satisfaction of having the battery in a high level. The reward awarded by the agent is $+10$ for each 20% of the battery charge the agent has available. That is, when less than 20% of the battery is available the agent receives 0 of reward, and when the battery level is above 80% the agent receives $+40$.
- **Price Paid** – The second reward intends to minimize the energy costs. Let ρ be the energy price per kWh at k, the reward is defined by $r_2 = \frac{1}{\rho \times c}$, where c is the energy consumed (in kWh) by the agent at k. That is, the reward is greater when the costs are smaller. In case of $c = 0$ (the agent is not charging), this reward returns a value correspondent to c equals to the EV typical energy consumption and ρ equals to an average price defined by the consumer or designer (i.e., if the current price is above the average price, the agent would prefer to wait until a cheaper period is available).
- **Transformer Overload** – The last reward avoids transformer overloads. If the transformer is in an overload status and the agent is charging, it receives a reward of 0. In case the

transformer load is within the desired bounds or the agent is not charging, the received reward is $+50$.

VII. Experimental Evaluation

We perform an empirical evaluation in a simulation based on real-world data to show that MASCO is an appropriate approach to learn policies in the coordinated charging problem. We evaluate the behavior of a small scale MAS, which is similar to the study cases of [14], [23], [28]. A transformer provides with safety a maximum of 40kWh for a neighborhood of 30 households, each with one EV. Using the Nissan Leaf as a reference, the EVs batteries have a capacity of 24kWh [8]. We assume that consumers daily travel follows the Danish driving pattern analyzed in [29], i.e., the average distance traveled by an EV is 42.7 km per day. We assume that the spent energy is defined as 0.11 kWh/km. The charging rate of the EV is considered as 2.3kWh and we assume that people leave home at around 7 AM and get home at roughly 6 PM. Finally, we defined the time step $t = 15$ minutes.

We evaluate two scenarios: *Dynamic price* (Danish) and *Time-of-use tariff* (Brazilian). For the former we downloaded the real Danish hourly energy price between January 1st and June 16th of 2016 from NordPollSpot[5], and for the latter we use the Brazilian time-of-use tariff based on [9][6]. Because of the lack of an evaluation of the Brazilian driving pattern we apply the Brazilian tariff in the same Danish driving pattern. The following algorithms/strategies are evaluated:

- **MASCO** – Our proposal is evaluated with different user preferences, given by $\boldsymbol{w}$: (i) $MASCO_{pr}$ – the consumer is only concerned in minimizing her energy costs, hence $\boldsymbol{w} = [0, 1, 0]$; (ii) $MASCO_{bal}$ – the consumer is concerned equally in all objectives, thus $\boldsymbol{w} = [1, 1, 1]$; and (iii) $MASCO_{pt}$ – EVs are expected to minimize costs while avoiding overloads, thus $\boldsymbol{w} = [0, 1, 1]$. All experiments are executed with parameters $\alpha = 0.2$, $\gamma = 0.99$, $\varepsilon = 0.2$, and $C = 1$.
- **Always Charging Policy (ACP)** – Whenever plugged, all agents charge their batteries. ACP is expected to cause high demand peaks at when the consumers are coming back home.
- **Random Policy (RP)** – Each EV completely ignores all other agents and has 50% of probability of using the *charge* action. As the available time to recharge is more than enough to fill the agents batteries, this policy is expected to cause smaller demand peak than ACP while providing enough energy to daily travels. However, the energy costs should be only slightly better than ACP since the current energy price is ignored.
- **DWL** – DWL is a state-of-the-art Multiagent RL similar to MASCO. The difference between the two lies in the preference vector, which can not be specified in DWL, unlike MASCO that can define the vector according to consumer preferences. We execute all experiments with parameters $\alpha = 0.2$, $\gamma = 0.99$, $\varepsilon = 0.2$, and $C = 1$.

In this work we do not consider the domestic energy consumption and only evaluate the effect of these different policies on the EVs energy demand. We consider a ring topology, thus G is assembled as the previous and posterior agent, which means that EVs make their decision without information regarding the majority of agents in the system. Consequently, each agent has a pair of Q and W tables for selfish optimization and a pair for each agent in G.

For all algorithms, we execute 800 days of learning until evaluating the final policy. This procedure was repeated 25 times to define the average performance. As all agents recharged enough for their daily travel in all experiments and for all algorithms, here we omit further analysis regarding the state of charge and focus in metrics in which the algorithms achieved different performances.

All experiments were implemented in BURLAP[7].

