Learn to Stay Cool: Online Load Management for Passively Cooled Base Stations

Yu, Zhanwei; Zhao, Yi; You, Lei; Yuan, Di

Published in:
Proceedings of IEEE Wireless Communications and Networking Conference

Publication date:
2024

Document Version
Peer reviewed version

Learn to Stay Cool: Online Load Management for Passively Cooled Base Stations

Zhanwei Yu[1], Yi Zhao[1], Lei You[2], and Di Yuan[1]

[1]Department of Information Technology, Uppsala University, Sweden
[2]Department of Engineering Technology, Technical University of Denmark, Denmark
[1]{*zhanwei.yu; yi.zhao; di.yuan*}*@it.uu.se,* [2]*leiyo@dtu.dk*

***Abstract*—Passively cooled base stations (PCBSs) are highly relevant for achieving better efficiency in cost and energy. However, dealing with the thermal issue via load management, particularly for outdoor deployment of PCBS, becomes crucial. This is a challenge because the heat dissipation efficiency is subject to (uncertain) fluctuation over time. Moreover, load management is an online decision-making problem by its nature. In this paper, we demonstrate that a reinforcement learning (RL) approach, specifically Soft Actor-Critic (SAC), enables to make a PCBS stay cool. The proposed approach has the capability of adapting the PCBS load to the time-varying heat dissipation. In addition, we propose a denial and reward mechanism to mitigate the risk of overheating from the exploration such that the proposed RL approach can be implemented directly in a practical environment, i.e., online RL. Numerical results demonstrate that the learning approach can achieve as much as 88.6% of the global optimum. This is impressive, as our approach is used in an online fashion to perform decision-making without the knowledge of future heat dissipation efficiency, whereas the global optimum is computed assuming the presence of oracle that fully eliminates uncertainty. This paper pioneers the approach to the online PCBSs load management problem.**

***Index Terms*—Passive cooling, load management, deep reinforcement learning.**

I. Introduction

Striving for simpler, cheaper, and more energy-efficient communication solutions. Passively cooled base stations (PCBSs) are gaining attention [1]. A PCBS typically has three main parts: a baseband unit (BBU), an active antenna unit (AAU), and a power module. They can be used to improve the coverage and throughput of the wireless mobile network. However, BBU thermal management poses a big challenge. Thermal management is not only closely tied to the load, notably the throughput, but also significantly affects the lifespan of the electronic components of PCBSs.

In traditional base stations, BBU modules are maintained at optimal temperatures using active cooling systems in machine cabinets or rooms with air conditioners, such as air-conditioned rooms. In contrast, PCBSs are commonly situated outdoors without the benefit of active cooling mechanisms. However, accurately modeling and predicting heat dissipation for outdoor passive cooling is complex due to the multitude of influencing factors, such as ambient temperature, wind speed, humidity, and solar exposure. With such uncertainties, the issue is best approached using online decision making.

In this paper, we utilize a reinforcement learning (RL) approach, specifically Soft Actor-Critic (SAC) [2], to tackle the challenge. Online problems require adjustments based on real-time data, making RL a promising choice for these dynamic situations. It facilitates fast decision-making and continuous adaptation. Moreover, we target learning from a practical environment instead of synthetic or pre-collected data, namely online RL. A notable risk is that the temperature might spike when the agent (i.e., the PCBS) explores the environment. This will shorten the lifespan of its electronic components. To counter this, we design an elaborate denial and reward mechanism, ensuring that the PCBS can explore while safeguarding against overheating. Our numerical results are promising as our learning approach reaches up to 88.6% of the global optimum. Note that our method operates online, making decisions without foreknowledge of future heat dissipation efficiency. In contrast, the global optimum assumes the existence of an oracle that completely removes uncertainty.

II. Literature Review

The works in [3]–[5] have studied the passive cooling techniques for base stations. The research in [3] conducted thermal simulations to evaluate the cooling performance of distinct CPU heatsinks. In [4], a novel heat dissipation structure was designed with antennas and digital beamforming chips on the opposite sides. The simulation demonstrated superior cooling efficiency compared with the traditional designs. Further, [5] explored various determinants for efficient PCBS cooling, encompassing environmental factors, strategic placement, and innovative material use.

There are some research works on temperature management in data centers via resource allocation. The study in [6], for instance, presented an optimal solution for dynamic computing resource allocation, considering two models of a computing unit in the BBU. The research in [7] introduced a computation model for resource efficiency and proposed alterations to the BBU's operational parameters to minimize power consumption. Furthermore, [8] presented a comprehensive approach to simultaneously regulate cooling and computational processes, aiming for overall energy conservation in data centers.

