3.1 Design of the Governor RL Agent

We model the problem of searching for the best values of T_l and T_a as executing an optimal policy over a Markov Decision Process (MDP) whose states are defined over T_l and T_a and whose reward function captures the IoT system performance. By executing such an optimal policy, the RL-agent is guaranteed to search over the space of T_l and T_a values to maximize the system's performance. In the remainder of this subsection, we discuss how to define the MDP's main components, namely: state space, action space, and reward function.

Governor MDP State space $\mathcal{X}_\mathcal{G}$: The MDP states are the different values that T_l and T_a can take. That is, each state $s_g \in \mathcal{X}_\mathcal{G}$ is determined by a tuple (T_l, T_a). Since RL-agents' performance depends heavily on the cardinality of the state space, we discretize the continuous values of T_l and T_a into a finite number of values where such discretization depends on the context of the application as it will be discussed in Sections 5 and 6.

Governor MDP Action space $\mathcal{A}_\mathcal{G}$: The action space $\mathcal{A}_\mathcal{G}$ in this MDP contains all the possible combinations of changing the T_l and T_a values. Particularly, the action can be either *decrement* with some value $\downarrow$, *increment* with some value $\uparrow$, or *remain* the same value $\circlearrowleft$. Such actions are defined for both T_l and T_a under the constraint that $T_l \geq T_a$. We designed the actions as an increment or a decrement rather than choosing a specific tuple (T_l, T_a) from all the possible combinations to make sure that there is no big sudden change in the actuator rate (T_a) which may compromise the human's experience.

Governor MDP Reward Function $R_\mathcal{G}$: Each state $s_g \in \mathcal{S}_\mathcal{G}$ is associated with a performance p_s. Such performance p_s is a measure of the human's experience and the performance of the IoT system, which depends on the context of the application, as it will be discussed in Section 4. The performance is calculated after running the Multisample Q-learning [14] using the values of T_l and T_a associated with state s_g. The associated performance p_s plays a role in determining the reward r_g value that accrued due to taking the action a_g at the state s_g. In particular, the reward value $r_g = R_\mathcal{G}(s_g, a_g)$ (positively or negatively) depends on a weighted difference between the performance of the system p_s at the state s_g (before modifying the values of T_l and T_a) and the performance of the system $p_{s'}$ at the new state s'_g (after modifying the values of T_l and T_a).

In this MDP setting, the transition probabilities in are known apriori since *increasing/decreasing/remain the same* T_l or T_a leads to a known state. However, the reward is unknown apriori because it depends on the human's experience. Moreover, it can change over time. Hence, to solve the MDP when the reward values are unknown, we use Q-learning which is summarized in Algorithm 1.

