Temperature-Constrained Reliability Optimization of Industrial Cyber-Physical Systems Using Machine Learning and Feedback Control

Junlong Zhou, *Member, IEEE*, Liying Li, Ahmadreza Vajdi, Xiumin Zhou, and Zebin Wu, *Senior Member, IEEE*

***Abstract*— As the backbone of Industry 4.0, industrial cyber-physical systems (ICPSs) that are geographically dispersed, federated, cooperative, and security-critical systems become the center of interest from both industry and academia. In ICPS, there are huge amounts of devices, such as sensors and actuators, which are embedded and networked together to improve the performance of real-time monitoring and control. Reliability and temperature are two important concerns of these embedded and networked devices in ICPS due to their stringent requirement of reliable execution and long lifespan. In this article, we study the problem of maximizing soft-error reliability of CPU- and GPU-integrated embedded platforms deployed in ICPS under the temperature constraint. To speed up the estimation of soft-error rate (SER) and temperature, we train an artificial neural network (ANN) that is able to quickly and accurately derive the system's SER and temperature. To solve the temperature-constrained reliability optimization problem, we propose a feedback control-based task scheduling scheme that adaptively determines the number of tasks admitted in the system and the number of replicas for the admitted tasks. We perform a series of simulation experiments to verify the efficacy of our scheme. The experimental results demonstrate that: 1) the estimated SER and temperature derived by our ANN-based method are very close to the ground-truth data and 2) our proposed feedback control-based task scheduling method can improve system reliability by up to 184.2% with a lower peak temperature when compared with one baseline and two state-of-the-art methods.**

***Note to Practitioners*—This article is motivated by the safety-critical industrial cyber-physical system (ICPS) applications necessitating reliable execution and long lifespan, which could be realized by increasing reliability and controlling operating temperature. Our goal is to improve the system reliability of CPU- and GPU-integrated multiprocessor systems-on-chip (MPSoCs) deployed in ICPS under the temperature constraint. Most of the existing papers target either reliability or temperature. A few recent papers have focused on reliability and temperature optimization simultaneously. However, they are not designed for ICPS and do not consider the widely accepted CPU- and GPU-integrated MPSoC platforms. This article proposes a machine learning-based approach that trains an artificial neural network (ANN) to facilitate the online estimation of system SER and temperature. Compared to the offline estimation using simulation tools, the online approach is more applicable to real-time ICPS applications. This article also designs a feedback control-based approach for improving system reliability and reducing peak temperature of the CPU- and GPU-integrated MPSoCs by determining the number of tasks to be admitted and the number of replicas for tasks.**

***Index Terms*— Artificial neural network (ANN), feedback control, industrial cyber-physical systems (ICPSs), soft-error reliability, temperature.**

I. Introduction

INDUSTRIAL cyber-physical system (ICPS) is an emerging technique to address the new demand for real-time and precise monitoring, manipulation, and maintenance of the industrial systems with a wide range of applications, such as factory production line, transportation, and smart construction site [1]–[4]. The essence of real-time and continuous monitoring for critical industrial applications [5]–[7] results in higher power consumption that increases hardware's temperature and in turn may shorten system's lifetime. As reported in [8], a difference in operating temperature from 10°C to 15°C could cause a 2× difference in system lifespan. In addition, devices in ICPS need to deal with the continuous scaling of processor feature size and the ever-increasing processor power density as well as the complicated and harsh environment that could

Manuscript received September 29, 2020; revised January 31, 2021; accepted February 23, 2021. This article was recommended for publication by Editor S. Hu upon evaluation of the reviewers' comments. This work was supported in part by the National Natural Science Foundation of China under Grant 61802185 and Grant 61772274; in part by the Natural Science Foundation of Jiangsu Province under Grant BK20180470 and Grant BK20180018; in part by the China Postdoctoral Science Foundation under Grant 2020M680068; in part by the Fundamental Research Funds for the Central Universities under Grant 30919011233, Grant 30917015104, Grant 30919011103, and Grant 30919011402; in part by the Open Research Fund of Engineering Research Center of Software/Hardware Co-Design Technology and Application [Ministry of Education (MoE)] at East China Normal University; and in part by the Open Research Fund of Shanghai Key Laboratory of Multidimensional Information Processing at East China Normal University under Grant 2020MIP001. *(Corresponding authors: Liying Li; Zebin Wu.)*

Junlong Zhou is with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China, also with the MoE Engineering Research Center of Software/Hardware Codesign Technology and Application, East China Normal University, Shanghai 200062, China, and also with the Shanghai Key Laboratory of Multidimensional Information Processing, East China Normal University, Shanghai 200062, China (e-mail: jlzhou@njust.edu.cn).

Liying Li is with the Department of Computer Science and Technology, East China Normal University, Shanghai 200062, China (e-mail: liyingli1995@foxmail.com).

Ahmadreza Vajdi, Xiumin Zhou, and Zebin Wu are with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China (e-mail: vajdi@njust.edu.cn; xiumin@njust.edu.cn; wuzb@njust.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TASE.2021.3062408.

Digital Object Identifier 10.1109/TASE.2021.3062408

make the devices more vulnerable to soft errors [9]–[12]. Therefore, considering temperature and soft-error reliability simultaneously in ICPS is of special importance.

There are lots of studies concentrating on increasing soft-error reliability. A novel system-level reliability metric for multiprogram workloads is introduced in [13]. Redundancy is effective in alleviating the effect of soft errors in multicore systems. Chen *et al.* [14] attempted to utilize the techniques of system-level simultaneous redundant threading, chip-level redundant multithreading, and mixed redundant threading together in mitigating soft-error effects in multicores. In addition, they develop a series of dynamic federated scheduling-based approaches to increase system reliability under hard task deadline constraints. Chatterjee *et al.* [15] exploited the timing characteristics of tasks to design a dynamic fault-tolerant strategy based on temporal redundancy and/or spatial redundancy. Li *et al.* [16] proposed a framework called DHASER for ASIP-based multicore systems, which provides task-level analysis, analysis-based processor customization, and dynamic reliability-aware task scheduling. DHASER allows each processor to use its specific error recovery according to task characteristics (i.e., soft-error vulnerability and task execution deadline). A mapping strategy [17] considers that the reliability and performance heterogeneity of tasks and cores is proposed for redundant multithreading in multicore systems. A series of state-of-the-art approaches [18]–[20] is presented to optimize the reliability of multicore systems suffering both soft errors and hard errors. However, none of the above-mentioned work considers temperature.

Considerable research efforts have been devoted to reducing temperature. Thermal-aware scheduling is useful in optimizing thermal profiles of tasks on processors. A system-level thermal-aware scheduling method is designed in [21] in order to derive the post-thermal map from the current thermal profiles based on lookup tables. Yun *et al.* [22] design a thermal-aware scheduler based on the techniques of job migration and power gating, to mitigate the generation of thermal hotspots on multicore systems. In another thermal-aware optimization approach [23], the authors focus on minimizing the latest completion time (makespan) by determining task-to-core mapping and tasks' operating speed/voltage. Zhou *et al.* [8] designed a two-stage thermal-aware task scheduling scheme for minimizing the energy consumption of heterogeneous real-time multiprocessor systems-on-chip (MPSoCs). Zhou *et al.* [24] also developed a task splitting technique that decreases system peak temperature by alternatively executing hot tasks and slack time. Utilizing a multiobjective evolutionary algorithm for task scheduling is another attempt [25] that optimizes performance, energy, and temperature simultaneously. Unlike the above-mentioned methods [8], [21]–[25] that are designed for traditional planar (2-D) processors, the research effort [26] focuses on proposing a thermal-aware task scheduling scheme for reducing the temperature of 3-D processors and the work [27] reviews the state-of-the-art optimization techniques for thermal-aware 3-D processors. However, all these works do not deal with soft errors to optimize reliability.

A few recent papers have focused on reliability and temperature simultaneously. As one of the first works, Zhou and Wei [28] proposed a stochastic task scheduling approach that uses a thermal-aware task sequencing approach for reduction of processor temperature and features by considering the uncertainty in the occurrence of soft errors. To investigate the reliability and temperature-constrained makespan minimization problem deeply, a mixed-integer linear programming (MILP) formulation is presented in [29]. Since the MILP approach generally takes exponential time to finish, the authors developed a two-stage heuristic algorithm that determines the assignment, replication, operating frequency, and execution order for tasks to minimize the makespan under the thermal and reliability constraints. Although the above-mentioned approaches are very useful to improve reliability and temperature, these aforementioned approaches are not designed for ICPS in which there are significant uncertainties and disturbances to be handled. In addition, they do not consider the widely accepted CPU- and GPU-integrated MPSoCs, which are suitable platforms for real-time embedded applications requiring massively parallel processing. Ma *et al.* [30] proposed a hybrid framework that optimizes soft-error reliability of CPU- and GPU-integrated platforms under the constraints of temperature, deadline, and lifetime reliability. However, the framework is not designed for ICPS and does not use emerging learning techniques.

