2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

A Scale-Independent Multi-Objective Reinforcement Learning with Convergence Analysis

Mohsen Amidzadeh, *Member, IEEE*
Department of Computer Science, Aalto Universiy, Finland
mohsen.amidzade@aalto.fi

***Abstract*—Many sequential decision-making problems need optimization of different objectives which possibly conflict with each other. The conventional way to deal with a multi-task problem is to establish a scalar objective function based on a linear combination of different objectives. However, for the case where we have conflicting objectives with different scales, this method needs a trial-and-error approach to properly find proper weights for the combination. As such, in most cases, this approach cannot guarantee an optimal Pareto solution. In this paper, we develop a single-agent scale-independent multi-objective reinforcement learning on the basis of the Advantage Actor-Critic (A2C) algorithm. A convergence analysis is then done for the devised multi-objective algorithm providing a convergence-in-mean guarantee. We then perform some experiments over a multi-task problem to evaluate the performance of the proposed algorithm. Simulation results show the superiority of developed multi-objective A2C approach against the single-objective algorithm.**

***Index Terms*—Multi-objective reinforcement learning, advantage actor-critic algorithm, convergence analysis.**

I. Introduction

Many sequential decision-making problems include multiple objective functions competing with each other. The common approach to finding an optimum solution for these problems is a scalarization approach based on considering a preference for different objectives. However, the Pareto solutions cannot be obtained via this method [1]. As such, a trial-and-error approach might be needed to tune optimum scalarization settings. This difficulty also appears for the sequential decision-making problems being modeled based on a Multi-Objective Markov Decision Process (MO-MDP) [2]. To find an optimal policy for a MO-MDP with different tasks, a Multi-Objective Reinforcement Learning (MO-RL) algorithm needs to be developed. A typical method for MO-RLs is the scalarization approach to first construct a scalar reward based on a combination of competing rewards (either linear or non-linear), and then apply a single-objective RL algorithm [3], [4]. However, this approach mainly makes the solution highly dependent on the selected combination.

Most developed MO-RL algorithms are restricted to the discrete environment. [5] consider a multi-objective Bellman operator by which a value-based reinforcement learning algorithm is devised to obtain the Pareto solutions in a discrete environment. [6] develop a MO-RL algorithm on the basis of deep Q-learning and optimistic linear support learning. They consider a scalarized vector and potential optimal solutions to have a convex combination of the objectives. However, they need to search over all potential scalarizing vectors as the importance of distinct objectives is not a priori knowledge. [7] leverage a multi-objective variant of Q-learning with a single-agent approach in order to learn a preference-based adjustment being generalized across different preferences. However, they use a convex envelope of the Pareto frontier during the updating process. This is often sample inefficient and leads to a sub-optimal policy, though it is efficient from the computational complexity viewpoint. Apart from the approach for discrete state-action spaces, there exist MO-RL algorithms devised for the continuous environment. [8] devise a MO-RL by meta-learning. They learned a meta-policy distribution trained with multiple tasks. The multi-objective problem is converted to a number of single-objective problems using a parametric scalarizing function. However, the solution depends on the distribution based on which the parameters of the scalarizing function are drawn. Reward-specific state-value functions are formulated based on a correlation matrix to indicate the relative importance of the objectives on each other [9]. However, they need to tune this matrix weight to find the proper inter-objective relationship. [10] develop a MO-RL framework based on the *maximum a posteriori policy optimization* algorithm [11]. They learned objective-specific policy distributions to find the Pareto solutions in a scale-invariant manner. However, they still need to adjust objective-specific coefficients controlling the influence of objectives on the policy update.

In this paper, we propose a MO-RL algorithm for the continuous-valued state-action spaces without considering any preferences for different objectives. In contrast to [10], [9], [8], a single-policy approach is devised which simplifies the algorithm architecture, and additionally there is no need for an initial assumption for reward preference. As such, the devised algorithm can be considered scale-invariant.

The contributions of this paper are summarized as follows:
i. We devise a single-policy multi-objective RL algorithm without the importance of the competing objectives available as a priori knowledge. We develop our algorithm on the basis of Advantage Actor-Critic (A2C) [12] using reward-specific state-value functions.
ii. We provide a convergence analysis of the proposed scale-invariant MO-RL algorithm.
iii. We evaluate the devised algorithm over a multi-task problem. Results show that it outperforms the single-objective A2C algorithm with a scalar reward from the sample-

979-8-3503-0123-6/23/$31.00 ©2023 IEEE

efficiency and scale-invariance perspectives.

Notations: In this paper, we mainly use lower-case a for scalars, bold-face lower-case $\mathbf{a}$ for vectors and bold-face upercase $\mathbf{A}$ for matrices. Further, $\mathbf{A}^\top$ is the transpose of $\mathbf{A}$, $\|\mathbf{a}\|$ and $\|\mathbf{A}\|$ are the euclidean norm of $\mathbf{a}$ and the corresponding induced matrix norm of $\mathbf{A}$, respectively, and $\boldsymbol{\nabla}_{\mathbf{a}} g(\cdot)$ is the gradient vector of multivariate function $g(\cdot)$ with respect to (w.r.t.) vector $\mathbf{a}$. We indicate the m-th element of the vector $\boldsymbol{a}$ by a_m. Further, $\{a_m\}_1^n$ collects the components of vector $\boldsymbol{a}$ from $m=1$ to $m=n$. We use $\mathbf{I}_n$, $\mathbf{1}_n$, $\mathbf{0}$ and $\mathbf{e}_m$ to denote the identity matrix of size $n \times n$, a n-dimensional vector with all elements equal to one, a vector with all elements equal to zero, and a vector with all elements being zero except the m-th element that is one, respectively.

II. Background

A. Pareto Optimality

We brief the notion of Pareto optimality for a multi-objective optimization (MOO) problem. For this, consider the following unconstrained problem:

$$Q_1: \quad \min_{x \in \mathcal{X}} \quad [f_1(\boldsymbol{x}), \ldots, f_r(\boldsymbol{x})],$$

where $f_j : \mathbb{R}^N \to \mathbb{R}$, $\mathcal{X}$ is the feasible set, and r is the number of objectives. Then, $\boldsymbol{x}^* \in \mathcal{X}$ is called a Pareto optimal solution of Q_1, if there is no other solution like $\boldsymbol{y} \in \mathcal{X}$ so as to dominates $\boldsymbol{x}^*$, i.e., $f_i(\boldsymbol{y}) \leq f_i(\boldsymbol{x}^*)$ for all $i \in \{1, \ldots, r\}$ and there is one j so that $f_j(\boldsymbol{y}) < f_j(\boldsymbol{x}^*)$.

For MOO Q_1, if there exists a vector $\boldsymbol{\alpha} \in [0,1]^r$ with $\sum_{j=1}^r \alpha_j = 1$ so that:

$$\sum_{j=1}^r \alpha_j \nabla f_j(\hat{\boldsymbol{x}}) = 0,$$

then $\hat{\boldsymbol{x}}$ is a Pareto optimal solution.

We now mention the following Lemma [13], [14]:

Lemma II.1. *Assume a vector-valued multivariate function $\boldsymbol{f} = (f_1, \ldots, f_r)$, $f_j : \mathbb{R}^n \to \mathbb{R}$ for $j \in \{1, \ldots, r\}$. Define $\boldsymbol{q}(\cdot) = \sum_{j=1}^r \alpha_j^* \nabla f_j(\cdot)$, then $-\boldsymbol{q}(\cdot)$ is a descent direction for all functions $\{f_j(\cdot)\}_1^r$, where $\{\alpha_j^*\}$ are the solution of the following optimization problem:*

$$Q_2: \min_{\{\alpha_j\}_1^r} \quad \Big\| \sum_{j=1}^r \alpha_j \nabla f_j(\cdot) \Big\|^2,$$
$$\text{s.t.} \quad \sum_{j=1}^r \alpha_j = 1, \quad \alpha_j \geq 0 \quad \text{for } j \in \{1, \ldots, r\}.$$

Accordingly, we can get:

Corollary II.1. *The solution of Q_1, for all $\alpha_j \geq 0$, reads:*

$$\boldsymbol{\alpha}^* = \frac{\big(\nabla F(\cdot)^\top \nabla F(\cdot)\big)^{-1} \mathbf{1}_r}{\mathbf{1}_r^\top \big(\nabla F(\cdot)^\top \nabla F(\cdot)\big)^{-1} \mathbf{1}_r}, \tag{1}$$

where $\nabla F(\cdot)$ is an $n \times r$ matrix with $\nabla F(\cdot) = [\nabla f_1, \ldots, \nabla f_r](\cdot)$.