A. Brazilian Tariff Experiment

Figure 2 shows the transformer load per hour in a day, after 800 days of training. ACP causes a high transformer load in the peak hours, which shows that “dumb” charging is not adequate and it is prone to cause instabilities in the grid. The peak hour is when most of EVs have already arrived home at approximately 7PM. RP avoids overloads because EVs are roughly taking turns to charge (50% charging probability while the battery is not full), causing a moderate size peak when all EVs are at home. DWL has a load profile similar to RP, with a moderate size load in the peak hour. $MASCO_{pr}$ agents avoid to charge right after arriving home, which is depicted by the low charge between 7PM and 11PM. However, when the lower energy price period begins at 11PM , agents instantaneously start to charge their batteries, causing a high energy demand until all agents have charged their batteries. This behavior reflects the consumer preference to buy energy in the cheapest period, however, it only shifted the transformer load peak to when the energy price is lower. $MASCO_{bal}$ causes a moderate peak when the EVs are arriving home but in average no overloads happen. The agents remain charging in a safe pace until nearly all EVs have their battery full, and a small peak is caused near 7AM by some EVs recharging when most cars have already gone to their daily journey. Finally, $MASCO_{pt}$ maintains a low load during the high cost period (6PM - 11PM). At 11PM the transformer load is increased but the overload limit is not exceeded in average. It is noteworthy that EVs behavior corresponds to the defined preferences in all cases. Also, when the preference takes into account the transformer load, MASCO is able to avoid overloads while allowing EVs to recharge their batteries.

Table I summarizes the performance of each algorithm. The numbers are averages from 25 executions with the 95% confidence interval. Numbers in bold and in red correspond to the best and the worst performance achieved by all algorithms in that metric, respectively. The algorithm name in bold corresponds to the best performance overall. ACP achieved the worst performance in both metrics, causing overloads and charging EVs with higher prices. RP does not causes many overloads, but the energy costs are worse than DWL and MASCO with any of the evaluated preferences. As noted in

[5] http://www.nordpoolspot.com/

[6] Cheapest period between 10PM and 5PM; intermediate period between 5PM and 6PM, and between 9PM and 10PM; and most expensive period between 6PM and 9PM.

[7] http://burlap.cs.brown.edu/index.html

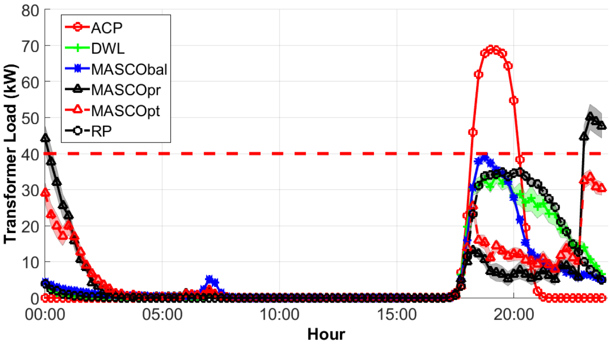

Fig. 2. Average Transformer Load observed in 5 days after 800 days of training. The shaded area represents the 95% confidence interval over 25 repetitions. The dashed line corresponds to the maximum desired load.

[23], DWL cannot be customized to meet consumer preferences, and by the observed results it chose to prioritize the battery and transformer objectives, which achieves results comparable with RP. $MASCO_{pr}$ achieved the cheapest energy price, however its number of overload is only lower than ACP. $MASCO_{pt}$ achieved a slightly higher cost, however, the number of overload was very low during the simulation, which is the best performance in load balancing among all the evaluated algorithms in this experiment (together with RP). The results achieved by $MASCO_{bal}$ indicate that EVs try to charge their batteries faster than $MASCO_{pt}$. As there is plenty of time to wait for a cheapest period, $MASCO_{bal}$ would only achieve the best performance if the consumer had the profile of using the EV for much more time during the day, situation in which a faster recharging would be needed.

TABLE I
AVERAGE ENERGY COSTS AND NUMBER OF OVERLOADS PER DAY AFTER TRAINING FOR THE BRAZILIAN TARIFF EXPERIMENT.