For better readability, Table I shows a summary of the related papers discussed in this article. As can be seen from the table, there is no existing work considering soft-error reliability and temperature simultaneously for CPU- and GPU-integrated MPSoCs in ICPS. Unlike the existing work, we are interested in the reliability and temperature concerns of CPU- and GPU-integrated embedded systems deployed in ICPS. For ICPS applications running on such platforms, we focus on addressing the problem of maximizing system soft-error reliability under the temperature constraint. We adopt the machine learning and feedback control techniques to solve the temperature-constrained reliability optimization problem, which are powerful in handling the uncertainties and disturbances in ICPS. To be specific, the main contributions of this article are summarized as follows.

1) We formulate the temperature-constrained reliability optimization problem for CPU- and GPU-integrated embedded systems deployed in ICPS. To the best of our knowledge, this is the first work focusing on reliability and temperature concerns for ICPS running on CPU and GPU integrated MPSoC systems.
2) We train an artificial neural network (ANN) to facilitate the online estimation of system soft-error rate (SER) and temperature. Compared to the offline estimation using simulation tools, which is generally time-consuming, the trained network is more applicable to ICPS applications that typically require a real-time response.
3) We design a feedback control-based task scheduling scheme that determines the number of tasks admitted in the system and the number of replicas for tasks, in order to improve reliability and reduce the temperature.

Extensive simulations are carried out to validate the proposed machine learning and feedback control-based task scheduling scheme. The results demonstrate that our proposed scheme

TABLE I
CLASSIFICATION OF RELATED WORK IN THE LITERATURE

References	ICPS	Soft-Error Reliability	Temperature	CPU-GPU Integrated MPSoCs	3D Processor	Timing	Machine Learning	Task Scheduling
Zhou et al. [8]			√			√		√
Naithani et al. [13]		√						√
Chen et al. [14]		√				√		√
Chatterjee et al. [15]		√						√
Li et al. [16]		√						√
Chen et al. [17]		√						√
Zhou et al. [18]		√				√		√
Ma et al. [19]		√	√			√		√
Zhou et al. [20]		√				√		√
Cui et al. [21]			√			√		√
Yun et al. [22]			√					√
Hanumaiah et al. [23]			√			√		√
Zhou et al. [24]			√			√		√
Sheikh et al. [25]			√					√
Zhou et al. [26]			√		√			
Cao et al. [27]			√		√			√
Zhou et al. [28]		√	√			√		√
Zhou et al. [29]		√	√			√		√
Ma et al. [30]		√	√	√		√		√
This paper	√	√	√	√		√	√	√

increases system reliability by up to 184.2% with a lower peak temperature compared to three peer approaches.

The rest of this article is organized as follows. In Section II, we introduce the system models used in the proposed scheme. Section III formulates the studied problem and provides a high-level overview of the proposed scheme. Sections IV and V describe the proposed ANN-based dynamic SER and temperature estimation and the proposed feedback control-based task scheduling in detail, respectively. Section VI shows the experimental results and analysis. Concluding remarks are given in Section VII. Table II summarizes the definition of main notations used in the system model and problem formulation.

II. SYSTEM MODELS

This section introduces the system models used in embedded devices deployed in ICPS, including the processor and task models, temperature model, holistic cross-layer SER model, and reliability model.

A. Processor and Task Models

Consider a DVFS-enabled MPSoC that integrates one GPU (represented by ρ_G) and M homogeneous CPUs (represented by $\rho_C = \{\rho_1, \rho_2, \ldots, \rho_M\}$). The GPU is equipped with P discrete frequencies $\{f_1^G, f_2^G, \ldots, f_P^G\}$ that correspond to P voltages $\{v_1^G, v_2^G, \ldots, v_P^G\}$, where $f_{\min}^G = f_1^G \leq f_2^G \leq \cdots \leq f_P^G = f_{\max}^G$ holds. All frequencies are normalized to $f_{\max}^G$, that is, $f_{\max}^G = 1.0$. Each CPU ρ_i $(1 \leq i \leq M)$ in ρ_C supports multiple frequencies from the minimum frequency to the maximum frequency, which is represented by $f_{\min}^C$ and $f_{\max}^C$, respectively. ρ_i also supports W discrete normalized frequencies $\{f_1^C, f_2^C, \ldots, f_W^C\}$ corresponding to W voltages $\{v_1^C, v_2^C, \ldots, v_W^C\}$ where $f_{\min}^C = f_1^C \leq f_2^C \leq \cdots \leq f_W^C = f_{\max}^C$ holds and the frequencies are normalized with respect to $f_{\max}^C$ where $f_{\max}^C = 1.0$.

A task set Γ consisting of n independent real-time tasks, represented by $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_n\}$, is assumed to be executed on the CPU–GPU integrated MPSoC platform. Each task could be executed on either CPU or GPU. Each real-time task τ_i $(1 \leq i \leq n)$ can be modeled as $\tau_i : \{P_i, D_i, B_i\}$, where P_i and D_i denote the period and the deadline of task τ_i, respectively. B_i is the execution time of task τ_i at the maximum frequency ($f_{\max}^G$ for GPU or $f_{\max}^C$ for CPU). Given B_i, the execution time of τ_i at frequency f, denoted by E_i, can be derived by $E_i = B_i/f$. If task τ_i is executed on the GPU (ρ_G), $f \in \{f_1^G, f_2^G, \ldots, f_P^G\}$ holds; otherwise, task τ_i is executed on the CPU and $f \in \{f_1^C, f_2^C, \ldots, f_W^C\}$ holds.

TABLE II
MAIN NOTATIONS USED IN SYSTEM MODEL AND PROBLEM FORMULATION

Notation	Definition
ρ_G	the GPU core
ρ_C	the set of CPU cores
$\rho_1, \rho_2, ..., \rho_M$	M homogeneous CPU cores
$f_1^G, f_2^G, ..., f_P^G$	P discrete frequencies supported by the GPU core
$v_1^G, v_2^G, ..., v_P^G$	P discrete voltages supported by the GPU core
$f_1^C, f_2^C, ..., f_W^C$	W discrete frequencies supported by the CPU cores
$v_1^C, v_2^C, ..., v_W^C$	W discrete voltages supported by the CPU cores
$\tau_1, \tau_2, ..., \tau_n$	n independent real-time tasks in set Γ
P_i	the period of task τ_i
D_i	the deadline of task τ_i
B_i	the execution time of task τ_i at the maximum frequency
E_i	the execution time of task τ_i at frequency f
SER	soft-error rate
SER_{cir}	the circuit layer SER
SER_{com}	the component layer SER
SER_{sys}	the system layer SER
R_i	the reliability of task τ_i due to soft errors
R_i^{rep}	the reliability of task τ_i with n replicas
R_{sys}	the system reliability
$T(t)$	the instantaneous temperature at time t
T_{sst}	the steady state temperature
T_{int}	the initial temperature
T_{amb}	the chip die's ambient temperature
$\Re$	the thermal resistance of a core
θ	the thermal capacitance of a core
ϕ	the power consumption of a core
U	the system utilization
T_{peak}	the peak temperature
ε	a pre-defined temperature threshold

B. Holistic Cross-Layer SER Model

Energetic neutrons hit transistors of electronic systems, and some electrical charge is generated. The generated electrical

charge may temporally change transistor state, which may lead to unexpected results such as data missing. Hazucha and Svensson [32] presented a method to derive the circuit layer SER, denoted by $\mathrm{SER}_{\mathrm{cir}}$, by using

$$\mathrm{SER}_{\mathrm{cir}} = F \times CS_{\mathrm{cir}} \tag{1}$$

where F represents the flux of neutrons, which is dependent on geographic locations, and CS_{cir} denotes the atmospheric neutron cross section, which can be derived by

$$CS_{\mathrm{cir}} = A_{\mathrm{cir}} \times K_{\mathrm{cir}} \times \exp\left\{-Q'_c/Q'_s\right\} \tag{2}$$

where A_{cir} represents the circuit area that is sensitive to soft errors, K_{cir} is a constant depending on the technology and design of the circuit, Q'_c denotes the threshold, above which circuit malfunction occurs, and Q'_s represents the charge collection efficiency.