B. Multi-Objective Markov Decision Process

A Multi-Objective Markov Decision Process (MO-MDP) is expressed according to the tuple $\big(\mathcal{S}, \mathcal{A}, P_{\mathcal{T}}(\cdot), \{r_j(\cdot)\}_1^r\big)$, where $\mathcal{S}$ is a set of states or the state space, $\mathcal{A}$ is a set of actions or the action space, $P_{\mathcal{T}}(\cdot) : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0,\ 1]$ is the transition probability describing the system environment, and $r_j(\cdot) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, for $j \in \{1, \ldots, r\}$, is the j-th immediate reward function. The system state and action, at time t, are denoted by $\boldsymbol{s}_t \in \mathcal{S}$ and $\boldsymbol{a}_t \in \mathcal{A}$, respectively. The transition probability $P_{\mathcal{T}}(\boldsymbol{s}_{t+1} | \boldsymbol{s}_t, \boldsymbol{a}_t)$ shows the probability that being in state $\boldsymbol{s}_t$ and performing action $\boldsymbol{a}_t$ leads to the next state $\boldsymbol{s}_{t+1}$. Therefore, we have: $\boldsymbol{s}_{t+1} \sim P_{\mathcal{T}}(\cdot | \boldsymbol{s}_t, \boldsymbol{a}_t)$. The reward function $r_j(\boldsymbol{s}_t, \boldsymbol{a}_t)$ indicates the j-th immediate reward being obtained by transitioning from state $\boldsymbol{s}_t$ to state $\boldsymbol{s}_{t+1}$ by acting $\boldsymbol{a}_t$.

In this work, we are interested in a stochastic policy representation. For this, the action $\boldsymbol{a}_t$ is determined by drawing from a conditional policy distribution $\pi(\cdot | \boldsymbol{s}_t)$, i.e., $\boldsymbol{a}_t \sim \pi(\cdot | \boldsymbol{s}_t)$. Based on the Markov property, the probability of a trajectory $\tau : \boldsymbol{s}_1 \to \boldsymbol{a}_1 \to \boldsymbol{s}_2 \to \boldsymbol{a}_2 \to \ldots \to \boldsymbol{s}_{T+1}$, is determined by:

$$\begin{aligned} \mathsf{P}(\tau) :=& \mathsf{P}(\boldsymbol{s}_1, \boldsymbol{a}_1, \boldsymbol{s}_2, \boldsymbol{a}_2, \ldots, \boldsymbol{s}_{T+1}) \\ =& \mathsf{P}(\boldsymbol{s}_1) \prod_{t=1}^T \pi(\boldsymbol{a}_t | \boldsymbol{s}_t) P_{\mathcal{T}}(\boldsymbol{s}_{t+1} | \boldsymbol{a}_t, \boldsymbol{s}_t). \end{aligned} \tag{2}$$

In MO-MDP problems, the cumulative discounted rewards $\{R_j(t)\}_1^r$ are defined based on a summation over a finite horizon T as:

$$R_j(t) := \mathbb{E}\left\{ \sum_{k=t}^T \gamma^{k-t} r_j(\boldsymbol{s}_k, \boldsymbol{a}_k) \right\},$$

where the expectation is with respect to $\mathsf{P}(\tau)$ and $\gamma :\ 0 < \gamma \leq 1$ is the discount factor. The aim is to find a single-agent stochastic policy, such that the cumulative discounted rewards $\{R_j(t)\}_1^r$ are minimized.

$$\begin{aligned} P_1: \quad & \min_{\pi(\cdot | \boldsymbol{s}_t)} \quad \big[R_1(t), \ldots, R_r(t)\big], \quad 0 \leq t \leq T \\ & \text{s.t.} \quad \boldsymbol{a}_t \sim \pi(\cdot | \boldsymbol{s}_t) \\ & \text{s.t.} \quad \boldsymbol{s}_{t+1} \sim P_{\mathcal{T}}(\cdot | \boldsymbol{s}_t, \boldsymbol{a}_t). \end{aligned} \tag{3}$$

Notice that P_1 is a MOO problem and as such the minimization is regarded from the Pareto optimality perspective.

C. A2C Algorithm

Here, we address the structure of A2C [12] as the basis of the multi-objective reinforcement algorithm we intend to devise. For the standard A2C algorithm, a single-objective MDP with a single immediate reward function $r(\cdot) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is taken into account, so $r = 1$. The aim is to design an optimal policy distribution being parameterized by a parameter $\boldsymbol{\theta} \in \Theta$, where Θ is a parameterization set of interest. Here, we use the notation $\pi_{\boldsymbol{\theta}}(\cdot, \cdot)$ to show this parametric policy distribution. Consequently, the probability of trajectory (2) depends on $\boldsymbol{\theta}$, and can be expressed as:

$$\mathsf{P}_{\boldsymbol{\theta}}(\tau) = \mathsf{P}(\boldsymbol{s}_1) \prod_{t=1}^T \pi_{\boldsymbol{\theta}}(\boldsymbol{a}_t | \boldsymbol{s}_t) P_{\mathcal{T}}(\boldsymbol{s}_{t+1} | \boldsymbol{a}_t, \boldsymbol{s}_t). \tag{4}$$

Now, the objective can be expressed by $J(\boldsymbol{\theta}) := \mathbb{E}_{\mathsf{P}_{\boldsymbol{\theta}}(\tau)}\{\sum_{k=1}^{T} r(\boldsymbol{s}_k, \boldsymbol{a}_k)\}$ that needs to be minimized with respect to policy distribution $\pi_{\boldsymbol{\theta}}(\cdot|\cdot)$. Computing the gradient of $J(\boldsymbol{\theta})$ gives [12]:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbb{E}\left\{ \sum_{k=1}^{T} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\boldsymbol{a}_k|\boldsymbol{s}_k) A(\boldsymbol{s}_k, \boldsymbol{a}_k) \right\}, \quad (5)$$

where $V(\boldsymbol{s}_k) = \mathbb{E}\{\sum_{k'=k}^{T} r(\boldsymbol{s}_{k'}, \boldsymbol{a}_{k'}) \big| \boldsymbol{s}_k\} : \mathcal{S} \to \mathbb{R}$, $Q(\boldsymbol{s}_k, \boldsymbol{a}_k) = \mathbb{E}\{\sum_{k'=k}^{T} r(\boldsymbol{s}_{k'}, \boldsymbol{a}_{k'}) \big| \boldsymbol{s}_k, \boldsymbol{a}_k\} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and $A(\boldsymbol{s}_k, \boldsymbol{a}_k) = r(\boldsymbol{s}_k, \boldsymbol{a}_k) + \gamma V(\boldsymbol{s}_{k+1}) - V(\boldsymbol{s}_k) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, are the state-value, action-value, and advantage functions, respectively. Note that the policy distribution $\pi_{\boldsymbol{\theta}}(\cdot|\cdot)$ with random parameter $\boldsymbol{\theta}$ is managed by an actor agent which can employ a neural network to generate action based on a given state.

Here, it is of benefit to remark on two practical points of the A2C algorithm. First, a Stochastic Gradient Descent (SGD) is applied in A2C, where the parameter $\boldsymbol{\theta}$ is updated by the actor agent based on the following rule:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \mu_a \nabla_{\boldsymbol{\theta}} \hat{J}(\boldsymbol{\theta}),$$

where μ_a is the actor learning rate and $\nabla_{\boldsymbol{\theta}} \hat{J}(\boldsymbol{\theta}) = \sum_{k=1}^{T} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\boldsymbol{a}_k|\boldsymbol{s}_k) A(\boldsymbol{s}_k, \boldsymbol{a}_k)$. Additionally, it is conventional to represent the state-value function $V(\boldsymbol{s}_k)$ by a $\boldsymbol{\phi}$-parametric approximation $V_{\boldsymbol{\phi}}(\boldsymbol{s}_k)$ with $\boldsymbol{\phi} \in \Phi$, where Φ is a parameterization set of interest. A neural network with random parameter $\boldsymbol{\phi}$ can be employed by a critic agent for this representation. Accordingly, the advantage function can be represented by $A_{\boldsymbol{\phi}}(\boldsymbol{s}_k, \boldsymbol{a}_k) := r(\boldsymbol{s}_k, \boldsymbol{a}_k) + \gamma V_{\boldsymbol{\phi}}(\boldsymbol{s}_{k+1}) - V_{\boldsymbol{\phi}}(\boldsymbol{s}_k)$. Then, based on the Bellman's equation $V(\boldsymbol{s}_k) = \mathbb{E}_{\boldsymbol{s}_{k+1}, \boldsymbol{a}_k|\boldsymbol{s}_k}\{r(\boldsymbol{s}_k, \boldsymbol{a}_k) + \gamma V(\boldsymbol{s}_{k+1})\}$, with $\mathbb{E}_{\boldsymbol{s}_{k+1}, \boldsymbol{a}_k|\boldsymbol{s}_k}\{\cdot\} = \iint (\cdot) \pi_{\boldsymbol{\theta}}(\boldsymbol{a}_k|\boldsymbol{s}_k) P_{\mathcal{T}}(\mathbf{s}_{k+1}|\mathbf{a}_k, \mathbf{s}_k) d\mathbf{a}_k d\mathbf{s}_{k+1}$, the following objective, called critic loss, is regarded to update parameter $\boldsymbol{\phi}$:

$$\sum_{k=1}^{T} \left(\mathbb{E}_{\boldsymbol{s}_{k+1}, \boldsymbol{a}_k|\boldsymbol{s}_k} \{ r(\boldsymbol{s}_k, \boldsymbol{a}_k) + \gamma V(\boldsymbol{s}_{k+1}) - V_{\boldsymbol{\phi}}(\boldsymbol{s}_k) \} \right)^2.$$

However, in practice, the following SGD approach is used:

$$\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} - \mu_c \sum_{k=1}^{T} A_{\boldsymbol{\phi}}(\boldsymbol{s}_k, \boldsymbol{a}_k) \nabla_{\boldsymbol{\phi}} V_{\boldsymbol{\phi}}(\boldsymbol{s}_k),$$

where μ_c is the critic learning rate. It is noteworthy that the update process of the actor agent can also be expressed based on the advantage function as:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \mu_a \sum_{k=1}^{T} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\boldsymbol{a}_k|\boldsymbol{s}_k) A_{\boldsymbol{\phi}}(\boldsymbol{s}_k, \boldsymbol{a}_k). \quad (6)$$

Here, we need to emphasize that environment stochasticity affects the values of θ and ϕ, so we treat them as random variables.