Algorithm 1 Governor RL

Hyper parameters: Learning parameters: α, γ, ϵ
Require:
Sates $\mathcal{X}_\mathcal{G} = \{(T_l, T_a)_1, \ldots, (T_l, T_a)_x\}$
Actions $\mathcal{A}_\mathcal{G} = \{(\circlearrowleft T_l, \circlearrowleft T_a), (\circlearrowleft T_l, \uparrow T_a), (\uparrow T_l, \circlearrowleft T_a), (\uparrow T_l, \uparrow T_a), (\circlearrowleft T_l, \downarrow T_a), (\downarrow T_l, \circlearrowleft T_a), (\downarrow T_l, \downarrow T_a), (\downarrow T_l, \uparrow T_a), (\uparrow T_l, \downarrow T_a)\}$,
Reward function $R_\mathcal{G} : \mathcal{X}_\mathcal{G} \times \mathcal{A}_\mathcal{G} \rightarrow \mathbb{R}$
Transition function $T_\mathcal{G} : \mathcal{X}_\mathcal{G} \times \mathcal{A}_\mathcal{G} \rightarrow \mathcal{X}_\mathcal{G}$
Multisample Q-learning algorithm: $MuQL(T_l, T_a)$
Learning rate $\alpha \in [0, 1]$, $\alpha = 0.9$
Discounting factor $\gamma \in [0, 1]$, $\gamma = 0.1$
ϵ-Greedy exploration strategy $\epsilon \in [0, 1]$, $\epsilon = 0.2$
Weighted Performance Difference $\mathcal{W}$
procedure $GovQL(\mathcal{X}_\mathcal{G}, \mathcal{A}_\mathcal{G}, R_\mathcal{G}, T_\mathcal{G}, \alpha, \gamma)$
  Initialize $Q_\mathcal{G} : \mathcal{X}_\mathcal{G} \times \mathcal{A}_\mathcal{G} \rightarrow \mathbb{R}$ with 0
  while true **do**
    Start in state $s_g \in \mathcal{X}_\mathcal{G}$
    $p_s \leftarrow MuQL(s_g)$ ▷ Calculate the performance of s_g
    Calculate $\pi(s_g)$ according to $Q_\mathcal{G}$ and exploration strategy:
      with probability ϵ: $\pi(s_g) \leftarrow$ choose an action a at random,
      with probability $1 - \epsilon$: $\pi(s_g) \leftarrow \text{argmax}_a\, Q_\mathcal{G}(s_g, a))$
    $a_g \leftarrow \pi(s_g)$
    $s'_g \leftarrow T_\mathcal{G}(s_g, a_g)$ ▷ Receive the new state
    $p_{s'} \leftarrow MuQL(s'_g)$ ▷ Calculate the performance of s'_g
    $r_g \leftarrow R_\mathcal{G}(s_g, a_g) = \mathcal{W}(p_{s'}, p_s)$ ▷ Receive the reward
    $Q_\mathcal{G}(s'_g, a_g) \leftarrow (1-\alpha) \cdot Q_\mathcal{G}(s_g, a_g) + \alpha \cdot (r_g + \gamma \cdot \max_{a'} Q_\mathcal{G}(s'_g, a'))$
    $s_g \leftarrow s'_g$
  return $Q_\mathcal{G}$

3.2 Design of the Mediator RL Agent

Similarly to the Governer RL agent, we model the problem of searching for the best weights (w_1, w_2, ... w_n) for the adaptation actions collected for individual personalized perfromance (a_1, a_2, ..., a_n) (from the Multisample RL agents adapting to multi-humans) as executing an optimal policy over a Markov Decision Process (MDP).

Mediator MDP State space $\mathcal{X}_\mathcal{M}$: The MDP states are the different values of w_i that can be summed up to 1. We discretize their continuous values into finite number of state space $\mathcal{X}_\mathcal{M}$, where their values are chosen from a predefined set of values $\{0, 0.2, 0.4, 0.6, 0.8, 1\}$. The state space's size depends on the number of actions that we need to take their weighted average, which reflects the number of humans in the environment. For example, if we have three adaptation actions (a_1, a_2, and a_3), then the state s_m will be a tuple of three values (w_1, w_2, and w_3). Hence, the size of the state space is 21 states.

Mediator MDP Action space $\mathcal{A}_\mathcal{M}$: The action space $\mathcal{A}_\mathcal{M}$ of the mediator MDP contains all the possible jumps $\nearrow$ to all the states. Hence, each state can have 21 actions, all with the same probability. Such a design choice allows the Mediator RL-agent to rapidly switch between weights and converge faster to the optimal assignment of weights.

Mediator MDP Reward function $R_\mathcal{M}$: Each state s_m has a performance p_s that is associated with it. The performance p is a measure of all the humans' cumulative experience and the performance of the IoT system, which depends on the context of the application, as will be discussed in Section 6. The performance is calculated after applying the weighted average action ($a_t = \frac{\sum_{i=1}^{n} w_i a_i}{\sum_{i=1}^{n} w_i}$) at a rate of the minimum T_a across all individual humans (as chosen by their individual Governor RL agents). Each human will have a different experience with the applied action a_t. Hence, the cumulative performance p_s associated with this state s_m will be a function of