After the soft errors first appear in the circuit layer, they may propagate and reach the component layer [31], [33], [34]. Riera *et al.* [34] proposed an approximate SER calculation method for the basic components containing SRAM, latches, and logic gates

$$\mathrm{SER}_{\mathrm{com}} = F \times A_{\mathrm{com}} \times K_{\mathrm{com}} \times \exp\{-Q_c/Q_s\} \tag{3}$$

where F is neutron flux, A_{com} denotes the area of the component that is sensitive to soft errors, K_{com} is a constant depending on the type of component, Q_c represents the threshold value, above which a component malfunction occurs, and Q_s is the component charge collection efficiency, which is in fact the average of Q'_c in (2).

Some soft errors at the component layers may be mitigated due to various degrading factors, while the others propagate and finally arrive the system layer by corrupting either data or instructions of applications [34]. There are different degrading factors associated with errors generated in combinational gates, flip-flops, and memory arrays. Readers are referred to [33] for further details about the degrading factors. We adopt architectural vulnerability factor (AVF) to model the SER at the system layer as

$$\mathrm{SER}_{\mathrm{sys}} = \sum_{i=1}^{L} \mathrm{AVF}_i \times \alpha_i \times \mathrm{SER}_{\mathrm{com}_i} \tag{4}$$

where AVF_i is the AVF value of component of type i ($1 \le i \le L$), which can be calculated by using the method in [35]. L represents the number of different component types. α_i denotes the ratio of type i component to the number of all components. $\mathrm{SER}_{\mathrm{com}_i}$ represents the SER of type i component and can be derived by (3).

C. Reliability Model

The reliability model due to soft errors is applicable to both GPU and CPU since GPU and CPU are identical at the device level [30]. Following the exponential reliability model [30], which is defined as the probability that no soft errors occur during the task execution on the core, the reliability of task τ_i is formulated as

$$R_i = \exp\left\{-\mathrm{SER}_{\mathrm{sys}} \times B_i/f\right\} \tag{5}$$

where f is the operating frequency of the core. Assume that there are n replicas for τ_i. Then, with the consideration of these replicas, the reliability of task τ_i is derived as

$$R_i^{\mathrm{rep}} = 1 - (1 - R_i)^{n+1}. \tag{6}$$

The system overall reliability, represented by R_{sys}, depends on whether all tasks are executed successfully on the cores. Thus, it is expressed as

$$R_{\mathrm{sys}} = \prod_{i=1}^{N} R_i \tag{7}$$

where R_i can be replaced by R_i^{rep} if the task is replicated.

D. Temperature Model

It is crucial to accurately qualify the thermal property of electronic devices in ICPS. In this article, we adopt a widely accepted dynamic thermal model in [37] to capture the temperature profiles of CPU and GPU cores. The instantaneous temperature of a core at time instance t can be obtained by

$$T(t) = T_{\mathrm{sst}} - (T_{\mathrm{sst}} - T_{\mathrm{int}}) \times \exp\{-t/(\Re \times \theta)\} \tag{8}$$

where T_{sst} is the core's steady-state temperature, T_{int} denotes the core's initial temperature, and $\Re$ and θ are the core's thermal resistance and capacitance, respectively. Both $\Re$ and θ are two processor architecture-dependent constants. T_{sst} is calculated as

$$T_{\mathrm{sst}} = T_{\mathrm{amb}} + \phi \times \Re \tag{9}$$

where T_{amb} is the chip die's ambient temperature and ϕ is the core's power consumption that can be derived using the CMOS power model [24] as

$$\phi = \mu\left(\lambda f^3 + \xi f + \kappa f T\right) \tag{10}$$

where μ is an active factor manifesting the heterogeneous nature of tasks, f is the core's frequency, λ denotes the effective switching capacitance, T is the chip's temperature derived by (8), and ξ and κ represent two curve fitting constants depending on the processor.

III. Problem Definition and Methodology Overview

In this article, we study the problem of maximizing soft-error reliability of CPU- and GPU-integrated embedded platforms deployed in ICPS under the temperature and real-time constraints. Our goal is to design a fault-tolerant task scheduling methodology for improving system soft-error reliability while satisfying temperature and timing constraints. In the following, we first define the problem to be addressed and then present an overview of the proposed methodology.

A. Problem Definition

We adopt the earliest deadline first (EDF) policy to schedule tasks on the MPSoC system. Although we use the EDF policy in this article, the proposed optimization method is applicable to other scheduling policies. The scheduling algorithm used in this article is a partitioned approach. Under partitioned scheduling, tasks are statically mapped to cores and task migration is not allowed. Under the EDF policy, tasks in a task set are schedulable if the system utilization is no greater than 1. Our goal is to maximize the system reliability defined in (7) under the temperature constraint. The optimization problem of the system reliability is thus formulated as

$$\begin{aligned} \text{Maximize:} \quad & R_{\text{sys}} \\ \text{Subject to:} \quad & U(\rho_G) \le 1 && (11) \\ & U(\rho_i) \le 1 \quad \forall \rho_i \in \rho_C && (12) \\ & T_{\text{peak}}(\rho_G, \rho_C) \le T_{\text{th}} && (13) \end{aligned}$$

where $U(\rho_G)$ is the utilization of the GPU core ρ_G and $U(\rho_i)$ is the utilization of any CPU core $\rho_i \in \rho_C$. The utilization U of the GPU or any CPU core in a given time interval is defined as the ratio of the amount of time that the core executes tasks and the interval. $T_{\text{peak}}(\rho_G, \rho_C)$ is the peak temperature of tasks executed on the GPU- and CPU-integrated MPSoC and T_{th} is a predefined temperature threshold that is specified based on the requirement of applications in ICPS. The peak temperature T_{peak} is computed as

$$T_{\text{peak}}(\rho_G, \rho_C) = \max_{\forall \rho_i \in \rho_C, \forall t \in [0,H]} \{T(\rho_G, t), T(\rho_i, t)\} \quad (14)$$

where $T(\rho_G, t)$ and $T(\rho_i, t)$ are the instantaneous temperature of the GPU core ρ_G and CPU core ρ_i at time instance t, respectively, and $[0, H]$ is a schedule length for tasks executing on the cores. The first and second constraints ensure that the tasks are schedulable on the GPU and CPU cores under the real-time requirement. The third constraint guarantees that the system peak temperature is below a safe threshold.

B. Methodology Overview

For better understanding, Fig. 1 shows a high-level overview of our proposed machine learning and feedback control-based task scheduling scheme for optimizing system soft-error reliability under temperature and timing constraints. The proposed task scheduling scheme consists of two components. One component is an ANN-based dynamic approach that is proposed to quickly and accurately estimate system SER and temperature profiles. The approach exploits the simulated SER and temperature data to train an ANN. The simulated SER and temperature data are generated by the SPICE tool [38] and the HotSpot tool [37] offline, which are developed based on the cross-layer SER model (see Section II-B) and the instantaneous temperature model (see Section II-D), respectively. The trained network can be used to estimate system SER and temperature dynamically. The other component is a feedback control-based approach that is proposed to optimize system reliability and control system peak temperature. Depending on the online estimation of system SER and temperature via the trained network, the approach jealously and adaptively determines the number of tasks admitted into the system and the number of replicas for admitted tasks, in order to increase system reliability and reduce chip temperature. The output of our proposed two-component scheme is a feedback control-based task scheduling solution.

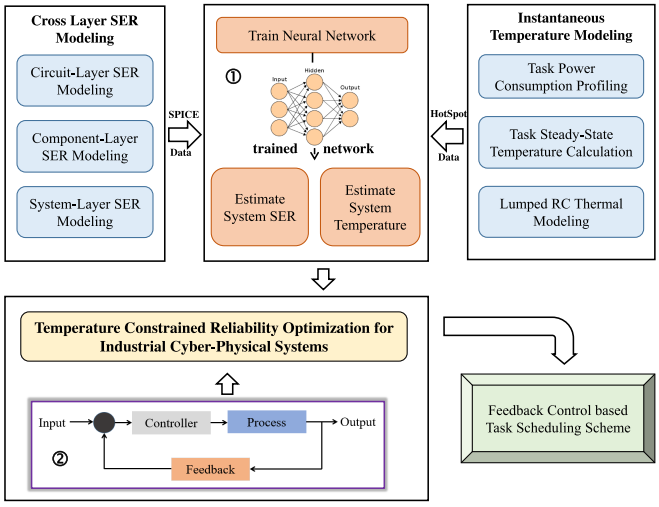

Fig. 1. Overview of the proposed machine learning and feedback control-based task scheduling scheme.