III. Multi-Objective A2C Algorithm

In the sequels, we develop a single-agent multi-objective A2C algorithm based on the result of Lemma II.1. We call the proposed algorithm *MO-A2C*. For this, we formulate multi-objective actor (MO-actor) and multi-objective critic (MO-critic) agents as follows:

A. *MO-Critic Agent*

Consider a MO-MDP with immediate reward functions $\{r_j(\cdot)\}_1^r$. We formulate a MO-critic agent applying a shared $\boldsymbol{\phi}$-parametric neural network in order to learn multiple state-value functions $\{V_{\boldsymbol{\phi},j}(\cdot)\}_1^r$ corresponding to rewards $\{r_j(\cdot)\}_1^r$. We thus introduce the j-th advantage function:

$$A_{\boldsymbol{\phi},j}(\boldsymbol{s}_t, \boldsymbol{a}_t) = r_j(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma V_{\boldsymbol{\phi},j}(\boldsymbol{s}_{t+1}) - V_{\boldsymbol{\phi},j}(\boldsymbol{s}_t).$$

Then, based on the Lemma II.1, we establish the following reward-specific MO-critic loss:

$$\hat{J}_{\mathrm{cr,j}}(\boldsymbol{\phi}) = \sum_{k=1}^{T} A_{\boldsymbol{\phi},j}^2(\boldsymbol{s}_k, \boldsymbol{a}_k), \quad (7)$$

with the MO-critic agent updating $\boldsymbol{\phi}$ by the rule:

$$\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} - \mu_c \sum_{j=1}^{r} \alpha_{\mathrm{cr,j}} \nabla_{\boldsymbol{\phi}} \hat{J}_{\mathrm{cr,j}}(\boldsymbol{\phi}), \quad (8)$$

where $\nabla_{\boldsymbol{\phi}} \hat{J}_{\mathrm{cr,j}}(\boldsymbol{\phi}) = -\sum_{k=1}^{T} A_{\boldsymbol{\phi},j}(\boldsymbol{s}_k, \boldsymbol{a}_k) \nabla_{\boldsymbol{\phi}} V_{\boldsymbol{\phi},j}(\boldsymbol{s}_k)$ and $\{\alpha_{\mathrm{cr,j}}\}_1^r$ are obtained by:

$$\boldsymbol{\alpha}_{\mathrm{cr}} = \underset{\substack{\{\alpha_j \geq 0\}_1^r \\ \sum_{j=1}^{r} \alpha_j = 1}}{\operatorname{argmin}} \left\| \sum_{j=1}^{r} \alpha_j \nabla_{\boldsymbol{\phi}} \hat{J}_{\mathrm{cr,j}}(\boldsymbol{\phi}) \right\|^2. \quad (9)$$

B. *MO-Actor Agent*

For the MO-actor agent, we consider a $\boldsymbol{\theta}$-parametric neural network, which receives $\boldsymbol{s}_k$ and outputs the policy distribution $\pi_{\boldsymbol{\theta}}(\cdot|\boldsymbol{s}_k)$, from which the action vector $\boldsymbol{a}_k$ is drawn. Accordingly, the following approach, which slightly differs from the MO-critic updating procedure, is devised for the MO-actor agent. Inspired by Eqs. (5) and (6), for the j-th reward-specific loss we have:

$$\nabla_{\boldsymbol{\theta}} \hat{J}_{\mathrm{ac,j}}(\boldsymbol{\theta}, \boldsymbol{\phi}) = -\sum_{k=1}^{T} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\boldsymbol{a}_k|\boldsymbol{s}_k) A_{\boldsymbol{\phi},j}(\boldsymbol{s}_k, \boldsymbol{a}_k), \quad (10)$$

and the MO-actor agent updates $\boldsymbol{\theta}$ by the rule:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \mu_a \sum_{j=1}^{r} \alpha_{\mathrm{ac,j}} \nabla_{\boldsymbol{\theta}} \hat{J}_{\mathrm{ac,j}}(\boldsymbol{\theta}, \boldsymbol{\phi}), \quad (11)$$

where $\{\alpha_{\mathrm{ac,j}}\}_1^r$ are found by:

$$\boldsymbol{\alpha}_{\mathrm{ac}} = \underset{\substack{\{\alpha_j \geq 0\}_1^r \\ \sum_{j=1}^{r} \alpha_j = 1}}{\operatorname{argmin}} \left\| \sum_{j=1}^{r} \alpha_j \nabla_{\boldsymbol{\theta}} J_{\mathrm{ac,j}}^{\phi}(\boldsymbol{\theta}) \right\|^2, \quad (12)$$

with

$$\nabla_{\boldsymbol{\theta}} J_{\mathrm{ac,j}}^{\phi}(\boldsymbol{\theta}) := -\mathbb{E}_{\boldsymbol{\phi}}\, \mathbb{E} \left\{ \sum_{k=1}^{T} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\boldsymbol{a}_k|\boldsymbol{s}_k) A_{\boldsymbol{\phi},j}(\boldsymbol{s}_k, \boldsymbol{a}_k) \right\}. \quad (13)$$

Note that, in contrast to MO-critic loss (see Eq. (9)), we exploit the expected loss $\nabla_{\boldsymbol{\theta}} J_{\mathrm{ac,j}}^{\phi}(\boldsymbol{\theta})$ to optimize $\boldsymbol{\alpha}_{\mathrm{ac}}$ in Eq. (12). To estimate it, we use a *moving average* of $\nabla_{\boldsymbol{\theta}} \hat{J}_{\mathrm{ac,j}}(\boldsymbol{\theta}, \boldsymbol{\phi})$ over different episodes and name it as **episodic average**.

Figure 1 illustrates the diagram of devised multi-objective A2C algorithm. A pseudo-code of this algorithm is also shown in Algorithm 1 with i being the iteration or episode index, $E_{\max}$ being the total number of episodic realizations the algorithm is learned for, and $\mathcal{I} = \{1, \ldots, E_{\max}\}$ is the set of episodes.

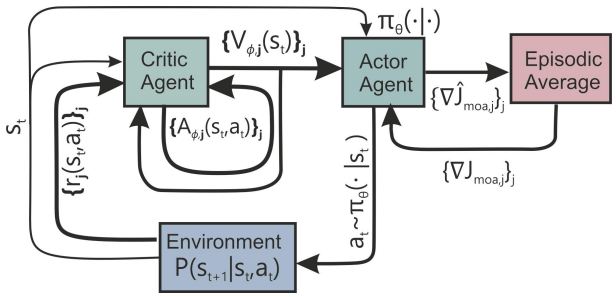

Fig. 1: The diagram of proposed *MO-A2C* algorithm. A thick line shows reward-specific information flows while a thin line indicates a single information flow.

Algorithm 1: Pseudo-Code of *MO-A2C*.

for $i \in \mathcal{I} := \{1, \dots, E_{\max}\}$ **do**
  Input: Initial state vector $\boldsymbol{s}_0$, MO-actor and MO-critic agents parameterized by $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$.
  for $t = 1$ ***to*** T **do**
    Select an action $\boldsymbol{a}_t$ following $\pi_{\boldsymbol{\theta}}(\cdot|\boldsymbol{s}_t)$, interact with the environment.
    Observe new state $\boldsymbol{s}_{t+1}$ and immediate rewards $\{r_j(\boldsymbol{s}_t, \boldsymbol{a}_t)\}_1^r$.
    Computes $\{A_{\boldsymbol{\phi},j}(\boldsymbol{s}_t, \boldsymbol{a}_t)\}_1^r$ using estimated Value-functions $\{V_{\boldsymbol{\phi},j}(\boldsymbol{s}_t)\}_1^r$ and based on Eq. (7).
    Buffer $\{V_{\boldsymbol{\phi},j}(\boldsymbol{s}_t)\}_1^r$, $\{A_{\boldsymbol{\phi},j}(\boldsymbol{s}_t, \boldsymbol{a}_t)\}_1^r$, $\{r_j(\boldsymbol{s}_t, \boldsymbol{a}_t)\}_1^r$ and $\log\big(\pi_{\boldsymbol{\theta}}(\boldsymbol{a}_t|\boldsymbol{s}_t)\big)$.
    Compute episodic average of MO-actor losses $\{\nabla \hat{J}_{\mathrm{ac,j}}\}_1^r$ to estimate expected losses $\{\nabla J_{\mathrm{ac,j}}\}_1^r$.
    if *update is needed,* **then**
      MO-critic update process:
      Obtain $\boldsymbol{\alpha}_{\mathrm{cr}}$ based on Eq. (9).
      Compute MO-critic losses $\{\hat{J}_{\mathrm{cr,j}}(\boldsymbol{\phi})\}_1^r$ and apply the rule:
      $$\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} - \mu_c \sum_{j=1}^{r} \alpha_{\mathrm{cr,j}} \nabla_{\boldsymbol{\phi}} \hat{J}_{\mathrm{ac,j}}(\boldsymbol{\phi}).$$
      MO-actor update process:
      Obtain $\boldsymbol{\alpha}_{\mathrm{ac}}$ based on Eq. (12).
      Eestimate $\{\nabla J_{\mathrm{ac,j}}\}_1^r$ by **episodic averaging**.
      Compute the gradient of MO-actor losses $\{\nabla_{\boldsymbol{\theta}} \hat{J}_{\mathrm{ac,j}}(\boldsymbol{\theta}, \boldsymbol{\phi})\}_1^r$ and apply the rule:
      $$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \mu_a \sum_{j=1}^{r} \alpha_{\mathrm{ac,j}} \nabla_{\boldsymbol{\theta}} \hat{J}_{\mathrm{ac,j}}(\boldsymbol{\theta}, \boldsymbol{\phi}).$$
    end
  end
end