Alg.	Costs (R$)	Over.
ACP	4.07 ± 0.01	8.40 ± 0.21
RP	3.74 ± 0.02	$\mathbf{0.20 \pm 0.21}$
DWL	3.56 ± 0.04	0.72 ± 0.59
$MASCO_{pr}$	$\mathbf{1.97 \pm 0.10}$	5.00 ± 0.76
$MASCO_{bal}$	2.90 ± 0.07	1.08 ± 0.58
$\boldsymbol{MASCO_{pt}}$	2.33 ± 0.09	$\mathbf{0.20 \pm 0.27}$

In conclusion, $MASCO$ achieved the best performance among the algorithms in this experiment, and $MASCO_{pt}$ is the best option to balance load while minimizing costs.

B. Danish Tariff Experiment

Figure 3 depicts the observed transformer load per hour in a day after the training phase. ACP and RP have the same performance as in Section VII-A because their policies are fixed. DWL has a transformer load profile similar to RP, in which a moderate peak is caused when EVs come home. $MASCO_{pr}$ and $MASCO_{bal}$ caused a more accentuated load during peak hours. This outcome is unexpected, since this preference prioritizes only price but is not able to learn how to recharge in cheapest periods. It means that energy price is highly variable and agents take longer to learn the best charging hours. However, note that $MASCO_{pr}$ has more EVs recharging between 0AM and 4AM (the cheapest period). This means the algorithm is slowly learning how to reduce costs, even though the price is highly variable. $MASCO_{pt}$ causes the lower load during peak hours, slowly decreasing the load until 3AM, when all agents have recharged.

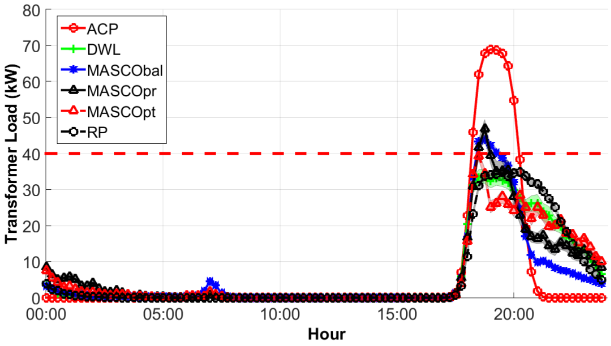

Fig. 3. Average Transformer Load observed in 5 days after 800 days of training. The shaded area represents the 95% confidence interval over 25 repetitions. The dashed line corresponds to the maximum desired load.

Table II summarizes results. The numbers are averages from 25 executions with the 95% confidence interval. Numbers in bold and in red correspond to the best and the worst performance achieved by all algorithms in that metric, respectively. The algorithm name in bold corresponds to the best performance overall. $MASCO_{pr}$ achieved worse overall results than RP, since a slightly lower price is achieved only through a high addition of overloads. RP achieved surprising results when compared to DWL and $MASCO_{pr}$, meaning that following only the price is ineffective and achieves results comparable to a random actuation. $MASCO_{bal}$ achieved the best performance in terms of cost, however, causing a high number of overloads. Finally, although very similar to DWL, $MASCO_{pt}$ achieves the best result overall.

TABLE II
AVERAGE ENERGY COSTS AND NUMBER OF OVERLOADS PER DAY AFTER TRAINING FOR THE DANISH TARIFF EXPERIMENT.

Alg.	Costs (DKK)	Over.
ACP	0.781 ± 0.003	8.40 ± 0.21
RP	0.726 ± 0.003	$\mathbf{0.20 \pm 0.21}$
DWL	0.713 ± 0.007	0.48 ± 0.38
$MASCO_{pr}$	0.727 ± 0.005	2.80 ± 0.84
$MASCO_{bal}$	$\mathbf{0.633 \pm 0.010}$	3.76 ± 0.67
$\boldsymbol{MASCO_{pt}}$	0.711 ± 0.005	$\mathbf{0.40 \pm 0.21}$

This two experiments show that MASCO is a robust architecture that achieved the best results amongst the evaluated algorithms in both tariffs. We conclude our article and discuss possible further work in the next section.

VIII. CONCLUSION AND FURTHER WORKS

In this work we proposed a *Multiagent System* architecture that allows EVs to recharge their batteries while minimizing energy costs and avoiding transformer overloads. Our architecture, called MASCO, is based on *Multiagent Reinforcement Learning* and learns how to alternate between a selfish policy that maximizes local objectives, and a collaborative policy that intends to help other agents to improve their performance.

MASCO makes minimal assumptions regarding the distribution grid and thus can be used together with many actual distribution infrastructures. Differently from many previous methods, MASCO does not require a manual setting of the required energy for the consumer daily travel, that is not assumed to be available to the EV. Also, MASCO can work in systems composed of agents following different strategies.