IV. Proposed ANN-Based Dynamic SER and Temperature Estimation

We aim at improving system soft-error reliability while satisfying temperature and timing constraints. To this end, it is crucial to dynamically and accurately obtain the system SER and temperature. A common method to derive the system SER and the temperature is to use simulation tools, such as SPICE and HotSpot simulator. However, it usually takes a long time to get simulation results by using the methods based on simulations. Therefore, these methods are not applicable to real-time ICPS applications. In this article, we proposed an ANN-based method to quickly and accurately obtain the system SER and temperature. We first introduce the basics of ANN and then present the ANN-based estimation method.

A. Artificial Neural Network

ANN is a widely accepted machine learning method due to its versatility and accuracy. A typical architecture of an ANN is shown in Fig. 2. Generally, a typical ANN architecture contains an input layer, one or more hidden layers, and one output layer. Each layer in the ANN is composed of one or more neuron nodes and layers are connected through multiple weighted connections. Each neuron node has one or more inputs and produces a single output that can be sent to multiple other neurons.

The training process of an ANN contains forward propagation and backward propagation. In the forward propagation, the neuron nodes in the input layer take n_{in} external signals $\{I_1, I_2, \ldots, I_{n_{\text{in}}}\}$ as inputs and deliver these signals to the neuron nodes in hidden layers. Then, the neuron nodes in the hidden layer calculate these input signals according to

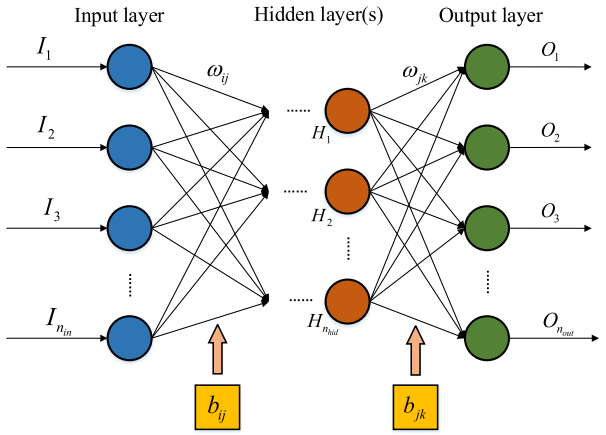

Fig. 2. Typical architecture of an ANN topology.

the trainable weights and biases. Let n_{hid} be the number of the hidden layer neuron nodes. The output of the jth ($1 \le j \le n_{\text{hid}}$) neuron node of the hidden layer can be calculated by using

$$H_j = \sigma\left(\sum_{i=1}^{n_{\text{in}}} \omega_{ij} I_i + b_{ij}\right) \tag{15}$$

where the parameters ω_{ij} and b_{ij} ($1 \le i \le n_{\text{in}}, 1 \le j \le n_{\text{hid}}$) denote the trainable parameters weight and bias between the ith input layer neuron node and jth hidden layer node. σ is a tanh activation function expressed as

$$\sigma(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}. \tag{16}$$

Activation functions increase the ability of neural networks to fit nonlinear relationships, which is necessary to improve the performance of the neural networks. Like the logistic sigmoid, the tanh function is also sigmoidal (S-shaped) but instead outputs the values that range $(-1, 1)$. Thus, strongly negative inputs to the tanh will map to negative outputs. In addition, only zero-valued inputs are mapped to near-zero outputs. These properties make the network less likely to get stuck during training. Although we use the tanh function as the activation function in this work, other activation functions, such as the rectified linear unit (ReLU) function and Sigmoid, are also applicable. The signals processed by the hidden layer(s) are finally propagated to and processed by the output layer neuron nodes. Let n_{out} be the number of output layer neuron nodes, and the result of the kth ($1 \le k \le n_{\text{out}}$) output layer node is

$$O_k = \sigma\left(\sum_{j=1}^{n_h} \omega_{jk} H_j + b_{jk}\right) \tag{17}$$

where ω_{jk} and b_{jk} ($1 \le j \le n_{\text{hid}}, 1 \le k \le n_{\text{out}}$) are the weight and bias between the jth neuron node in the last hidden layer and the kth output layer node.

In the backward proportion, the mean squared error (MSE)/loss between the result of the output layer and the ground truth, which is represented by e, is calculated as

$$e = \frac{1}{n_{\text{out}}} \sum_{k=1}^{n_{\text{out}}} (G_k - O_k)^2 \tag{18}$$

where G_k and O_k denote the ground truth and actual result of the kth neuron node in the output layer, respectively. If the loss is larger than the predefined threshold, it is feedback from the output layer to the hidden layer(s) and finally to the input layer of the ANN. ANN continuously adjusts the trainable parameters (weights and biases) to improve the accuracy of its estimation performance. Specifically, the partial derivative of the loss relative to each weight is calculated, and then, the weight is updated via some approach, such as stochastic gradient descent and extreme learning machines [36]. The training process is repeated until the loss between the result of the output layer and the ground truth is smaller than a threshold value, which depends on the system design requirements.

The details on our ANNs developed for system SER and temperature estimation are presented in Section IV-B.

B. ANN-Based Dynamic SER and Temperature Estimation

For system SER estimation, we first adopt the holistic cross-layer SER model in Section II-A for SPICE simulation. The obtained data from the SPICE simulation are used for ANN training. Then, the trained ANN is utilized for system SER estimation. We select four fluctuate variables, including temperature, voltage, neuron flux, and critical charge of the system in our cross-layer system SER model as the input signals of the ANN. The neuron flux (F) varies with geographic location of the system. The critical charge of the system is in fact a charge threshold, beyond which a malfunction of a system occurs. It is related to the critical charge of the components in the system. Let Q_c^{sys} be the critical charge of the system, and it can be derived by using

$$Q_c^{\text{sys}} = \sum_{i=1}^{L} \alpha_i \times Q_{c_i} \tag{19}$$

where L is the number of component types, α_i denotes the proportion of type i component in all types of components, and Q_{c_i} represents the critical charge of the type i component. It is actually the threshold value, above which a circuit malfunction of the type i component occurs. The ANN takes the above as input signals and then quickly and accurately outputs the estimated system SER. Therefore, the numbers of input and output neuron nodes are 4 and 1, respectively. We adopt an empirical method to calculate the number of neuron nodes in the hidden layers, which is expressed as

$$n_{\text{hid}} = \frac{n_{\text{train}}}{\gamma \times (n_{\text{in}} + n_{\text{out}})} \tag{20}$$

where n_{train}, n_{in}, and n_{out} denote the numbers of training samples, input layer nodes, and output layer nodes, respectively. γ is a constant number between 2 and 10.

Similarly, we use temperature data from HotSpot simulation to train the ANN model and adopt the pretrained model for the temperature prediction. We select the thermal resistance ($\Re$), thermal capacitance (θ), power consumption (ϕ), time

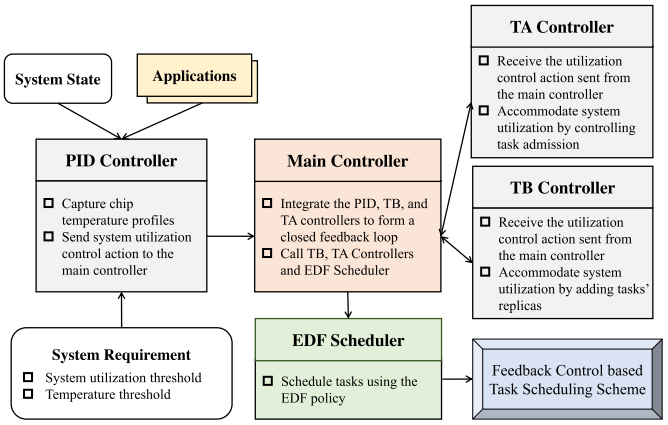

Fig. 3. Overview of the proposed feedback control task scheduling scheme.

(t), and die's ambient temperature (T_{amb}) as the inputs of the ANN. The ANN is trained to output predicted temperature (T). The values of thermal resistance ($\Re$) and thermal capacitance (θ) depend on the processor architecture. Different tasks in the task set have different power consumption (ϕ). The chip die's ambient temperature (T_{amb}) varies over time. ANN takes the above four variables as input signals and is trained to accurately outputs the predicted temperature (T). The number of hidden layer neuron nodes is obtained by using (20).