IV. Convergence Analysis of MO-A2C Algorithm

Here, we intend to analyze the convergence of the proposed *MO-A2C* algorithm. For this, we first make some assumptions:

Assumption 1: The ϕ-parametric state-value functions are unbiased, i.e.,

$$\mathbb{E}_{\boldsymbol{\phi}}\{V_{\boldsymbol{\phi},j}(\boldsymbol{s}_k)\} = V_j(\boldsymbol{s}_k), \qquad j \in \{1, \dots, r\}.$$

According to this assumption and Eqs. (10) and (13) we can get:

$$\begin{aligned}\nabla_{\boldsymbol{\theta}} J^{\phi}_{\mathrm{ac,j}}(\boldsymbol{\theta}) &= \mathbb{E}_{\boldsymbol{\phi}}\, \mathbb{E}\left\{\nabla_{\boldsymbol{\theta}} \hat{J}_{\mathrm{ac,j}}(\boldsymbol{\theta}, \boldsymbol{\phi}) \,\middle|\, \boldsymbol{\theta}, \boldsymbol{\phi}\right\} \\ &= -\sum_{k=1}^{T} \mathbb{E}_{\boldsymbol{\phi}}\, \mathbb{E}\left\{\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\boldsymbol{a}_k|\boldsymbol{s}_k) A_{\boldsymbol{\phi},j}(\boldsymbol{s}_k, \boldsymbol{a}_k)\right\} \\ &= -\mathbb{E}\left\{\sum_{k=1}^{T} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\boldsymbol{a}_k|\boldsymbol{s}_k) A_j(\boldsymbol{s}_k, \boldsymbol{a}_k)\right\} := \nabla_{\boldsymbol{\theta}} J_{\mathrm{ac,j}}(\boldsymbol{\theta}).\end{aligned}$$

Inspired by this result, we define the MO-actor expected losses as follows:

$$J_{\mathrm{ac,j}}(\boldsymbol{\theta}) := -\mathbb{E}\left\{\sum_{k=1}^{T} \log \pi_{\boldsymbol{\theta}}(\boldsymbol{a}_k|\boldsymbol{s}_k) A_j(\boldsymbol{s}_k, \boldsymbol{a}_k)\right\},\ j = \{1, \dots, r\}.$$

Note that $J_{\mathrm{ac,j}}(\boldsymbol{\theta})$ differs from the original loss $-\mathbb{E}_{\mathsf{P}_{\boldsymbol{\theta}}(\tau)}\big\{\sum_{k=1}^{T} r_j(\boldsymbol{s}_k, \boldsymbol{a}_k)\big\}$ merely due to the terms not depending on $\boldsymbol{\theta}$. Furthermore, this distinction does not impose any restrictions on the upcoming analysis.

We now take into account two conventional assumptions based on the literature [15].

Assumption 2: The MO-actor expected losses $\{J_{\mathrm{ac,j}}(\boldsymbol{\theta})\}_1^r$ are Γ-strongly convex w.r.t $\boldsymbol{\theta}$. As such we get:

$$J_{\mathrm{ac,j}}(\boldsymbol{\theta}') - J_{\mathrm{ac,j}}(\boldsymbol{\theta}) \geq \nabla_{\boldsymbol{\theta}} J_{\mathrm{ac,j}}(\boldsymbol{\theta})^{\top}(\boldsymbol{\theta}' - \boldsymbol{\theta}) + \frac{\Gamma}{2}\|\boldsymbol{\theta}' - \boldsymbol{\theta}\|^2.$$

Furthermore, they are Lipschitz continuous functions with constant L w.r.t $\boldsymbol{\theta}$, so we have:

$$J_{\mathrm{ac,j}}(\boldsymbol{\theta}') - J_{\mathrm{ac,j}}(\boldsymbol{\theta}) \leq \nabla_{\boldsymbol{\theta}} J_{\mathrm{ac,j}}(\boldsymbol{\theta})^{\top}(\boldsymbol{\theta}' - \boldsymbol{\theta}) + \frac{L}{2}\|\boldsymbol{\theta}' - \boldsymbol{\theta}\|^2.$$

Notice that Assumptions 2 is made for the expected losses $\{J_{\mathrm{ac,j}}(\boldsymbol{\theta})\}_1^r$ and not for the stochastic losses $\{\hat{J}_{\mathrm{ac,j}}(\boldsymbol{\theta}, \boldsymbol{\phi})\}_1^r$.

Assumption 3: Consider the Jacobian matrix $\nabla \hat{\boldsymbol{J}}(\boldsymbol{\theta}, \boldsymbol{\phi}) = [\nabla_{\boldsymbol{\theta}} \hat{J}_{\mathrm{ac},1}, \dots, \nabla_{\boldsymbol{\theta}} \hat{J}_{\mathrm{ac,r}}](\boldsymbol{\theta}, \boldsymbol{\phi})$ where $\nabla \boldsymbol{J}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{\phi}}\, \mathbb{E}\left\{\nabla \hat{\boldsymbol{J}}(\boldsymbol{\theta}, \boldsymbol{\phi}) \,\middle|\, \boldsymbol{\theta}, \boldsymbol{\phi}\right\}$. Then, its conditional covariance is bounded by a positive semi-definite matrix $\mathbf{B}$:

$$\mathbb{E}_{\boldsymbol{\phi}}\, \mathbb{E}\left\{\nabla \hat{\boldsymbol{J}}(\boldsymbol{\theta}, \boldsymbol{\phi})^{\top} \nabla \hat{\boldsymbol{J}}(\boldsymbol{\theta}, \boldsymbol{\phi}) \,\middle|\, \boldsymbol{\theta}, \boldsymbol{\phi}\right\} - \nabla \boldsymbol{J}(\boldsymbol{\theta})^{\top} \nabla \boldsymbol{J}(\boldsymbol{\theta}) \leq \mathbf{B}.$$

This assumption indicates that the covariance matrix of Jacobian of the stochastic losses is upper-bounded by $\mathbf{B}$.

Now, we have the following theorems for the *MO-A2C* algorithm:

Theorem IV.1. *Consider expected losses $\{J_{\mathrm{ac,j}}(\cdot)\}_1^r$ and stochastic losses $\{\hat{J}_{\mathrm{ac,j}}(\cdot, \cdot)\}_1^r$ complying with Assumptions 2 and 3, and $\boldsymbol{\alpha}_{\mathrm{ac}}$ being the solution of Eq. (12). Moreover, consider SGDes (8) and (11) characterized by iteration number i and MO-actor learning rate $\{\mu_i\}_{i \in \mathcal{I}}$ with*

$$\mu_i \leq \min\left\{\frac{1}{L}, \frac{1}{L\|\mathbf{B}\|}\mathbb{E}\left\{\frac{1}{\mathbf{1}_r^{\top}\left(\nabla \boldsymbol{J}(\boldsymbol{\theta}^i)^{\top} \nabla \boldsymbol{J}(\boldsymbol{\theta}^i)\right)^{-1} \mathbf{1}_r}\right\}\right\},$$

which generate sequences $\{\boldsymbol{\phi}^i\}_{i \in \mathcal{I}}$ and $\{\boldsymbol{\theta}^i\}_{i \in \mathcal{I}}$, we have:

$$\mathbb{E}\, J_{\mathrm{ac,j}}(\boldsymbol{\theta}^{i+1}) \ \leq\ \mathbb{E}\, J_{\mathrm{ac,j}}(\boldsymbol{\theta}^i).$$

Proof. Please refer to Appendix A. □

Notice that Theorem IV.1 guarantees all the MO-actor expected losses $\left\{\mathbb{E}\, J_{\mathrm{ac,j}}(\boldsymbol{\theta}^i)\right\}_{j=1}^r$ continually reduce as the

algorithm iteration increases. It thus empowers us to simultaneously improve all of the cumulative rewards with each iteration, on average.