We experimentally showed that MASCO allows to balance energy costs and avoid transformer overloads, while following consumer preferences. In our experiments, MASCO achieved the best performance in both dynamic and time-of-use tariffs among the evaluated algorithms. This outcome shows that our architecture is a promising way to solve the Coordinated Charging Problem. This work can be extended along different lines, and future efforts can focus on:

- *Integration with Energy Management Systems (EMS)*: An EMS allows the consumer to generate, buy, and sell energy in order to minimize costs or profit from the energy market. EVs are adequate energy storage devices, and further work can propose an EMS that takes into account the problems here discussed, which are inherently associated with EVs, to plan and act.
- *Scaling Up MASCO*: MASCO is already a distributed and scalable approach, however, this can be further improved by the use of relational techniques to state space generalization [30], or value function approximation [31].
- *Reuse of Knowledge*: In a real-world situation, new EVs will join neighborhoods in which already trained agents are actuating. In this situation the knowledge from expert EVs can be shared with newcomers [32]. Taylor *et al.* [28] proposed a Transfer Learning [33] approach to share knowledge amongst agents, but this approach needs to be further improved to be used in real-world applications.

References

[1] Y. Cao, S. Tang, C. Li, P. Zhang, Y. Tan, Z. Zhang, and J. Li, "An Optimized EV Charging Model Considering TOU Price and SOC Curve," *IEEE Trans. Smart Grid*, vol. 3, no. 1, pp. 388–393, Mar. 2012.

[2] R. W. Uluski, "The Role of Advanced Distribution Automation in the Smart Grid," in *IEEE PES General Meeting*, July 2010, pp. 1–5.

[3] J. A. P. Lopes, F. J. Soares, and P. M. R. Almeida, "Integration of Electric Vehicles in the Electric Power System," *Proc. IEEE*, vol. 99, no. 1, pp. 168–183, Jan. 2011.

[4] S. Shao, T. Zhang, M. Pipattanasomporn, and S. Rahman, "Impact of TOU Rates on Distribution Load Shapes in a Smart Grid with PHEV Penetration," in *IEEE PES T&D*, Apr. 2010, pp. 1–6.

[5] E. L. Karfopoulos and N. D. Hatziargyriou, "A Multi-Agent System for Controlled Charging of a Large Population of Electric Vehicles," *IEEE Trans. Power Syst.*, vol. 28, no. 2, pp. 1196–1204, 2013.

[6] K. Clement-Nyns, E. Haesen, and J. Driesen, "The Impact of Charging Plug-In Hybrid Electric Vehicles on a Residential Distribution Grid," *IEEE Trans. Power Syst.*, vol. 25, no. 1, pp. 371–380, 2010.

[7] R. A. Waraich, M. D. Galus, C. Dobler, M. Balmer, G. Andersson, and K. W. Axhausen, "Plug-In Hybrid Electric Vehicles and Smart Grids: Investigations Based on a Microsimulation," *Transport. Res. C-Emer. Technol.*, vol. 28, pp. 74 – 86, 2013.

[8] J. Hu, A. Saleem, S. You, L. Nordstrm, M. Lind, and J. stergaard, "A Multi-Agent System for Distribution Grid Congestion Management with Electric Vehicles," *Eng. Appl. Artif. Intell.*, vol. 38, pp. 45 – 58, 2015.

[9] E. A. B. Bueno, W. Utubey, and R. R. Hostt, "Evaluating the Effect of the White Tariff on a Distribution Expansion Project in Brazil," in *IEEE Conf. On Innovative Smart Grid Technol. LATAM*, Apr. 2013, pp. 1–8.

[10] Y. Yang, Q.-S. Jia, G. Deconinck, X. Guan, Z. Qiu, and Z. Hu, "Distributed Coordination of EV Charging with Renewable Energy in a Microgrid of Buildings," *IEEE Trans. Smart Grid*, vol. 9, no. 6, pp. 6253–6264, Nov. 2018.

[11] C. Cao and B. Chen, "Generalized nash equilibrium problem based electric vehicle charging management in distribution networks," *Int. J. of Energy Res.*, vol. 42, no. 15, pp. 4584–4596, 2018.

[12] M. Vasirani and S. Ossowski, "A Proportional Share Allocation Mechanism for Coordination of Plug-In Electric Vehicle Charging," *Eng. Appl. Artif. Intell.*, vol. 26, no. 3, pp. 1185 – 1197, 2013.