Note that the training data for two neural networks (one for system SER estimation and the other for temperature estimation) are obtained from the SPICE and HotSpot simulations, respectively. The simulation processes are only used to obtain the relevant data and the ANNs are trained offline. In contrast, our prediction is executed online to replace the time-consuming simulation process. After the offline training, the trained ANNs can be used online to quickly and accurately estimate the system SER and the temperature. Therefore, it is unnecessary to update or retrain the neural network models at runtime.

V. Proposed Feedback Control-Based Task Scheduling Scheme

This section first presents the overview of our proposed feedback control-based task scheduling scheme and then shows the details on the proposed algorithm.

A. Overview

Control theory has been widely used in many aspects of ICPS. Unlike open-loop control that needs to know everything accurately to work right, feedback (closed-loop) control does not require knowing everything and allows making errors in estimation and prediction. Thus, feedback control is more suitable for managing systems in ICPS, which deeply interacts with uncertain environments. Fig. 3 shows an overview of our proposed feedback control-based task scheduling scheme. The proposed scheme incrementally solves the optimization problem by exploiting the history of system states in the previous profiling window. As demonstrated in the figure, it is mainly composed of a proportional–integral–derivative (PID) controller, a main controller, a task access (TA) controller, a task backup (TB) controller, and an EDF scheduler. The controllers and scheduler cooperate to adjust the system state and determine the scheduling and replication of tasks. The PID controller captures the tasks' temperature and sends the utilization control action $\Delta U(\rho_G)$ and $\Delta U(\rho_i)$ for GPU and CPU cores to the main controller for reducing temperature if the tasks' peak temperature exceeds a temperature threshold. $\Delta U(\rho_G)$ [or $\Delta U(\rho_i)$] is the amount of utilization of the GPU ρ_G (or CPU core ρ_i) that should be increased (i.e., $\Delta U > 0$) or decreased (i.e., $\Delta U < 0$) and it depends on the difference between the system's current peak temperature and the temperature threshold. The proposed scheme uses the TA and TB controllers to accommodate the core utilization. Specifically, it first uses the TB controller to accommodate core utilization by adding or removing the replicas of tasks. It then uses the TA controller to accommodate the rest of ΔU by controlling the tasks assigned to the core if the TB controller cannot accomplish the required ΔU. The proposed scheme finally uses the EDF scheduler to schedule the assigned tasks along with their replicas.

B. Feedback Control-Based Task Scheduling Algorithm

The pseudocode of our feedback control scheme is described in Algorithm 1. The algorithm takes the tasks to be executed and the GPU–CPU-integrated MPSoC platform as input, represented by $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_n\}$ and $\rho = \{\rho_G, \rho_1, \rho_2, \ldots, \rho_M\}$. The algorithm first initializes the mapping, operating frequency, and replication for tasks using the reliability-aware approach proposed in [39] (lines 1-2). During the mapping of tasks to cores, we treat GPU and CPU as equal since they are identical at the device level. The reliability and temperature models are applicable to both GPU and CPU. Based on the initial solution with EDF scheduling, the algorithm estimates system states, such as the GPU and CPU cores' utilization and temperature profiles (line 3). If the system meets all the constraints given in (11)–(13), the algorithm directly schedules tasks using the EDF scheduler (line 16). Otherwise, it iteratively uses the feedback control scheme to adjust the scheduling solution until the design constraints are met (lines 4–15).

During the iteration, the algorithm calculates the total utilization accommodation ΔU for the GPU (line 5) and CPU cores (line 11) using the PID controller and uses the ΔU as the input to the TB controller that is responsible for deriving the core utilization accommodation ΔU_{TB} and determining the updated number of replicas for tasks (lines 6 and 12). After obtaining ΔU_{TB}, the utilization ΔU_{TA} accommodated by the TA controller can be readily derived as $\Delta U - \Delta U_{\text{TB}}$ (line 7 and line 13). The TA controller needs to accommodate the core utilization ΔU_{TA} if $\Delta U_{\text{TA}} \neq 0$ (lines 8, 9, 14, and 15). The details on the PID controller, TA controller, TB controller, main controller, and EDF scheduler are presented in the following.

1) *The PID Controller:* It is a control loop mechanism that can strengthen the robustness of a control process against external disturbances by continuously computing the difference between a desired set point and a measured variable and then utilizing a correction, which is based on proportional, integral, and derivative terms. It first

Algorithm 1 Feedback Control-Based Task Scheduling

Input: The tasks $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_n\}$ to be executed; The GPU-CPU integrated MPSoC platform $\rho = \{\rho_G, \rho_1, \ldots, \rho_M\}$;

```
for τ_i ∈ Γ do
    initialize the mapping, operating frequency, and replication of task τ_i using the approach [39];
estimate the GPU and CPU cores' utilization and temperature profiles based on the initial solution with EDF scheduling;
while Constraints in Eqs. (11)-(13) are not met do
    use the PID controller to decide the utilization accommodation of the GPU, represented by ΔU(ρ_G);
    derive the utilization accommodation ΔU_TB(ρ_G) for the GPU using the TB controller;
    calculate the utilization ΔU_TA(ρ_G) accommodated by the TA controller as ΔU(ρ_G) − ΔU_TB(ρ_G);
    if ΔU_TA(ρ_G) ≠ 0 then
        use the TA controller to accommodate ΔU_TA(ρ_G);
    for ρ_i ∈ ρ_C do
        use the PID controller to decide the utilization accommodation of CPU core ρ_i, represented by ΔU(ρ_i);
        derive the utilization accommodation ΔU_TB(ρ_i) using the TB controller;
        calculate the utilization ΔU_TA(ρ_i) accommodated by the TA controller as ΔU(ρ_i) − ΔU_TB(ρ_i);
        if ΔU_TA(ρ_i) ≠ 0 then
            use the TA controller to accommodate ΔU_TA(ρ_i);
schedule tasks using the EDF scheduler;
```

samples the tasks periodically to derive the system peak temperature. When the peak temperature is higher than a predefined temperature threshold ε, it then obtains the control variable ΔU using the basic PID control formula [40] and delivers the derived ΔU to the main controller.

2) *The TB Controller:* It increases or decreases the core utilization by adding or removing replicas for tasks. As indicated in (7), the system reliability R_{sys} is maximized when all tasks' reliabilities are equal and maximal. Thus, when increasing core utilization for improving system reliability, the approach adds the replicas for tasks in a way that makes the tasks' reliabilities equal. When decreasing core utilization for reducing system peak temperature, the approach gradually removes the replicas of tasks.
3) *The TA Controller:* It is only used when the TB controller is unable to accommodate the whole utilization adjustment ΔU. The controller is responsible for completing the rest of ΔU by controlling the number of tasks that could be handled by the system. The tasks are assigned priorities according to the EDF rule and the tasks with higher priorities are admitted to be handled first.
4) *The Main Controller and EDF Scheduler:* The main controller integrates the PID, TB, and TA controllers to form a closed feedback loop and the EDF scheduler decides the schedule of the admitted tasks and their replicas.

VI. Simulation

This section evaluates the efficacy of the proposed scheme through simulation experiments. The performance of the proposed ANN-based system SER and temperature estimation approach is verified first, followed by the validation of the proposed feedback control approach for ICPS.

A. Experimental Settings

We generate 1000 samples for the tasks in the task set Γ. The period (P_i) and deadline (D_i) of tasks in Γ are randomly generated in the interval of (0.25, 1.5) s, and (0.05, 0.1) s, respectively. The AVF of component SRAM, latch, and the logic gate is set to 0.1, 0.15, and 0.08, respectively [41]. The ratios of the numbers of these components are set to 0.775, 0.025, and 0.2, respectively [42]. The critical charge (Q_c) of the above components is generated in the interval of [1.94, 120.10] fC, [1.90, 39.05] fC, and [3.33, 214.4] fC, respectively [34]. Neutron flux F evenly distributes in (0, 550] $\text{cm}^{-2}\text{s}^{-1}$.

The effective switching capacitance λ is set to 15. The active factor of the task (μ_i) is generated in the interval [0.4, 1] according to uniform distribution [24]. The chip die's ambient temperature (T_{amb}) is randomly generated in the interval of [25, 35] °C and the initial temperature (T_{ini}) is assumed to be equal to the ambient temperature. The thermal resistance ($\Re$) for GPU and CPU is set to 1.85 °C/W and 1.83 °C/W, respectively. The thermal capacitance (ϕ) for GPU and CPU is set to 0.12 J/°C and 0.084 J/°C, respectively.