Theorem IV.2. *Consider the framework of Theorem IV.1, and assume MOO P_1 with a $\boldsymbol{\theta}$-parametric policy distribution $\pi_{\boldsymbol{\theta}}(\cdot|\cdot)$ being optimized by SGDes (8) and (11) with generated sequences $\{\boldsymbol{\phi}^i\}_{i\in\mathcal{I}}$ and $\{\boldsymbol{\theta}^i\}_{i\in\mathcal{I}}$ and MO-actor learning rates $\{\mu_i\}_{i\in\mathcal{I}}$ complying with assumptions of Theorem IV.1. Furthermore, assume there exists a Pareto optimal solution $\boldsymbol{\theta}^*$ of P_1 dominating $\boldsymbol{\theta}^i$ for the objectives $\{J_{\mathrm{ac,j}}(\cdot)\}_{j=1}^r$ with $i \in \mathcal{I}$, we then have:*

$$\mathbb{E}\|\boldsymbol{\theta}^{i+1} - \boldsymbol{\theta}^*\| \leq (1 - \Gamma\mu_i)\,\mathbb{E}\|\boldsymbol{\theta}^i - \boldsymbol{\theta}^*\| + (1+L)\mu_i^2\|\mathbf{B}\|. \tag{14}$$

Proof. Please refer to Appendix B. □

Accordingly, we can get:

Corollary IV.1. *Consider the SGD approach (11) with generated sequence $\{\boldsymbol{\theta}^n\}_{n\in\mathcal{I}}$ and MO-actor learning rate $\{\mu_n\}_{n\in\mathcal{I}}$ complying with assumptions of Theorem IV.1. Set μ_n so that $\lim_{n\to\infty}\mu_n = 0$, then $\lim_{n\to\infty}\mathbb{E}\|\boldsymbol{\theta}^n - \boldsymbol{\theta}^*\|^2 = 0$.*

Proof. We use the result of [16]. As such, based on Theorem IV.2 we have:

$$\begin{aligned}\Delta_{n+1} - \epsilon &\leq (1-\Gamma\mu_n)(\Delta_n - \epsilon) - \mu_n(\Gamma\epsilon - (1+L)\mu_n\|\mathbf{B}\|)\\ &\overset{a}{\leq} (1-\Gamma\mu_n)(\Delta_n - \epsilon).\end{aligned}$$

where $\Delta_n = \mathbb{E}\|\boldsymbol{\theta}^n - \boldsymbol{\theta}^*\|^2$ and $\epsilon > 0$. For (a), we considered that $\Gamma\epsilon - (1+L)\mu_n\|\mathbf{B}\| \geq \mathbf{0}$ for large n. Hence, for $\Gamma\mu_n \leq 1$ it reads:

$$[\Delta_{n+1} - \epsilon]^+ \leq (1-\Gamma\mu_n)[\Delta_n - \epsilon]^+,$$

where $[x]^+ = x + |x|$. By iterating, we get:

$$[\Delta_{n+k} - \epsilon]^+ \leq \prod_{i=0}^{k-1}(1-\Gamma\mu_{n+i})\,[\Delta_n - \epsilon]^+.$$

Considering that $\lim_{k\to\infty}\prod_{i=0}^{k-1}(1-\Gamma\mu_{n+i}) = 0$, we have: $\lim_{m\to\infty}[\Delta_m - \epsilon]^+$. Since it holds for any value $\epsilon > 0$, the statement follows. □

Note that the result of Corollary IV.1 guarantees that a convergence-in-mean can be achieved by choosing a suitable MO-actor learning rate.

In the sequels, we study how the learning rate can be chosen to achieve a convergence-in-mean. For the learning rate μ_i being sufficiently small, the evolution of SGD (11) can be regarded in a continuous time perspective with parameter t. As such, based on 14, the dynamics of quantity $\Delta_i = \mathbb{E}\|\boldsymbol{\theta}^i - \boldsymbol{\theta}^*\|$ can be expressed by the following differential inequality:

$$d\Delta_t \leq -\Gamma\mu_t\Delta_t dt + (1+L)\mu_t^2\|\mathbf{B}\|dt,$$

which gives the following solution:

$$\Delta_\tau \leq \Delta_0\, e^{-\Gamma\int_0^\tau \mu_s ds} + (1+L)\|\mathbf{B}\|\int_0^\tau \mu_t^2\, e^{-\Gamma\int_t^\tau \mu_s ds}dt.$$

Then it can be simply verified that for the selection $\mu_t = \frac{c_0}{t}$, we can get $\lim_{\tau\to\infty}\Delta_\tau \to 0$, which thus yields:

$$\lim_{i\to\infty}\mathbb{E}\|\boldsymbol{\theta}^i - \boldsymbol{\theta}^*\| \to 0.$$

This complies with the result of Corollary IV.1 and shows a strategy to choose a dynamics for $\{\mu_i\}_{i\in\mathcal{I}}$.

V. Experiments

A. Evaluation over a Multi-Task Problem

To empirically evaluate the devised *MO-A2C* algorithm, we consider a practical multi-task problem from the context of edge caching for cellular networks. For this, we take into account the system model presented in [17]. The environment of this task is a mobile cellular network which serves requesting mobile users by employing base-stations. The cellular network operates in a time-slotted fashion with time index $t \in \{1,\ldots,T\}$, where T is the total duration within which the network operation is considered.

The network itself constitutes file-requesting users, transmitting base-stations as well as a library containing N files from which the users request files. The users are spatially distributed across the network. They are interested in different files based on a *file popularity* which determines the probability that a file is preferred by a typical user. As such, files will be requested with different probabilities. The base-stations are also spatially distributed and are equipped with caches. The base-station caches are with a limited capacity of M files. They proactively store files from the library at their caches. For this, a probabilistic approach is used to place the files at base-station caches; file $n \in \{1,\ldots,N\}$ is cached at a typical base-station in time t with probability $p_{t,n}$.

To model the location of users and base-stations, we use two Poisson point processes with intensities λ_u and λ_b, respectively. The network employs the base-stations to multicast the cached files to satisfy users. Moreover, the base-stations exploit a resource allocation mechanism to transmit different files. As such, disjoint file-specific radio resources are allocated so that file n is transmitted at time t by occupying bandwidth $w_{t,n}$. The total amount of radio resources being needed to satisfy users is named **bandwidth consumption cost**.

Not all users can be successfully served by the network due to a reception outage probability. As such, at time t, a typical user preferring file n is not able to receive the needed file with a reception outage probability $\mathcal{O}_{t,n}$. The unsatisfied users will fetch the file directly from the network using a reactive transmission and consuming a sufficient amount of radio resources. However, this causes a network load, namely as **backhaul cost**, due to on-demand file transmission from the core-network to the requesting user.

The aim of this problem is to design a cache policy in order to satisfy as many users as possible with the minimum level of resource consumption and backhaul cost. More specifically, the cache policy should consider three competing objectives. (a) **quality-of-service (QoS) metric** that measures the percentage of users that can successfully receive their needed files. This metric shows the probability that a requesting user is properly satisfied by downloading its needed file, and it can be expressed at time t based on the file-specific reception outage probability $\mathcal{O}_{t,n}$. (b) **bandwidth**

(BW) consumption metric indicates the total radio resources being allocated to respond users. (c) **backhaul (BH) cost** measures the network load needed to fetch files directly from the network than the cache of base-stations. Note that we express QoS metric, BW consumption, and BH load objectives at time t based on negative immediate rewards that are denoted by $r_{\text{QoS}}(t)$, $r_{\text{BW}}(t)$ and $r_{\text{BH}}(t)$, respectively. Note that these immediate rewards depend on the file-specific cache probabilities $\{p_{t,n}\}_1^N$ and file-specific resource allocation $\{w_{t,n}\}_1^N$ [17]. As such, a multi-task problem with three competing objectives can be formulated.

For this problem, we define the system state $\boldsymbol{s}_t$ as a vector containing file-specific request probabilities of users from the network. We denote this vector by $r_{t,n}$ which shows the probability that file n is requested from the network by a typical user at time t. The system action $\boldsymbol{a}_t$ is a vector comprising of the file-specific cache probabilities $p_{t,n}$ and file-specific resource allocation $w_{t,n}$. Hence, the state and action vectors are expressed by:

$$\boldsymbol{s}_t = \big[\{r_{t,n}\}_1^N\big]^\top, \boldsymbol{a}_t = \big[\{p_{t,n}\}_1^N, \{w_{t,n}\}_1^N\big]^\top, t \in \{1, \ldots, T\}.$$

This cache policy problem can be formulated based on a MO-MDP with a continuous state-action space [17], and as such it can be designed based on a multi-objective RL algorithm. Therefore, we apply the algorithm 1 for this problem whose solution is denoted by *MO-A2C*. We also construct a scalar reward based on a linear combination and then use a single-objective A2C algorithm whose solution is denoted by *SO-A2C*.

B. Experiment Setup and Hyper-parameters

We consider the following settings for the considered system environment. The number of files is $N = 100$, the capacity of base-stations $M = 10$, and the spatial intensity of base-stations and users are $\lambda_b = 10$ and $\lambda_u = 10^5$, respectively, with the units of points/km^2. The desired rate of transmission is 1 Mbits/second. This quantity affects the reception outage probability. The total number of time-slots is T = 256 and the discount factor is set γ = 0.96.