In addition, both [9] and [10] study load management using RL. The study in [10] introduces a deep RL framework for mobility load management in cellular networks, utilizing cell

individual offset adjustments. Depending on network size, it employs enhanced deep Q-learning or advanced RL methods like Deep Deterministic Policy Gradient and twin delayed deep deterministic policy gradient. In [9], the authors introduce a multi-objective RL framework for load balancing for multi-band downlink cellular networks. Utilizing meta-RL and policy distillation techniques, the framework is designed to adapt to diverse objective trade-offs quickly. Different from our paper, the works in [9] and [10] do not consider the passively cooling systems. To the best of our knowledge, no existing study has addressed load management for PCBS.

III. System Model and Problem Formulation

Consider a PCBS operating over a time-slotted horizon, represented by $\mathcal{I} = \{1, 2, .., I\}$. Let t_i denote the BBU chip temperature at time slot i, with the assumption that this temperature of updated at the outset of each slot. For convenience, the term "temperature" refers to the BBU chip temperature, distinguishing it from the "ambient temperature". The ambient temperature for time slot i is represented as t_i^{amb} (unit: °C). We use δ to represent the length of a time slot. While there is no specific constraint on the magnitude of δ, for PCBS thermal management, a time span on the order of seconds is reasonable for δ (unit: second).

The dynamics of temperature are influenced by heat production and its dissipation. Let $p_i^{\uparrow}$ and $p_i^{\downarrow}$ (unit: Watt) respectively represent heat generation and dissipation for time slot i. According to the principles of heat transfer [11], the relationship between them is given by

$$t_{i+1} = t_i + \lambda\delta(p_i^{\uparrow} - p_i^{\downarrow}), \tag{1}$$

where $\lambda \geq 0$ (unit: °C/Joule) defines the reciprocal of the thermal capacitance of the BBU. Heat generation $p_i^{\uparrow}$ correlates to the power consumption of the BBU's chip. This consumption has two parts, i.e., dynamic and static components, represented by p_i^{d} and p_i^{s} (unit: Watt) respectively [12], i.e.,

$$p_i^{\uparrow} = p_i^{\text{d}} + p_i^{\text{s}}. \tag{2}$$

The dynamic power consumption is determined by the amount of computations, and this is in direct proportion to the data traffic served by the BBU per unit time [13], i.e., load. Represented by x_i (unit: Mbps) for the load in time slot i, the relationship is given by

$$p_i^{\text{d}} = \mu x_i, \tag{3}$$

where μ (unit: Watt/Mbps) denotes a BBU-specific coefficient. The static power consumption, p_i^{s}, is observed to exhibit exponential growth relative to the temperature t_i [12], [14]. This can be represented as

$$p_i^{\text{s}} = f(t_i) = a\text{e}^{bt_i} + c, \tag{4}$$

where a, b, and c being parameters related to the chip's material properties. Define $\sigma_i > 0$ as the heat dissipation efficiency in the time slot i. Heat dissipation is a result of both σ_i and the temperature differential between the chip and its ambient surroundings. In accordance with Newton's law of cooling [11], this relationship is expressed as

$$p_i^{\downarrow} = \sigma_i(t_i - t_i^{\text{amb}}). \tag{5}$$

A weighted throughput optimization problem in PCBS can be formulated as

$$\max_{\boldsymbol{x},\boldsymbol{t}} \sum_{i\in\mathcal{I}} w_i x_i \tag{6a}$$

$$\text{s.t. } t_{i+1} \geq t_i + \lambda\delta\left[\mu x_i + f(t_i) - \sigma_i(t_i - t_i^{\text{amb}})\right], \forall i \in \mathcal{I} \tag{6b}$$

$$t_i^{\text{amp}} \leq t_i \leq t^{\max}, \forall i \in \mathcal{I} \cup \{I+1\} \tag{6c}$$

$$0 \leq x_i \leq x_i^{\max}, \forall i \in \mathcal{I} \tag{6d}$$

where $w_i > 0$ is the weight of load x_i. Constraint (6b) is derived from combining (1)-(5), and it dictates the temporal evolution of temperature. Constraints (6b) and (6c) are the bounds on temperature and load, respectively.

Remark 1. *We use inequality* (6b) *instead of equality because the horizon is slotted in the problem. In such a time-slotted horizon, the temperature might exceed the ambient temperature if equality is used, violating constraint* (6c)*. Moreover, it can be easily proved that at the optimum, either* (6b) *holds equality or* $t_{i+1} = t_i^{amb}$*. Hence, no optimality is lost with inequality.*

The convex nature of problem (6), as a result of the convex constraint (6b), permits solutions using tools like CVX. However, employing convex optimization to address the problem needs to predict or estimate the future information over all the time slots, in particular heat dissipation efficiency σ_i. Predictions may lack sufficient accuracy, rendering the solution ineffective. In fact, the problem exhibits online characteristics: decisions at the beginning of each time slot shall be made based on existing information. Given this nature, we apply the RL approach to solve the online version of the problem.