[13] A. Ghosh and V. Aggarwal, "Control of Charging of Electric Vehicles Through Menu-Based Pricing," *IEEE Trans. Smart Grid*, vol. 9, no. 6, pp. 5918–5929, Nov. 2018.

[14] I. Dusparic, C. Harris, A. Marinescu, V. Cahill, and S. Clarke, "Multi-Agent Residential Demand Response Based on Load Forecasting," in *1st IEEE Conf. on Technol. for Sustainability*, Aug. 2013, pp. 90–96.

[15] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley, "A Survey of Multi-Objective Sequential Decision-Making," *Journal of Artif. Intell. Res.*, vol. 48, pp. 67–113, 2013.

[16] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 1st ed. Cambridge, MA, USA: MIT Press, 1998.

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

[18] A. Abels, D. M. Roijers, T. Lenaerts, A. Nowé, and D. Steckelmacher, "Dynamic weights in multi-objective deep reinforcement learning," in *Proc. 36th Int. Conf. on Mach. Learn.*, ser. Proc. of Mach. Learn. Res., K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. Long Beach, California, USA: PMLR, 09–15 Jun 2019, pp. 11–20. [Online]. Available: http://proceedings.mlr.press/v97/abels19a.html

[19] 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. Intel.*, vol. 29, pp. 134 – 151, 2014.

[20] M. L. Littman, "Markov Games as a Framework for Multi-Agent Reinforcement Learning," in *Proc. 11th Int. Conf. on Mach. Learn.*, 1994, pp. 157–163.

[21] L. Busoniu, R. Babuska, and B. De Schutter, "A Comprehensive Survey of Multiagent Reinforcement Learning," *IEEE Trans. Syst., Man, Cybern. Syst., Part C: Appl. and Rev.*, vol. 38, no. 2, pp. 156–172, 2008.

[22] I. Dusparic and V. Cahill, "Distributed W-Learning: Multi-Policy Optimization in Self-Organizing Systems," in *3rd IEEE Int. Conf. on Self-Adaptive and Self-Organizing Syst.*, Sep. 2009, pp. 20–29.

[23] F. L. D. Silva and A. H. R. Costa, "Multi-Objective Reinforcement Learning through Reward Weighting," in *Proc. 2nd Workshop on Synergies Between Multiagent Syst., Mach. Learn. and Complex Syst., at IJCAI*, vol. 1, 2015, pp. 25 – 36. [Online]. Available: http://www.ufrgs.br/tri2015/

[24] A. Messac, C. Puemi-Sukam, and E. Melachrinoudis, "Aggregate objective functions and pareto frontiers: required relationships and practical implications," *Optim. and Eng.*, vol. 1, no. 2, pp. 171–188, 2000.

[25] M. Humphrys, "Action selection methods using Reinforcement Learning," in *Proc. 4th Int. Conf. Simul. of Adapt. Behav.*, 1996, pp. 135–144.

[26] 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, June 2011.

[27] S. Garcia, J. Luengo, J. A. Sáez, V. Lopez, and F. Herrera, "A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 4, pp. 734–750, 2012.

[28] A. Taylor, I. Dusparic, E. Galvan-Lopez, S. Clarke, and V. Cahill, "Accelerating Learning in Multi-Objective systems through Transfer Learning," in *Int. Joint Conf. on Neural Netw.*, Jul 2014, pp. 2298–2305.

[29] Q. Wu, A. H. Nielsen, J. Ostergaard, S. T. Cha, F. Marra, Y. Chen, and C. Trholt, "Driving Pattern Analysis for Electric Vehicle (EV) Grid Integration Study," in *IEEE PES Innovative Smart Grid Technol. Conf. Europe*, Oct. 2010.

[30] P. Tadepalli, R. Givan, and K. Driessens, "Relational Reinforcement Learning: An Overview," in *In Proc. of the Workshop on Relational Reinforcement Learning*, 2004.

[31] M. Geist and O. Pietquin, "Algorithmic Survey of Parametric Value Function Approximation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 6, pp. 845–867, 2013.

[32] F. L. D. Silva, R. Glatt, and A. H. R. Costa, "Simultaneously Learning and Advising in Multiagent Reinforcement Learning," in *Proc. 16th Int. Conf. on Auton. Agents and Multiagent Syst.*, 2017, pp. 1100–1108.

[33] F. L. D. Silva and A. H. R. Costa, “A Survey on Transfer Learning for Multiagent Reinforcement Learning Systems,” *J. Artif. Intell. Res.*, vol. 64, pp. 645–703, 2019.