For the *MO-A2C* algorithm, the actor and critic learning rates are set to 1×10^{-3}. Two separate neural networks, each with one hidden layer, represent the MO-actor and MO-critic agents. The MO-critic network outputs three values representing the reward-specific state-value functions $V_{\boldsymbol{\phi},j}(\cdot)$. The number of neurons in the hidden layer for the critic is 64 and the rectified linear unit (ReLU) activation function is used for its neuron. The MO-actor network represents the RL single-policy distribution $\pi_{\boldsymbol{\theta}}(\cdot,\cdot)$. The number of neurons in the hidden layer for the actor network is 128.

For the *SO-A2C* algorithm, the actor and critic learning rates are the same as *MO-A2C*. Moreover, the same architecture is considered for the actor and critic neural networks, except that the number of neurons in the hidden layer of the actor is 64, which is set to give the best performance. The critic network outputs only a single state-value function related to the scalar reward. Notice that the scalar reward is obtained based on a linear combination of the aforementioned rewards $r_{\text{Qos}}(t)$, $r_{\text{BW}}(t)$ and $r_{\text{BH}}(t)$, defined as follows:

$$r_{\text{sc}} = \lambda_{\text{QoS}}\, r_{\text{QoS}}(t) + \lambda_{\text{BW}}\, r_{\text{BW}}(t) + \lambda_{\text{BH}}\, r_{\text{BH}}(t),$$

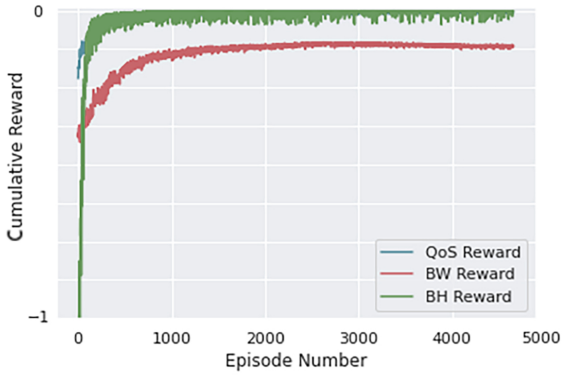

Fig. 2: Discounted cumulative reward for QoS metric, BW consumption and BH load obtained by *MO-A2C*.

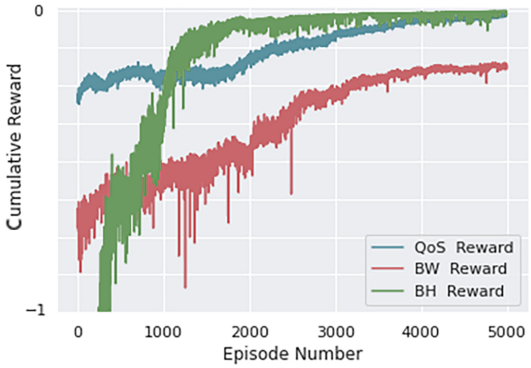

Fig. 3: Discounted cumulative reward for QoS metric, BW consumption and BH load obtained by *SO-A2C*.

where λ_{QoS}, λ_{BW} and λ_{BH} are the scalarization scales. We evaluate the following combinations for these scales:

$$[\lambda_{\text{QoS}},\, \lambda_{\text{BW}},\, \lambda_{\text{BH}}] \in \big\{[1,\, 1,\, 1],\, [0.1,\, 1,\, 1],\, [1,\, 10,\, 0.1]\big\}.$$

C. Experiment Results

We apply the multi-task problem explained in Section V-A. Additionally, it is noteworthy that the considered rewards, i.e., $r_{\text{Qos}}(t)$, $r_{\text{BW}}(t)$ and $r_{\text{BH}}(t)$, are re-scaled to lie in the range $[0,\, 1]$ and then are used by *MO-A2C* and *SO-A2C* algorithms.

We evaluate the sample efficiency of *MO-A2C* and compare it to *SO-A2C* with scalarization scales $[\lambda_{\text{QoS}},\, \lambda_{\text{BW}},\, \lambda_{\text{BH}}] = \mathbf{1}_3$. The training performance of *MO-A2C* and *SO-A2C* are plotted in Figures 2 and 3, in terms of the cumulative rewards of mentioned metrics (QoS metric, BW consumption, and BH load) for different episodes ($E_{\max}$). According to Figures 2 and 3, it can be seen that *MO-A2C* outperforms *SO-A2C* from sample-efficiency perspective, as it can be learned after 2×10^3 episodes while *SO-A2C* needs more that 5×10^3 episode samples to be learned.

To evaluate the scale-dependency of *MO-A2C* and *SO-A2C*, we consider a test scenario. For *SO-A2C*, we use the scalarization scales $[\lambda_{\text{QoS}}, \lambda_{\text{BW}}, \lambda_{\text{BH}}] = [0.1, 1, 1]$. For *MO-A2C*, we multiply the QoS reward by 0.1 and keep the other two rewards unchanged, and then evaluate the algorithms over the re-scaled rewards. For this scenario, we plot the training performance of *MO-A2C* and *SO-A2C* in Figures

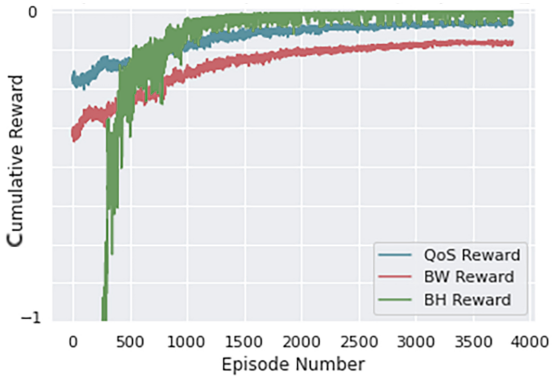

Fig. 4: Discounted cumulative reward for QoS metric, BW consumption and BH load obtained by *MO-A2C*. QoS reward is multiplied by 0.1 and other rewards are kept unchanged.

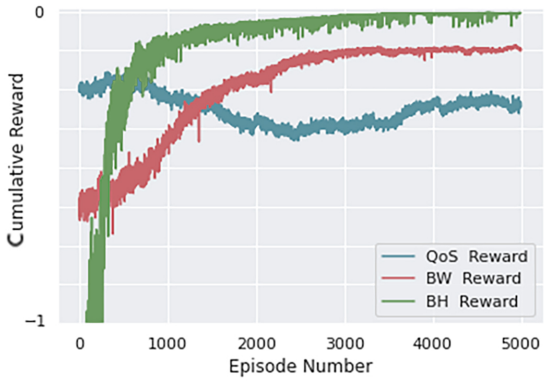

Fig. 5: Discounted cumulative reward for QoS metric, BW consumption and BH load obtained by *SO-A2C*. To constitute a scalarized reward, we use the scales $[\lambda_{\text{QoS}},\ \lambda_{\text{BW}},\ \lambda_{\text{BH}}] = [0.1,\ 1,\ 1]$.

4 and 5, in terms of the cumulative rewards for different episodes. Based on Figures 4 and 5, it can be inferred that *MO-A2C* is more insensitive towards the scaling than *SO-A2C*. More specifically, *MO-A2C* is able to learn the RL agent after around 3×10^3 episodes despite of QoS reward being re-scaled. However, *SO-A2C* agent is not properly learned even after 5×10^3 episodes.

We also consider another test scenario for scale-invariance evaluation of *MO-A2C*. For *SO-A2C*, the scales $[\lambda_{\text{QoS}},\ \lambda_{\text{BW}},\ \lambda_{\text{BH}}] = [1,\ 10,\ 0.1]$ are used and for *MO-A2C*, we multiply the rewards related to BW consumption and BH load by 10 and 0.1, respectively, and keep the QoS reward unchanged. We then evaluate *MO-A2C* over the re-scaled rewards. The training performance of *MO-A2C* and *SO-A2C* are sketched in Figures 6 and 7, in terms of the cumulative rewards for different episodes. Comparison of Figures 6 and 7 confirms that *MO-A2C* is more robust to re-scaling factors than *SO-A2C* and that *MO-A2C* can be considered as a scale-invariance approach.

VI. Conclusion

In this paper, we devised a scale-independent multi-objective reinforcement learning approach on the grounds of the advantage actor-critic (A2C) algorithm. By making some assumptions, we then provided a convergence analysis based on which a convergence-in-mean can be guaranteed. We compared our algorithm for a multi-task problem against

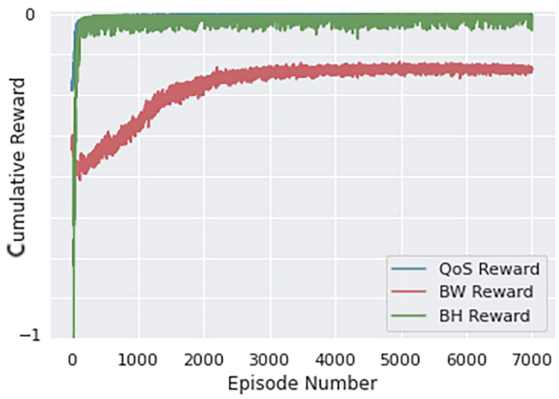

Fig. 6: Discounted cumulative reward for QoS metric, BW consumption and BH load obtained by *MO-A2C*. Rewards related to BW consumption and BH load are multiplied by 10 and 0.1, respectively.