IV. SAC-Based Load Management Policy

The objective of RL is to provide a load management policy enabling PCBS to schedule loads effectively across various states by maximizing the long-term cumulative reward. For this reason, we model the PCBS as an agent, and the action of the agent is the variable to be optimized. The weighted throughput is maximized by finding the highest possible reward. We detail the fundamental components of this agent as follows.

A. State and Action

Based on (1)-(5), for a given time slot i, the agent needs access the current system specifics, which encompass:

- Ambient temperature, t_i^{amb},
- Chip temperature, t_i,
- The weighting factor of the present time slot, w_i,
- Optionally, the heat dissipation efficiency, σ_i.

Note that if a time slot is short, then it is reasonable to assume that the heat dissipation efficiency of the time slot at the very beginning of the slot remains for the entire slot; otherwise, we may not assume the knowledge of heat dissipation for the current slot. Therefore we consider two scenarios:

1) Informed-Heat-Dissipation (IHD) Scenario: In this case, PCBS knows the heat dissipation efficiency for the current time slot. Thus, the state $\boldsymbol{s}_i$ in the time slot i is expressed by

$$\boldsymbol{s}_i = \left[t_i^{\text{amb}}, t_i, w_i, \sigma_i\right]. \tag{7}$$

2) Uninformed-Heat-Dissipation (UHD) Scenario: In contrast, the heat dissipation efficiency is not known, so we utilize $\boldsymbol{s}_i'$ to delineate its state as:

$$\boldsymbol{s}_i' = \left[t_i^{\text{amb}}, t_i, w_i\right]. \tag{8}$$

With state $\boldsymbol{s}_i$ in time slot i, the agent takes an action in action space. The action a_i is defined as

$$a_i = x_i. \tag{9}$$

B. Denial and Reward Mechanism

We employ SAC to refine policies with the aim of maximizing rewards in the environment [2]. SAC finds the maximum of the expected sum of rewards and takes the expected entropy objective to adopt stochastic policies over $\pi_\tau(\boldsymbol{s}_i)$ into consideration. The maximum entropy objective function is defined as:

$$J(\pi) = \sum_{i=0}^{I} \mathbb{E}_{(\boldsymbol{s}_i, a_i)\sim\rho_\pi}[r(\boldsymbol{s}_i, a_i) + \alpha H(\pi(\cdot|\boldsymbol{s}_i))], \tag{10}$$

where $r(\boldsymbol{s}_i, a_i)$ is the reward function, and α is a factor to determine the importance of entropy relative to the reward, and $\pi(\cdot|\boldsymbol{s}_i)$ is the probability distribution of subsequent actions given the state $\boldsymbol{s}_i$, typically resorting to a Gaussian distribution. Additionally, $H(\pi(\cdot|\boldsymbol{s}_i))$ embodies the entropy component, represented as $H(\pi(\cdot|\boldsymbol{s}_i)) \triangleq \mathbb{E}_a[-\log(\pi(a|\boldsymbol{s}_i))]$. It is important to highlight that the agent actively tries various actions to maximize the target entropy. This strategy improves the exploratory nature of SAC.

We would like to have SAC to be implemented in a practical environment instead of a synthetic one. In other words, we want to train the agent in an online environment, i.e., implementing online RL, as the synthetic environment might not match the practical one. However, an action from the action space might make the chip temperature exceed the safe temperature $t^{\max}$. For this reason, we provide the denial and reward mechanism that confines the temperature below the safe temperature and at the same time gives the agent correct feedback. We use Π to represent the zone of policy π, and it is defined as

$$\begin{aligned}\Pi = \big\{\pi | t_i + \mathbb{E}_{\sigma_i, x_i\sim\pi(\cdot|\boldsymbol{s}_i)}\big[\mu x_i + f(t_i) \\ - \sigma_i(t_i - t_i^{\text{amb}})\big] \le t_{\max}, \forall i \in \mathcal{I}\big\}.\end{aligned} \tag{11}$$

We define a temperature-safety condition for policy $\pi \in \Pi$ as

$$t_i + \mathbb{E}_{\sigma_i}\big[\mu x_{\max} + f(t_i) - \sigma_i(t_i - t_i^{\text{amb}})\big] \le t_{\max}, \forall i \in \mathcal{I} \tag{12}$$

Note that the policies not satisfying the condition might lead the temperature to exceed the safe temperature $t_{\max}$ in the next time slot if the agent takes full-load action in some earlier time slot. In our denial and reward mechanism, we operate across the policies and react with different rewards and decisions. Namely,

- If it falls outside of zone Π, then the action has to be denied as it will be overheating. Then we set a negative reward and also reject the decision.
- If it falls in zone Π but does not satisfy risk condition (12), then we allow the action while adding a penalty to the reward.
- If it falls in zone Π satisfying risk condition (12), it receives full reward.