In the validation of the proposed scheme, we consider six scenarios of MPSoCs and workloads: (1GPU, 8CPU, 20tasks), (1GPU, 16CPU, 20tasks), (1GPU, 8CPU, 40tasks), (1GPU, 16CPU, 40tasks), (1GPU, 8CPU, 60tasks), and (1GPU, 16CPU, 20tasks). The supply voltages of CPUs and GPU vary in the interval [0.7, 1.2] V and [0.8, 1.3] V, respectively, both with a step size of 0.1 V. The corresponding frequencies of CPUs and GPU change in the interval [1.0, 2.25] GHz and [1.3, 2.32] GHz, respectively. We use the same PID controller parameters as in [44]. The proposed scheme is compared to a baseline method no-backup (NBK) and two state-of-the-art approaches shared recovery (SR) [43] and evolutionary algorithm (EA) [20]. The main idea of these comparative algorithms is described as follows.

1) NBK is a method that does not employ any extra fault-tolerant techniques when soft errors occur. Thus, it is deemed as a baseline method.
2) SR [43] is a method that exploits the slack time to enhance system reliability. In this method, all tasks are assumed to share a common recovery.
3) EA [20] is a metaheuristic optimization algorithm that uses biological evolution-inspired operations to search for the optimal solution.

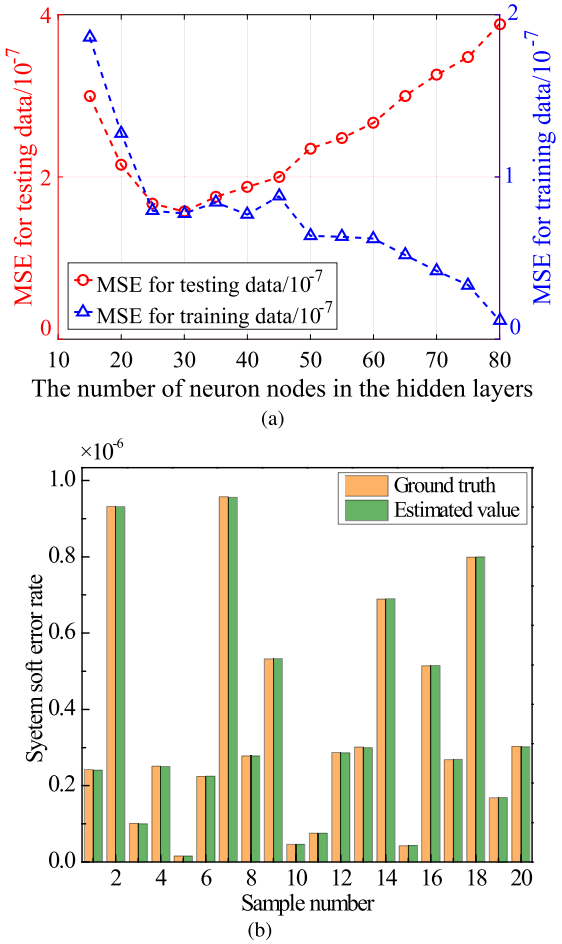

Fig. 4. Performance of the SER_{sys} estimation based on the ANN. (a) MSE with different numbers of hidden layer neuron nodes. (b) Comparison of estimated value and ground truth.

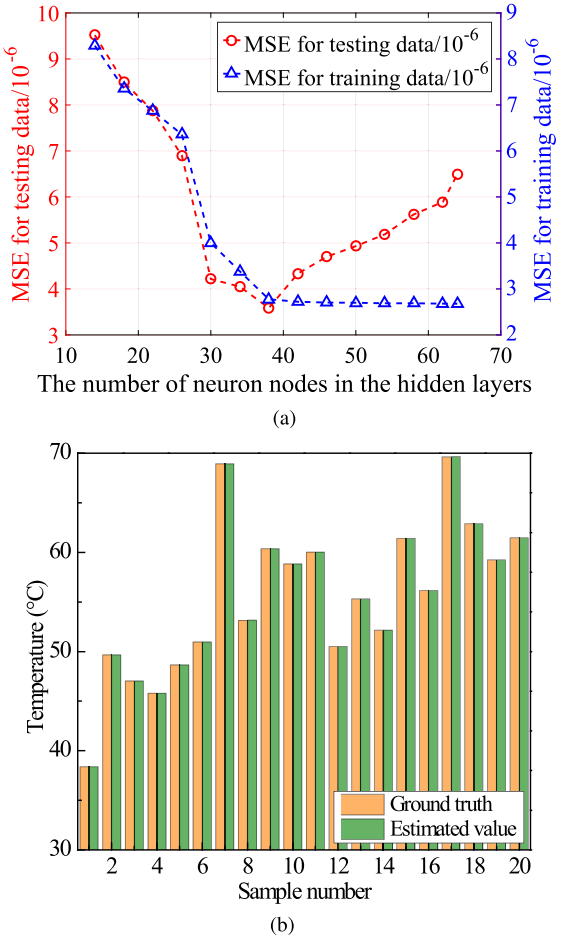

Fig. 5. Performance of the temperature estimation based on the ANN. (a) MSE with a different number of hidden layer neuron nodes. (b) Comparison of estimated value and ground truth.

B. Results on ANN-Based SER and Temperature Prediction

The number of hidden layer neuron nodes significantly impacts the ANN performance. Too few hidden layer nodes may lead to underfitting, while too many neuron nodes may increase the possibility of overfitting. Therefore, it is crucial to determine an appropriate number of hidden layer neuron nodes. For our application of interest, the number of hidden layer neuron nodes varies from 15 to 80 according to (20). The relationship between the number of the neuron nodes in the hidden layer and MSE in training and testing data is shown in Fig. 4(a). It can be seen that for the training data, as the number of hidden layer neuron nodes increases, MSE for the training data decreases. However, the figure also demonstrates that MSE for the testing data increases when the number of neuron nodes exceeds 30, and the difference between the MSE for the training and testing data increases as the number of neuron nodes in the hidden layer increases. In other words, overfitting appears when the number of neuron nodes in the hidden layer exceeds 30. Thus, we set the number of hidden layer neuron nodes of the ANN model to 30.

We obtain 1000 examples by using the SPICE simulation and adopt 800 of them to train an ANN. The other 200 examples are used to test the performance of the obtained ANN model. For the training process, the maximum difference between the ground truth and the estimated system SER from the trained ANN is 2.51%, and the minimum difference is 0%. The maximum and minimum differences are 2.88% and 0%, respectively, for the testing process. The test results clearly indicate that the estimated system SER derived by our ANN is very close to the ground-truth data. Due to the page limit, Fig. 4(b) only gives the comparison between the predicted system SER from the ANN model and the ground truth from the SPICE simulation of 20 samples.

In the architecture of ANN for the temperature prediction, the number of neuron nodes in the input and output layer is five and one, respectively. The number of hidden layer neuron nodes varies from 14 to 66 according to (20). Fig. 5(a) shows the relationship between the number of neuron nodes in the hidden layer and MSE in training and testing data. It can be seen that for the testing data, the value of MSE increases when the number of neuron nodes exceeds 38, and the difference between the MSEs for the training and testing data increases as the number neuron nodes in the hidden layer increases. To avoid overfitting, the number of hidden layer neuron nodes in the ANN for the temperature prediction is set to 38.

We obtain 1000 examples by using HotSpot simulation and adopt 800 of them to train an ANN. The other 200 examples

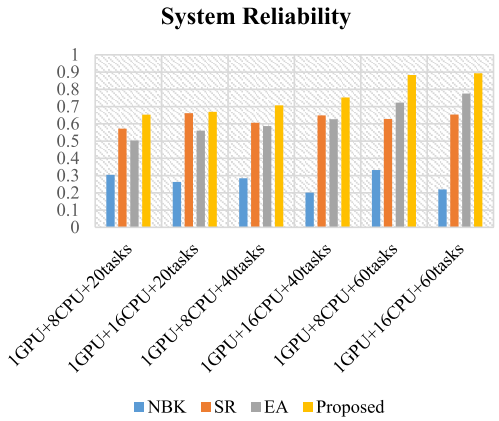

Fig. 6. System reliability achieved by the proposed feedback control scheme with that of baseline method NBK and two state-of-the-art approaches SR [43] and EA [20] under varying MPSoCs and workloads.