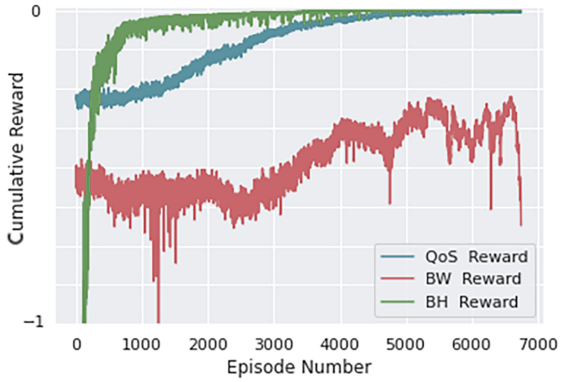

Fig. 7: Discounted cumulative reward for QoS metric, BW consumption and BH load obtained by *SO-A2C*. To constitute a scalarized reward, we use the scales $[\lambda_{\text{QoS}},\ \lambda_{\text{BW}},\ \lambda_{\text{BH}}] = [1,\ 10,\ 0.1]$.

a single-objective A2C with a scalarized reward. The simulation results show the capability of the developed algorithm from the sample-efficiency, optimality and scale-invariance perspectives.

VII. Acknowledgement

This work was partially funded by the Research Council of Finland under grant number 357533. We thank Prof. Olav Tirkkonen and Ashvin Srinivasan for the discussion about simulations.

References

[1] G. Kirlik and S. Sayın, "A new algorithm for generating all non-dominated solutions of multiobjective discrete optimization problems," *European Journal of Operational Research*, vol. 232, no. 3, pp. 479–488, 2014.

[2] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley, "A survey of multi-objective sequential decision-making," *Journal of Artificial Intelligence Research*, vol. 48, no. 1, p. 67–113, oct 2013.

[3] K. Van Moffaert, M. M. Drugan, and A. Nowé, "Scalarized multi-objective reinforcement learning: Novel design techniques," in *IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, 2013, pp. 191–199.

[4] S. Natarajan and P. Tadepalli, "Dynamic preferences in multi-criteria reinforcement learning," in *International Conference on Machine Learning*, 2005, p. 601–608.

[5] H. Iima and Y. Kuroe, "Multi-objective reinforcement learning for acquiring all pareto optimal policies simultaneously - method of determining scalarization weights," in *IEEE International Conference on Systems, Man, and Cybernetics*, 2014, pp. 876–881.

[6] H. Mossalam, Y. M. Assael, D. M. Roijers, and S. Whiteson, "Multi-objective deep reinforcement learning," preprint ArXiv 1610.02707, 2016.
[7] R. Yang, X. Sun, and K. Narasimhan, "A generalized algorithm for multi-objective reinforcement learning and policy adaptation," in *International Conference on Neural Information Processing Systems*, 2019.
[8] X. Chen, A. Ghadirzadeh, M. Björkman, and P. Jensfelt, "Meta-learning for multi-objective reinforcement learning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019, pp. 977–983.
[9] H. Zhan and Y. Cao, "Relationship explainable multi-objective optimization via vector value function based reinforcement learning," preprint ArXiv 1910.01919, 2019.
[10] A. Abdolmaleki, S. H. Huang, L. Hasenclever, M. Neunert, H. F. Song, M. Zambelli, M. F. Martins, N. Heess, R. Hadsell, and M. Riedmiller, "A distributional view on multi-objective policy optimization," *Proc. International Conference on Machine Learnin*, Jul 2020.
[11] A. Abdolmaleki, J. T. Springenberg, Y. Tassa, R. Munos, N. Heess, and M. Riedmiller, "Maximum a posteriori policy optimisation," in *International Conference on Learning Representations*, 2018.
[12] I. Grondman, L. Busoniu, G. A. D. Lopes, and R. Babuska, "A survey of actor-critic reinforcement learning: Standard and natural policy gradients," *IEEE Trans. on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 1291–1307, 2012.
[13] S. Schäffler, S. R. Schultz, and K. Weinzierl, "Stochastic method for the solution of unconstrained vector optimization problems," 2002.
[14] P. Ma, T. Du, and W. Matusik, "Efficient continuous pareto exploration in multi-task learning," in *International Conference on Machine Learning*, ser. ICML'20, 2020.
[15] S. Qiu, Z. Yang, J. Ye, and Z. Wang, "On finite-time convergence of actor-critic algorithm," *IEEE Journal on Selected Areas in Information Theory*, vol. 2, no. 2, pp. 652–664, 2021.
[16] G. Turinici, "The convergence of the stochastic gradient descent (SGD): a self-contained proof," Tech. Rep., 2021.
[17] M. Amidzadeh, H. Al-Tous, O. Tirkkonen, and J. Zhang, "Joint cache placement and delivery design using reinforcement learning for cellular networks," in *IEEE Vehicular Technology Conference*, 2021, pp. 1–6.

APPENDIX

A. Proof of Theorem IV.1

Proof. According to the update rule Eq. (11), we have:

$$\boldsymbol{\theta}^{i+1} = \boldsymbol{\theta}^i - \mu_i \nabla \hat{\boldsymbol{J}}(\boldsymbol{\theta}^i, \boldsymbol{\phi}^i)\boldsymbol{\alpha}_{\mathrm{ac}}^i,$$

where $\nabla \hat{\boldsymbol{J}}(\boldsymbol{\theta}^i, \boldsymbol{\phi}^i) = [\nabla_{\boldsymbol{\theta}} \hat{J}_{\mathrm{ac},1}, \ldots, \nabla_{\boldsymbol{\theta}} \hat{J}_{\mathrm{ac,r}}](\boldsymbol{\theta}^i, \boldsymbol{\phi}^i)$, based on which and Assumption 2, we thus obtain:

$$\begin{aligned} J_{\mathrm{ac,j}}(\boldsymbol{\theta}^{i+1}) - J_{\mathrm{ac,j}}(\boldsymbol{\theta}^i) \leq & -\mu_i \nabla J_{\mathrm{ac,j}}(\boldsymbol{\theta}^i)^\top \nabla \hat{\boldsymbol{J}}(\boldsymbol{\theta}^i, \boldsymbol{\phi}^i)\boldsymbol{\alpha}_{\mathrm{ac}}^i \\ & + \frac{\mu_i^2 L}{2} {\boldsymbol{\alpha}_{\mathrm{ac}}^i}^\top \nabla \hat{\boldsymbol{J}}(\boldsymbol{\theta}^i, \boldsymbol{\phi}^i)^\top \nabla \hat{\boldsymbol{J}}(\boldsymbol{\theta}^i, \boldsymbol{\phi}^i)\boldsymbol{\alpha}_{\mathrm{ac}}^i \end{aligned}$$

It then reads:

$$\begin{aligned} & \mathbb{E}\left\{J_{\mathrm{ac,j}}(\boldsymbol{\theta}^{i+1}) - J_{\mathrm{ac,j}}(\boldsymbol{\theta}^i)\right\} \\ & \leq \mathbb{E}\Big\{\mathbb{E}\Big\{\Big(\frac{\mu_i^2 L}{2}\nabla \hat{\boldsymbol{J}}(\boldsymbol{\theta}^i, \boldsymbol{\phi}^i)\boldsymbol{\alpha}_{\mathrm{ac}}^i \\ & \qquad - \mu_i \nabla \boldsymbol{J}(\boldsymbol{\theta}^i)\boldsymbol{e}_j\Big)^\top \nabla \hat{\boldsymbol{J}}(\boldsymbol{\theta}^i, \boldsymbol{\phi}^i)\boldsymbol{\alpha}_{\mathrm{ac}}^i \,\Big|\, \boldsymbol{\theta}^i, \boldsymbol{\phi}^i\Big\}\Big\} \\ & \overset{a}{\leq} -\mu_i \mathbb{E}\left\{\Big(\boldsymbol{e}_j - \frac{\mu_i L}{2}\boldsymbol{\alpha}_{\mathrm{ac}}^i\Big)^\top \nabla \boldsymbol{J}(\boldsymbol{\theta}^i)^\top \nabla \boldsymbol{J}(\boldsymbol{\theta}^i)\boldsymbol{\alpha}_{\mathrm{ac}}^i\right\} + \frac{\mu_i^2 L}{2}\|\mathbf{B}\|, \end{aligned} \tag{15}$$

where (a) was obtained based on

$$\begin{aligned} & \mathbb{E}_{\boldsymbol{\phi}^i}\, \mathbb{E}\left\{{\boldsymbol{\alpha}_{\mathrm{ac}}^i}^\top \nabla \boldsymbol{J}(\boldsymbol{\theta}^i)^\top \nabla \hat{\boldsymbol{J}}(\boldsymbol{\theta}^i, \boldsymbol{\phi}^i)\boldsymbol{\alpha}_{\mathrm{ac}}^i \,\middle|\, \boldsymbol{\theta}^i, \boldsymbol{\phi}^i\right\} \\ & \qquad = \quad {\boldsymbol{\alpha}_{\mathrm{ac}}^i}^\top \nabla \boldsymbol{J}(\boldsymbol{\theta}^i)^\top \nabla \boldsymbol{J}(\boldsymbol{\theta}^i)\boldsymbol{\alpha}_{\mathrm{ac}}^i, \end{aligned}$$