However, temperature-safety condition (12) is implicit to the agent when we implement SAC, as computing the expectations above requires knowing π, and π is not learned yet. We need to compute an explicit risk temperature. For time slot i, the risk temperature can be represented using an optimization problem as follows:

$$t_i^{\text{risk}} = \max_{\pi}\ t_i \tag{13a}$$

$$\text{s.t. } t_{\max} \ge t_i + \mathbb{E}_{\sigma_i, x_i\sim\pi(\cdot|\boldsymbol{s}_i)}[\mu x_i + f(t_i) - \sigma_i(t_i - t_i^{\text{amb}})] \tag{13b}$$

Constraint (13b) states if x_i is applied, then the resulting temperature t_{i+1} in the next time slot cannot exceed the safe temperature $t_{\max}$. Solving the optimization exactly requires knowledge of the distribution of the heat dissipation efficiency as well as the learned policy distribution, which are not known a priori to learning. We can use the following approximation to solve problem (13):

$$\begin{aligned}&t_i + \mathbb{E}_{\sigma_i, x_i\sim\pi(\cdot|\boldsymbol{s}_i)}[\mu x_i + f(t_i) - \sigma_i(t_i - t_i^{\text{amb}})] \\ \le& t_i + \mathbb{E}_{\sigma_i}[\mu x_{\max} + f(t_i) - \sigma_i(t_i - t_i^{\text{amb}})] \\ \approx& t_i + \mu x_{\max} + f(t_i) - \bar{\sigma}_i(t_i - t_i^{\text{amb}})\end{aligned} \tag{14}$$

where $\bar{\sigma}_i$ represents the estimated heat dissipation efficiency for the time slot i. Several methods exist to estimate it. For instance, one might employ historical average values or resort to the most conservative estimates, such as the worst recorded in history.

Hence, problem (13) can be approximated as

$$t_i^{\text{risk}} = \max\ t_i \tag{15a}$$

$$\text{s.t. } t_{\max} \ge t_i + \mu x_{\max} + f(t_i) - \bar{\sigma}_i(t_i - t_i^{\text{amb}}) \tag{15b}$$

Note that problem (15) can be efficiently solved using a bisection search method. With the t_i^{risk}, we can define the reward function as

$$r(\boldsymbol{s}_i, a_i) = \begin{cases} w_i x_i, & \hat{t}_{i+1} < t_i^{\text{risk}} \\ w_i x_i - (\hat{t}_{i+1} - t_i^{\text{risk}}), & t_i^{\text{risk}} \le \hat{t}_{i+1} < t_{\max} \\ -(\hat{t}_{i+1} - t_{\max}), & \hat{t}_{i+1} \ge t_{\max} \end{cases} \tag{16}$$

where $\hat{t}_i$ is an estimated temperature calculated before using the denial and reward mechanism. The proposed denial and reward mechanism is shown in Algorithm 1. Note that, for UHD scenario, we use the estimated value $\bar{\sigma}_i$ as the input of σ_i.

Algorithm 1: Denial and Reward Mechanism

Input: w_i, x_i, t_i^{amb}, σ_i
Output: x_i, t_{i+1}, r_i

Calculate t_i^{risk} by solving (15)
$\hat{t}_{i+1} \leftarrow t_i + \lambda\delta\left[\mu x_i + f(t_i) - \sigma_i(t_i - t_i^{\text{amb}})\right]$
if $\hat{t}_{i+1} > t^{\max}$ **then**
  $x_i \leftarrow 0$
  $t_{i+1} \leftarrow t_i + \lambda\delta\left[f(t_i) - \sigma_i(t_i - t_i^{\text{amb}})\right]$
  $r_i \leftarrow -(\hat{t}_{i+1} - t_{\max})$
else if $t_i^{\text{risk}} \leq \hat{t}_{i+1} < t^{\max}$ **then**
  $t_{i+1} \leftarrow \hat{t}_{i+1}$
  $r_i \leftarrow w_i x_i - (\hat{t}_{i+1} - t_i^{\text{risk}})$
else
  $t_{i+1} \leftarrow \hat{t}_{i+1}$
  $r_i \leftarrow w_i x_i$
return x_i, t_{i+1}, r_i

C. Mechanism of Soft Actor-Critic Learning

The state value function $Q(\boldsymbol{s}_i, a_i)$ and the action-state value function $V(\boldsymbol{s}_i)$ of the SAC can be defined as follows:

$$Q(\boldsymbol{s}_i, a_i) = r(\boldsymbol{s}_i, a_i) + \gamma\mathbb{E}_{s_{i+1}\sim p(.|\boldsymbol{s}_i, a_i)}[V(\boldsymbol{s}_{i+1})], \quad (17)$$

$$V(\boldsymbol{s}_i) = \mathbb{E}_{a_i\sim\pi}[Q(\boldsymbol{s}_i, a_i) - \alpha\log(\pi(a_i|\boldsymbol{s}_i))]. \quad (18)$$