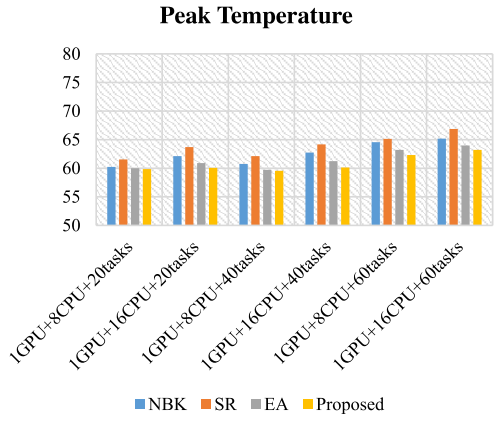

Fig. 7. Peak temperature of the proposed feedback control scheme with that of baseline method NBK and two state-of-the-art approaches SR [43] and EA [20] under varying MPSoCs and workloads.

are used to test the accuracy of the trained ANN. For the training process, the maximum difference between the ground truth and the estimated system SER from the trained ANN is 1.47%, and the minimum difference is 0%. Also, for the testing data, the maximum and minimum difference is 1.79%, and 0%, respectively. Thus, we can conclude that the estimated temperature derived by our ANN is also very close to the ground truth. Due to the page limit, Fig. 5(b) only shows the comparison between the estimated system SER from the trained ANN and the ground truth from the HotSpot of 20 examples.

C. Results on Reliability and Temperature

Fig. 6 compares the system reliability achieved by the proposed feedback control scheme with that of baseline method NBK and two state-of-the-art approaches SR [43] and EA [20] under varying MPSoCs and workloads. The results clearly show that the proposed scheme can always achieve the highest system reliability regardless of MPSoCs and workloads. Specifically, over the six scenarios of MPSoCs and workloads used in the experiments, the average system reliability achieved by NBK, SR, EA, and the proposed schemes are 0.2674, 0.6287, 0.6294, and 0.7599, respectively. The reliability improvements achieved by the proposed scheme over NBK, SR, and EA are 184.2%, 20.87%, and 20.73%. The reliability improvement is due to the effective solution generated by the proposed feedback control scheme.

Fig. 7 compares the peak temperature of the proposed feedback control scheme with that of the baseline method NBK and two state-of-the-art approaches SR [43] and EA [20] under varying MPSoCs and workloads. From the results in the figure, we can easily find that the proposed scheme can always have the lowest peak temperature regardless of MPSoCs and workloads. Specifically, over the six scenarios of MPSoCs and workloads used in the experiments, the average system reliabilities achieved by NBK, SR, EA, and the proposed scheme are 62.597 °C, 63.925 °C, 61.51 °C, and 60.85 °C, respectively. The maximal temperature reduction achieved by the proposed scheme is 2.075°C. Thus, the proposed scheme is much easier to meet the temperature constraint. The temperature reduction benefits from the effectiveness of the proposed feedback control scheme in controlling the temperature of tasks.

VII. Conclusion

In this article, we aim to solve the problem of maximizing soft-error reliability for ICPS applications running on heterogeneous CPU- and GPU-integrated MPSoC platforms under the peak temperature constraint. For this goal, we propose a feedback control-based task scheduling scheme that determines the number of tasks admitted into the system and the number of replicas for admitted tasks. In addition, considering that using simulation tools to derive system SER and temperature may be time-consuming, we design an ANN-based method that can quickly and accurately derive the SER and temperature data. The extensive experiments show that the estimated SER and temperature derived by our ANN is very close to the ground-truth data and the proposed feedback control scheme is effective in enhancing system reliability with a lower peak temperature compared to three representative methods. For the near future, we plan to extend our scheme to: 1) a more general hardware model that CPU cores could be heterogeneous and 2) a security-critical ICPS that needs to tackle security, privacy, as well as reliability challenges simultaneously.

References

[1] W. Sun, X. Yuan, J. Wang, Q. Li, L. Chen, and D. Mu, "End-to-end data delivery reliability model for estimating and optimizing the link quality of industrial WSNs," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 3, pp. 1127–1137, Jul. 2018.

[2] M. A. Saez, F. P. Maturana, K. Barton, and D. M. Tilbury, "Context-sensitive modeling and analysis of cyber-physical manufacturing systems for anomaly detection and diagnosis," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 1, pp. 29–40, Jan. 2020.

[3] Z. Li, J. Kang, R. Yu, D. Ye, Q. Deng, and Y. Zhang, "Consortium blockchain for secure energy trading in industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 14, no. 8, pp. 3690–3700, Dec. 2017.

[4] Z. Li, W. Wei, T. Zhang, M. Wang, S. Hou, and X. Peng, "Online multi-expert learning for visual tracking," *IEEE Trans. Image Process.*, vol. 29, pp. 934–946, 2020.

[5] X. Chen, J. Wang, and S. X. Ding, "Complex system monitoring based on distributed least squares method," *IEEE Trans. Autom. Sci. Eng.*, early access, Sep. 24, 2020, doi: 10.1109/TASE.2020.3022924.

[6] C. Zhang and P. Jiang, "RFID-driven energy-efficient control approach of CNC machine tools using deep belief networks," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 1, pp. 129–141, Jan. 2020.

[7] J. Zhou, J. Sun, M. Zhang, and Y. Ma, "Dependable scheduling for real-time workflows on cyber-physical cloud systems," *IEEE Trans. Ind. Informat.*, early access, Jul. 23, 2020, doi: 10.1109/TII.2020.3011506.

[8] J. Zhou, T. Wei, M. Chen, J. Yan, X. S. Hu, and Y. Ma, "Thermal-aware task scheduling for energy minimization in heterogeneous real-time MPSoC systems," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 35, no. 8, pp. 1269–1282, Aug. 2016.

[9] J. Zhou, X. S. Hu, Y. Ma, and T. Wei, "Balancing lifetime and soft-error reliability to improve system availability," in *Proc. 21st Asia South Pacific Design Autom. Conf. (ASP-DAC)*, Jan. 2016, pp. 685–690.

[10] T. Wei, P. Mishra, K. Wu, and J. Zhou, "Quasi-static fault-tolerant scheduling schemes for energy-efficient hard real-time systems," *J. Syst. Softw.*, vol. 85, no. 6, pp. 1386–1399, Jun. 2012.

[11] W. Jiang, X. Pan, K. Jiang, L. Wen, and Q. Dong, "Energy-aware design of stochastic applications with statistical deadline and reliability guarantees," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 38, no. 8, pp. 1413–1426, Aug. 2019.

[12] W. Jiang, Z. He, J. Zhan, and W. Pan, "Attack-aware detection and defense to resist adversarial examples," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, early access, Oct. 26, 2020, doi: 10.1109/TCAD.2020.3033746.

[13] A. Naithani, S. Eyerman, and L. Eeckhout, "Optimizing soft error reliability through scheduling on heterogeneous multicore processors," *IEEE Trans. Comput.*, vol. 67, no. 6, pp. 830–846, Jun. 2018.

[14] K.-H. Chen, G. V. der Bruggen, and J.-J. Chen, "Reliability optimization on multi-core systems with multi-tasking and redundant multi-threading," *IEEE Trans. Comput.*, vol. 67, no. 4, pp. 484–497, Apr. 2018.

[15] N. Chatterjee, S. Paul, and S. Chattopadhyay, "Task mapping and scheduling for network-on-chip based multi-core platform with transient faults," *J. Syst. Archit.*, vol. 83, pp. 34–56, Feb. 2018.

[16] T. Li, M. Shafique, S. Rehman, J. A. Ambrose, J. Henkel, and S. Parameswaran, "DHASER: Dynamic heterogeneous adaptation for soft-error resiliency in ASIP-based multi-core systems," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Nov. 2013, pp. 646–653.

[17] K.-H. Chen, J.-J. Chen, F. Kriebel, S. Rehman, M. Shafique, and J. Henkel, "Task mapping for redundant multithreading in multi-cores with reliability and performance heterogeneity," *IEEE Trans. Comput.*, vol. 65, no. 11, pp. 3441–3455, Nov. 2016.

[18] J. Zhou, X. S. Hu, Y. Ma, J. Sun, T. Wei, and S. Hu, "Improving availability of multicore real-time systems suffering both permanent and transient faults," *IEEE Trans. Comput.*, vol. 68, no. 12, pp. 1785–1801, Dec. 2019.

[19] Y. Ma, J. Zhou, T. Chantem, R. P. Dick, S. Wang, and X. S. Hu, "Online resource management for improving reliability of real-time systems on 'big–little' type MPSoCs," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 39, no. 1, pp. 88–100, Jan. 2020.