Assumption 3 and ${\boldsymbol{\alpha}_{\mathrm{ac}}^i}^\top \mathbf{B}\boldsymbol{\alpha}_{\mathrm{ac}}^i \leq \|\mathbf{B}\|\, \|\boldsymbol{\alpha}_{\mathrm{ac}}^i\|^{\mathbf{2}} \leq \|\mathbf{B}\|$. On the other hand, from Eq. (12), for all $\alpha_{\mathrm{ac,j}}^i \geq 0$, it reads:

$$\boldsymbol{\alpha}_{\mathrm{ac}}^i = \left[\mathbf{1}_r^\top \left(\nabla \boldsymbol{J}(\boldsymbol{\theta}^i)^\top \nabla \boldsymbol{J}(\boldsymbol{\theta}^i)\right)^{-1} \mathbf{1}_r\right]^{-1} \left(\nabla \boldsymbol{J}(\boldsymbol{\theta}^i)^\top \nabla \boldsymbol{J}(\boldsymbol{\theta}^i)\right)^{-1} \mathbf{1}_r.$$

By substituting this into Eq. (15), we get:

$$\begin{aligned} & \mathbb{E}\Big\{J_{\mathrm{ac,j}}(\boldsymbol{\theta}^{i+1}) - J_{\mathrm{ac,j}}(\boldsymbol{\theta}^i)\Big\} \\ & \leq -\Big(\mu_i - \frac{\mu_i^2 L}{2}\Big)\mathbb{E}\Big\{\frac{1}{\mathbf{1}_r^\top \left(\nabla \boldsymbol{J}(\boldsymbol{\theta}^i)^\top \nabla \boldsymbol{J}(\boldsymbol{\theta}^i)\right)^{-1} \mathbf{1}_r}\Big\} + \frac{\mu_i^2 L}{2}\|\mathbf{B}\| \\ & \overset{a}{\leq} -\frac{\mu_i}{2}\mathbb{E}\Big\{\frac{1}{\mathbf{1}_r^\top \left(\nabla \boldsymbol{J}(\boldsymbol{\theta}^i)^\top \nabla \boldsymbol{J}(\boldsymbol{\theta}^i)\right)^{-1} \mathbf{1}_r}\Big\} + \frac{\mu_i^2 L}{2}\|\mathbf{B}\| \;\leq\; \mathbf{0}, \end{aligned}$$

where we used $\mu_i L \leq 1$ for (a). Considering that the denominator of RHS of the recent equation is positive due to the positive-definiteness of $\left(\nabla \boldsymbol{J}(\boldsymbol{\theta}^i)^\top \nabla \boldsymbol{J}(\boldsymbol{\theta}^i)\right)^{-1}$, the statement follows for $\mu_i < \frac{2}{L}$. □

B. Proof of Theorem IV.2

We need the following Corollary to prove Theorem IV.2.

Corollary A.1. *Consider the framework of Lemma IV.1, we then get:*

$$\begin{aligned} & \mathbb{E}\left\{{\boldsymbol{\alpha}_{\mathrm{ac}}^i}^\top \nabla \boldsymbol{J}(\boldsymbol{\theta}^i)^\top \nabla \boldsymbol{J}(\boldsymbol{\theta}^i)\boldsymbol{\alpha}_{\mathrm{ac}}^i\right\} \\ & \qquad \leq \frac{2}{\mu_i}\mathbb{E}\left\{\sum_{j=1}^r \alpha_j^i \big(J_{\mathrm{ac,j}}(\boldsymbol{\theta}^i) - J_{\mathrm{ac,j}}(\boldsymbol{\theta}^{i+1})\big)\right\} + \mu_i L\|\mathbf{B}\|. \end{aligned}$$

Proof. Based on Eq. (15) and $\mu_i L \leq 1$, the statement follows. □

We now prove Theorem IV.2.

Proof. Based on SGD update (11), we obtain:

$$\begin{aligned} & \mathbb{E}\|\boldsymbol{\theta}^{i+1} - \boldsymbol{\theta}^*\|^2 = \mathbb{E}\|\boldsymbol{\theta}^i - \boldsymbol{\theta}^* - \mu_i \nabla \hat{\boldsymbol{J}}(\boldsymbol{\theta}^i, \boldsymbol{\phi}^i)\boldsymbol{\alpha}_{\mathrm{ac}}^i\|^2 \\ & \leq \mathbb{E}\|\boldsymbol{\theta}^i - \boldsymbol{\theta}^*\|^2 \\ & \quad - 2\mu_i \mathbb{E}\Big\{\mathbb{E}\Big\{\sum_{j=1}^r \alpha_{\mathrm{ac,j}}^i \nabla \hat{J}_{\mathrm{ac,j}}(\boldsymbol{\theta}^i, \boldsymbol{\phi}^i)^\top (\boldsymbol{\theta}^i - \boldsymbol{\theta}^*) \,\Big|\, \boldsymbol{\theta}^i, \boldsymbol{\phi}^i\Big\}\Big\} \\ & \quad + \mathbb{E}\left\{\mu_i^2 {\boldsymbol{\alpha}_{\mathrm{ac}}^i}^\top \nabla \hat{\boldsymbol{J}}(\boldsymbol{\theta}^i, \boldsymbol{\phi}^i)^\top \nabla \hat{\boldsymbol{J}}(\boldsymbol{\theta}^i, \boldsymbol{\phi}^i)\boldsymbol{\alpha}_{\mathrm{ac}}^i\right\} \\ & \overset{a}{\leq} \mathbb{E}\|\boldsymbol{\theta}^i - \boldsymbol{\theta}^*\|^2 - 2\mu_i \mathbb{E}_{\boldsymbol{\theta}^i}\left\{\sum_{j=1}^r \alpha_{\mathrm{ac,j}}^i \nabla J_{\mathrm{ac,j}}(\boldsymbol{\theta}^i)^\top (\boldsymbol{\theta}^i - \boldsymbol{\theta}^*)\right\} \\ & \quad + \mathbb{E}\left\{\mu_i^2 {\boldsymbol{\alpha}_{\mathrm{ac}}^i}^\top \nabla \hat{\boldsymbol{J}}(\boldsymbol{\theta}^i, \boldsymbol{\phi}^i)^\top \nabla \hat{\boldsymbol{J}}(\boldsymbol{\theta}^i, \boldsymbol{\phi}^i)\boldsymbol{\alpha}_{\mathrm{ac}}^i\right\} \\ & \overset{b}{\leq} (1 - \Gamma\mu_i)\, \mathbb{E}\|\boldsymbol{\theta}^i - \boldsymbol{\theta}^*\|^2 \\ & \quad + 2\mu_i \mathbb{E}\left\{\sum_{j=1}^r \alpha_{\mathrm{ac,j}}^i \big(J_{\mathrm{ac,j}}(\boldsymbol{\theta}^*) - J_{\mathrm{ac,j}}(\boldsymbol{\theta}^i)\big)\right\} \\ & \quad + \mu_i^2 \mathbb{E}\left\{\mathbb{E}\Big\{{\boldsymbol{\alpha}_{\mathrm{ac}}^i}^\top \nabla \hat{\boldsymbol{J}}^{i\top} \nabla \hat{\boldsymbol{J}}^i \boldsymbol{\alpha}_{\mathrm{ac}}^i \,\Big|\, \boldsymbol{\theta}^i, \boldsymbol{\phi}^i\Big\}\right\} \\ & \overset{c}{\leq} (1 - \Gamma\mu_i)\, \mathbb{E}\|\boldsymbol{\theta}^i - \boldsymbol{\theta}^*\|^2 + \mu_i^2\|\mathbf{B}\| \\ & \quad + 2\mu_i \mathbb{E}\left\{\sum_{j=1}^r \alpha_{\mathrm{ac,j}}^i \Big(J_{\mathrm{ac,j}}(\boldsymbol{\theta}^*) - J_{\mathrm{ac,j}}(\boldsymbol{\theta}^{i+1})\Big)\right\} \\ & \overset{d}{\leq} (1 - \Gamma\mu_i)\, \mathbb{E}\|\boldsymbol{\theta}^i - \boldsymbol{\theta}^*\|^2 + (1 + L)\mu_i^2\|\mathbf{B}\|, \end{aligned}$$

where (a) was achieved considering $\mathbb{E}_{\boldsymbol{\phi}}\mathbb{E}\{\nabla \hat{J}_{\mathrm{ac,j}}(\boldsymbol{\theta}, \phi)|\boldsymbol{\theta}, \phi\} = \nabla J_{\mathrm{ac,j}}(\boldsymbol{\theta})$ based on Assumption 1, (b) was obtained based on Assumption 2, (c) according to Assumption 3 and Corollary A.1, and for (d) we exploited $\boldsymbol{\theta}^*$ being a dominating Pareto optimum. □