We employ function approximators for the policy function, V-function, and Q-function. Stochastic gradient descent is used to optimize the networks iteratively. Our approach includes a tractable policy $\pi_\phi(a_i|\boldsymbol{s}_i)$, a parameterized state value function $V_\psi(s_i)$, and a soft Q-function $Q_\theta(\boldsymbol{s}_i, a_i)$. Here, ϕ, ψ, and θ denote the network parameters. Our algorithm integrates three distinct DNN types: V-network, policy network, and Q-network. To mitigate positive bias during policy enhancement, we employ dual Q-networks. Specifically, parameters θ_1 and θ_2 are used to shape two distinct Q-functions. They are trained individually, optimizing $J_Q(\theta_1)$ and $J_Q(\theta_2)$.

To detail the parameter updates, we first focus on the update of the soft value that comes from the approximation of the state value function. The soft value function is trained by minimizing the squared residual error

$$J_V(\psi) = \mathbb{E}_{\boldsymbol{s}_i\sim\mathcal{D}}\Big[\frac{1}{2}\big((V_\psi(\boldsymbol{s}_i) - \mathbb{E}_{a_i\sim\pi_\phi}\left[Q_\theta(\boldsymbol{s}_i, a_i) - \log\pi_\phi(a_i|\boldsymbol{s}_i)\right]\big)^2\Big], \quad (19)$$

where $\mathcal{D}$ is a replay buffer. Then, the gradient of the equation (19) is estimated using an unbiased estimator

$$\hat{\nabla}_\psi J_V(\psi) = \nabla_\psi V_\psi(\boldsymbol{s}_i)\left[V_\psi(\boldsymbol{s}_i) - Q_\theta(\boldsymbol{s}_i, a_i) + \log\pi_\phi(a_i|\boldsymbol{s}_i)\right], \quad (20)$$

wherein actions are chosen from the current policy set.

Furthermore, the soft Q-function parameter is trained by minimizing the following soft Bellman residual:

$$J_Q(\theta) = \mathbb{E}_{(\boldsymbol{s}_i, a_i)\sim\mathcal{D}}\left[\frac{1}{2}\left(Q_\theta(\boldsymbol{s}_i, a_i) - \hat{Q}(\boldsymbol{s}_i, a_i)\right)^2\right], \quad (21)$$

with $\hat{Q}(\boldsymbol{s}_i, a_i) = r(\boldsymbol{s}_i, a_i) + \gamma\mathbb{E}_{\boldsymbol{s}_{i+1}\sim p}\left[V_{\bar{\psi}}(\boldsymbol{s}_{i+1})\right]$. The gradient for (21) is optimized using stochastic gradients:

$$\hat{\nabla}_\theta J_Q(\theta) = \nabla_\theta Q_\theta(a_i, \boldsymbol{s}_i)\ (\ Q_\theta(\boldsymbol{s}_i, a_i) - r(\boldsymbol{s}_i, a_i) - \gamma V_{\bar{\psi}}(\boldsymbol{s}_{i+1})\ )\ , \quad (22)$$

where a target value network $V_{\bar{\psi}}$ is used for update. The parameter $\bar{\psi}$ is an exponentially moving average of the target value network weight, given by

$$\bar{\psi} \leftarrow \tau\psi + (1-\tau)\bar{\psi}, \quad (23)$$

where τ is a target smoothing coefficient to improve stability. Finally, the policy parameter is learned by minimizing the expected Kullback-Leibler divergence:

$$J_\pi(\phi) = \mathbb{E}_{\boldsymbol{s}_i\sim\mathcal{D}}\left[\mathrm{D}_{\mathrm{KL}}\left(\pi_\phi(\cdot|\boldsymbol{s}_i)\middle\|\frac{\exp(Q_\theta(\boldsymbol{s}_i, \cdot))}{Z_\theta(\boldsymbol{s}_i)}\right)\right]. \quad (24)$$

We use neural network transformation to reparameterize the policy, i.e.,

$$a_i = g_\phi(\epsilon_i; \boldsymbol{s}_i), \quad (25)$$

where ϵ_i is a noise vector. The objective can be rewritten as

$$J_\pi(\phi) = \mathbb{E}_{\boldsymbol{s}_i\sim\mathcal{D}, \epsilon_i\sim\mathcal{N}}\big[\log\pi_\phi(g_\phi(\epsilon_i; \boldsymbol{s}_i)|\boldsymbol{s}_i) - Q_\theta(\boldsymbol{s}_i, g_\phi(\epsilon_i; \boldsymbol{s}_i))\big]. \quad (26)$$