[20] J. Zhou *et al.*, "Resource management for improving soft-error and lifetime reliability of real-time MPSoCs," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 38, no. 12, pp. 2215–2228, Dec. 2019.

[21] J. Cui and D. L. Maskell, "Dynamic thermal-aware scheduling on chip multiprocessor for soft real-time system," in *Proc. 19th ACM Great Lakes Symp. VLSI (GLSVLSI)*, 2009, p. 393.

[22] B. Yun, K. G. Shin, and S. Wang, "Thermal-aware scheduling of critical applications using job migration and power-gating on multi-core chips," in *Proc. IEEE 10th Int. Conf. Trust, Secur. Privacy Comput. Commun.*, Nov. 2011, pp. 1083–1090.

[23] V. Hanumaiah, S. Vrudhula, and K. S. Chatha, "Performance optimal online DVFS and task migration techniques for thermally constrained multi-core processors," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 30, no. 11, pp. 1677–1690, Nov. 2011.

[24] J. Zhou, J. Yan, J. Chen, and T. Wei, "Peak temperature minimization via task allocation and splitting for heterogeneous MPSoC real-time systems," *J. Signal Process. Syst.*, vol. 84, no. 1, pp. 111–121, Jul. 2016.

[25] H. F. Sheikh, I. Ahmad, and D. Fan, "An evolutionary technique for performance-energy-temperature optimized scheduling of parallel tasks on multi-core processors," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 3, pp. 668–681, Mar. 2016.

[26] X. Zhou, J. Yang, Y. Xu, Y. Zhang, and J. Zhao, "Thermal-aware task scheduling for 3D multicore processors," *IEEE Trans. Parallel Distrib. Syst.*, vol. 21, no. 1, pp. 60–71, Jan. 2010.

[27] K. Cao, J. Zhou, T. Wei, M. Chen, S. Hu, and K. Li, "A survey of optimization techniques for thermal-aware 3D processors," *J. Syst. Archit.*, vol. 97, pp. 397–415, Aug. 2019.

[28] J. Zhou and T. Wei, "Stochastic thermal-aware real-time task scheduling with considerations of soft errors," *J. Syst. Softw.*, vol. 102, pp. 123–133, Apr. 2015.

[29] J. Zhou *et al.*, "Reliability and temperature constrained task scheduling for makespan minimization on heterogeneous multi-core platforms," *J. Syst. Softw.*, vol. 133, pp. 1–16, Nov. 2017.

[30] Y. Ma, J. Zhou, T. Chantem, R. P. Dick, S. Wang, and X. S. Hu, "Improving reliability of soft real-time embedded systems on integrated CPU and GPU platforms," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 39, no. 10, pp. 2218–2229, Oct. 2020.

[31] L. Li, J. Zhou, T. Wei, M. Chen, and X. S. Hu, "Learning-based modeling and optimization for real-time system availability," *IEEE Trans. Comput.*, early access, Apr. 29, 2020, doi: 10.1109/TC.2020.2991177.

[32] P. Hazucha and C. Svensson, "Impact of CMOS technology scaling on the atmospheric neutron soft error rate," *IEEE Trans. Nucl. Sci.*, vol. 47, no. 6, pp. 2586–2594, 2000.

[33] M. Ebrahimi and M. B. Tahoori, "Invited–cross-layer approaches for soft error modeling and mitigation," in *Proc. 53rd Annu. Design Autom. Conf.*, Jun. 2016, pp. 1–6.

[34] M. Riera, R. Canal, J. Abella, and A. Gonzalez, "A detailed methodology to compute soft error rates in advanced technologies," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, 2016, pp. 217–222.

[35] S. S. Mukherjee, C. Weaver, J. Emer, S. K. Reinhardt, and T. Austin, "A systematic methodology to compute the architectural vulnerability factors for a high-performance microprocessor," in *Proc. 22nd Digit. Avionics Syst. Conf.*, 2003, pp. 29–40.

[36] *Backpropagation*. Accessed: Mar. 3, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Backpropagation

[37] K. Skadron, M. R. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy, and D. Tarjan, "Temperature-aware microarchitecture: Modeling and implementation," *ACM Trans. Archit. Code Optim.*, vol. 1, no. 1, pp. 94–125, 2004.

[38] Hspice. *Synopsys*. Accessed: Mar. 1, 2020. [Online]. Available: https://www.synopsys.com/

[39] A. Das, A. Kumar, B. Veeravalli, C. Bolchini, and A. Miele, "Combined DVFS and mapping exploration for lifetime and soft-error susceptibility improvement in MPSoCs," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, 2014, pp. 1–6.

[40] C. Lu, J. A. Stankovic, G. Tao, and S. H. Son, "Design and evaluation of a feedback control EDF scheduling algorithm," in *Proc. 20th IEEE Real-Time Syst. Symp.*, Dec. 1999, pp. 56–67.

[41] M. Ebrahimi, L. Chen, H. Asadi, and M. B. Tahoori, "CLASS: Combined logic and architectural soft error sensitivity analysis," in *Proc. 18th Asia South Pacific Design Autom. Conf. (ASP-DAC)*, Jan. 2013, pp. 601–607.

[42] J. Srinivasan, S. V. Adve, P. Bose, and J. A. Rivers, "Exploiting structural duplication for lifetime reliability enhancement," in *Proc. 32nd Int. Symp. Comput. Archit. (ISCA)*, 2005, pp. 520–531.

[43] B. Zhao, H. Aydin, and D. Zhu, "Shared recovery for energy efficiency and reliability enhancements in real-time applications with precedence constraints," *ACM Trans. Design Autom. Electron. Syst.*, vol. 18, no. 2, pp. 1–21, Mar. 2013.

[44] L. Li *et al.*, "Feedback control of real-time EtherCAT networks for reliability enhancement in CPS," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2018, pp. 688–693.

Junlong Zhou (Member, IEEE) received the Ph.D. degree in computer science and technology from East China Normal University, Shanghai, China, in 2017.

He was a Visiting Scholar with the University of Notre Dame, Notre Dame, IN, USA, from 2014 to 2015. He is currently an Associate Professor with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China. His research interests include embedded systems, cloud-edge-IoT, and cyber-physical systems, where he has published two book chapters and 80 refereed articles, including 25+ in premier IEEE TRANSACTIONS.

Dr. Zhou received the Best Paper Award from IEEE iThings 2020. He has held chair positions in many IEEE/ACM conferences. He has been an Associate Editor of the *Journal of Circuits, Systems, and Computers* and the *IET Cyber-Physical Systems: Theory & Applications*, a Subject Area Editor of the *Journal of Systems Architecture*, and a Guest Editor for six international journals, such as *ACM Transactions on Cyber-Physical Systems*.

Liying Li received the B.S. degree from the Department of Computer Science and Technology, East China Normal University, Shanghai, China, in 2017, where she is currently pursuing the Ph.D. degree.

Her current research interests are in the areas of cyber-physical systems, IoT resource management, and distributed artificial intelligence.

Ahmadreza Vajdi received the Ph.D. degree in software engineering from the Nanjing University of Science and Technology, Nanjing, China, in 2019.

He is currently Post-Doctoral Researcher with the School of Computer Science and Engineering, Nanjing University of Science and Technology. His research interests include wireless sensor networks, cyber-physical systems, cloud-edge computing, and the Internet-of-Things.

Xiumin Zhou received the B.S. degree in computer science and engineering from the Nanjing University of Science and Technology, Nanjing, China, in 2012, where he is currently pursuing the Ph.D. degree. His current research interests are in the areas of embedded systems, cloud computing, and distributed systems.

Zebin Wu (Senior Member, IEEE) received the B.Sc. and Ph.D. degrees in computer science and technology from the Nanjing University of Science and Technology, Nanjing, China, in 2003 and 2007, respectively.

He is currently a Professor with the School of Computer Science and Engineering, Nanjing University of Science and Technology. He was a Visiting Scholar with the GIPSA-Laboratory, Grenoble INP, University Grenoble Alpes, Grenoble, France, from August 2018 to September 2018. He was a Visiting Scholar with the Department of Mathematics, University of California at Los Angeles, Los Angeles, CA, USA, from August 2016 to September 2016, and from July 2017 to August 2017. He was a Visiting Scholar with the Hyperspectral Computing Laboratory, Department of Technology of Computers and Communications, Escuela Politecnica, University of Extremadura, Caceres, Spain, from June 2014 to June 2015. His research interests include hyperspectral image processing, parallel and cloud computing, and big data processing.