The gradient of (26) can be approximated as

$$\hat{\nabla}_\phi J_\pi(\phi) = \nabla_\phi\log\pi_\phi(a_i|\boldsymbol{s}_i) + \left(\nabla_{a_i}\log\pi_\phi(a_i|\boldsymbol{s}_i) - \nabla_{a_i}Q(\boldsymbol{s}_i, a_i)\right)\nabla_\phi g_\phi(\epsilon_i; \boldsymbol{s}_i). \quad (27)$$

The unbiased gradient estimator extends the deterministic policy gradients to stochastic policies. The training algorithm for SAC-based passive cooling is summarized in Algorithm 2. Upon convergence, we obtain an online load management policy $\pi_\phi(a_i|\boldsymbol{s}_i)$.

V. Simulation Results

In this section, we present simulation results of our proposed SAC-based load management policy. The hyperparameters of the proposed SAC scheme and the system model parameters are summarized in Tables I and II, respectively. We evaluate the SAC-based load management (SAC-LM) policy under both IHD and UHD scenarios that are introduced in Section IV-A. To provide a comprehensive comparison, we consider the following baselines:

1) Oracle: The states of all the time slots are known, and the global optimum of (6) is obtained by the CVX tool.
2) PPO-LM: We also obtain the numerical results from the well-known proximal policy optimization (PPO) algorithm [15], considering both IHD and UHD scenarios. We call PPO-based load management policy PPO-LM.

To show the learning process, Fig. 1 depicts the average reward versus learning episodes for both SAC-based and PPO-

Algorithm 2: Training for SAC-based Passive Cooling

Input: θ_1, θ_2, ψ, ϕ

Output: Trained policy $\pi_\phi(a_i|\boldsymbol{s}_i)$

Initialize θ_1, θ_2, ψ, ϕ and experience memory $\mathcal{D}$

for each episode **do**

  $a_i \sim \pi_\phi(a_i|\boldsymbol{s}_i)$

  $\boldsymbol{s}_{i+1} \sim p(\boldsymbol{s}_{i+1}|\boldsymbol{s}_i, a_i)$

  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\boldsymbol{s}_i, a_i, \boldsymbol{r}_i, \boldsymbol{s}_{i+1})\}$

  Sample from $\mathcal{D}$ and compute $\nabla J_Q(\theta_1)$ and $\nabla J_Q(\theta_2)$ by (22)

  Update Q-networks parameters, $\theta_1 \leftarrow \theta_1 - \lambda_Q \hat{\nabla}_{\theta_1} J_Q(\theta_1)$, $\theta_2 \leftarrow \theta_2 - \lambda_Q \hat{\nabla}_{\theta_2} J_Q(\theta_2)$

  Sample from the fixed distribution and compute $\nabla J_\pi(\phi)$ by (27)

  Update policy network parameter, $\phi \leftarrow \phi - \lambda_\pi \hat{\nabla}_\phi J_\pi(\phi)$

  Sample from current policy and compute $\hat{\nabla}_\psi J_V(\psi)$ by (20)

  Update V network parameter, $\psi \leftarrow \psi - \lambda_V \hat{\nabla}_\psi J_V(\psi)$

  $\bar{\psi} \leftarrow \tau\psi + (1-\tau)\bar{\psi}$

  Update the next state $\boldsymbol{s}_i \leftarrow \boldsymbol{s}_{i+1}$

return $\pi_\phi(a_i|\boldsymbol{s}_i)$

TABLE I
SAC HYPERPARAMETERS

Hyperparameter	Value
Layers	2 fully connected layers
Layer hidden units	64
Activation function	ReLU
Batch size	256
Replay buffer size	1×10^6
Target smoothing coefficient	0.005
Target update interval	1
Discount rate	0.99
Learning iterations per round	1
Learning rate	3×10^{-4}
Optimizer	Adam
Loss	Mean squared error
Entropy target factor	0.2

based methods. These results are obtained with an average ambient temperature of 24°C and an average heat dissipation efficiency of 0.75. It is evident that SAC-LM takes a greater number of episodes to converge than PPO-LM, but SAC-LM outperforms PPO-LM under both IHD and UHD scenarios. This is attributed to SAC's underlying principle of maximum entropy RL, which emphasizes exploration. Furthermore, both SAC-LM and PPO-LM show better results under the IHD scenario. Thus assuming the availability of slightly more knowledge does help.

Fig. 2 illustrates performance in relation to the average ambient temperatures. As anticipated, both SAC-LM and PPO-LM benefit from knowledge of heat dissipation efficiency, as

TABLE II
SYSTEM MODEL PARAMETERS

Parameter	Value
Number of time slots I	100
Safe temperature $t^{\max}$	100°C
Average ambient temperature t_i^{amb}	$[16, 32]$°C
Maximum load $x_i^{\max}$	200 Mbps
Weight factor w_i	$[0, 1]$
Average heat dissipation efficiency σ	$[0.25, 1.25]W/$°C
Reciprocal of thermal capacitance λ	0.007°C/J [11]
load-to-dynamic-power coefficient μ	0.6 W/Mbps [13]
Duration of a time slot δ	30 seconds

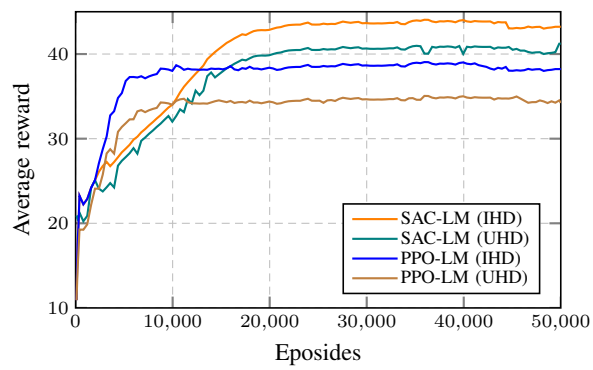

Fig. 1. Average reward versus episodes when the average ambient temperature is 20°C and the average heat dissipation efficiency is 0.75.

seen in the IHD scenario, leading to a more effective policy. With the knowledge of heat dissipation efficiency, SAC-LM can achieve as much as 88.6% of the optimal performance that is only possible with perfect knowledge of future. Even without this information, SAC-LM still manages an impressive performance, reaching 83.2% of the optimal level. Moreover, as the average ambient temperature rises, the weighted sum throughput declines due to lower heat dissipation.

Fig. 3 depicts the relationship between the weighted sum throughput and the average heat dissipation efficiency. As the average heat dissipation efficiency rises, the weighted sum throughput also increases because PCBS can dissipate more heat, thereby handling a greater load. In addition, as the average heat dissipation efficiency decreases, the performance gap between the SAC-LM and PPO-LM methods becomes larger. This is because the PPO algorithm might settle into local optimum, especially in less favorable conditions where the heat dissipation efficiency is particularly low.

Furthermore, we would like to examine the effectiveness of the proposed denial and reward mechanism. We evaluate SAC-LM (i.e., SAC algorithm with the proposed mechanism) and SAC algorithm with a naive reward function for 20,000 episodes, where the average heat dissipation efficiency is 0.75, and the average ambient temperature is 20°C. The naive reward function $\hat{r}(\boldsymbol{s}_i, a_i)$ is defined as follows.

$$\hat{r}(\boldsymbol{s}_i, a_i) = \begin{cases} w_i x_i, & \hat{t}_{i+1} < t^{\max} \\ w_i x_i - (\hat{t}_{i+1} - t_{\max}), & \hat{t}_{i+1} \geq t_{\max} \end{cases} \tag{28}$$

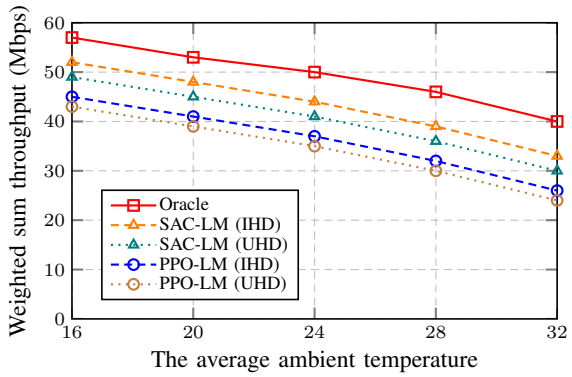

Fig. 2. The weighted sum throughput with respect to the average ambient temperature when the average heat dissipation efficiency is 0.75.

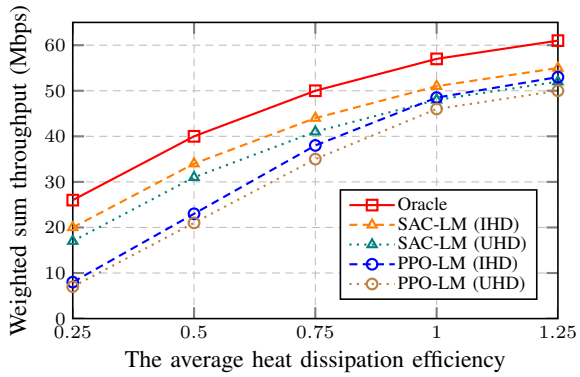

Fig. 3. The weighted sum throughput with respect to the average heat dissipation efficiency when the average ambient temperature is 20°C.

We show the results in Table III. Here, the overheating rate is the ratio of time slots in which the chip temperature surpasses the threshold $t_{\max}$ over the total number of time slots across the 20,000 episodes. A clear observation is that when the proposed denial and reward mechanism is employed, the temperature hardly exceeds $t_{\max}$. Conversely, in the absence of our mechanism, the PCBS is at risk of overheating in numerous time slots. It means that our proposed approach is suitable for a practical environment. Furthermore, the performance after the 20,000 episodes in the table indicates that our proposed mechanism can effectively guide the agent to find a good load management policy.

VI. Conclusion

To address the real-time load management challenges in PCBSs, we introduced a policy based on the SAC algorithm. We have elaborately designed a denial and reward mechanism that can prevent overheating during exploration within the RL framework. Simulation results demonstrate that our mechanism can help the PCBS efficiently find a good load management policy, and it also mitigates the risk of overheating from the exploration of the RL approach, paving the way for practical online RL applications of PCBSs.

TABLE III
Results for verifying the proposed mechanism

	SAC-LM		Naive	
	IHD	UHD	IHD	UHD
Overheating Rate	0%	0.2%	23.0%	47.5%
Performance (Mbps)	44.2	41.3	32.6	30.5

References

[1] Ericsson, "Ericsson 5G portfolio update puts energy efficiency center stage," 2022. [Online]. Available: https://mb.cision.com/Main/15448/3507058/1536906.pdf

[2] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*. PMLR, 2018, pp. 1861–1870.

[3] Y. Aslan, C. E. Kiper, A. Johannes van den Biggelaar, U. Johannsen, and A. Yarovoy, "Passive cooling of mm-wave active integrated 5G base station antennas using CPU heatsinks," in *2019 16th European Radar Conference (EuRAD)*, 2019, pp. 121–124.

[4] Y. Aslan, J. Puskely, A. Roederer, and A. Yarovoy, "Heat transfer enhancement in passively cooled 5G base station antennas using thick ground planes," in *2019 13th European Conference on Antennas and Propagation (EuCAP)*, 2019, pp. 1–5.

[5] K.-W. Duan, J.-W. Shi, and W.-Q. Tao, "Thermal design for the passive cooling system of radio base station with high power density," in *Advances in Heat Transfer and Thermal Engineering: Proceedings of 16th UK Heat Transfer Conference (UKHTC2019)*, 2021, pp. 617–620.

[6] S. K. Sah Tyagi, T. Lin, and Y. Zhou, "Thermal-aware dynamic computing resource allocation for BBU pool in centralized radio access networks," in *IEEE 85th Vehicular Technology Conference (VTC Spring)*, 2017, pp. 1–5.

[7] S. K. Sah Tyagi, Y. Zhou, T. Lin, A. Marahatta, and J. Shi, "Realization of a computational efficient BBU cluster for cloud ran," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–11, August 2018.

[8] J. Wan, X. Gui, R. Zhang, and L. Fu, "Joint cooling and server control in data centers: A cross-layer framework for holistic energy minimization," *IEEE Systems Journal*, vol. 12, no. 3, pp. 2461–2472, 2018.

[9] A. Feriani, D. Wu, Y. T. Xu, J. Li, S. Jang, E. Hossain, X. Liu, and G. Dudek, "Multiobjective load balancing for multiband downlink cellular networks: A meta- reinforcement learning approach," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 9, pp. 2614–2629, 2022.

[10] G. Alsuhli, K. Banawan, K. Attiah, A. Elezabi, K. Seddik, A. Gaber, M. Zaki, and Y. Gadallah, "Mobility load management in cellular networks: A deep reinforcement learning approach," *IEEE Transactions on Mobile Computing*, 2021.

[11] T. Heath, A. P. Centeno, P. George, L. Ramos, Y. Jaluria, and R. Bianchini, "Mercury and freon: Temperature emulation and management for server systems," *ACM SIGOPS Operating Systems Review*, vol. 40, no. 5, pp. 106–116, 2006.

[12] B. Goel, S. A. McKee, and M. Själander, "Techniques to measure, model, and manage power," ser. Advances in Computers. Elsevier, 2012, vol. 87, ch. 2, pp. 7–54.

[13] F. Marzouk, T. Akhtar, I. Politis, J. P. Barraca, and A. Radwan, "Power minimizing BBU-RRH group based mapping in C-RAN with constrained devices," in *IEEE International Conference on Communications (ICC)*, 2020, pp. 1–7.

[14] K. DeVogeleer, G. Memmi, P. Jouvelot, and F. Coelho, "Modeling the temperature bias of power consumption for nanometer-scale CPUs in application processors," in *2014 International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS XIV)*, 2014, pp. 172–180.

[15] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.