Improving Reliability and Sustainability of Hazard-Aware Cyber-Physical Systems

Peijin Cong, Junlong Zhou, *Member, IEEE*, Weiming Jiang, Mingsong Chen, *Senior Member, IEEE*, Shiyan Hu, *Senior Member, IEEE*, and Keqin Li, *Fellow, IEEE*

Abstract—The network system deployed in hazardous environments is a key component of hazard-aware cyber-physical systems (CPSs) and its performance highly depends on surrounding environments. Due to the mobility of network nodes (e.g., portable IoT devices), frequently changeable network topology and links, as well as other external interferences such as electromagnetic interference, ensuring adaptivity and reliability of hazard-aware CPSs is of utmost importance. Meanwhile, the timeliness of message transmission is stringent in hazardous environments because the violation of timing requirements may lead to serious consequences. Last but not least, portable IoT devices are typically energy limited, thus ensuring a sustainable message transmission is highly necessary. In this paper, we aim at optimizing the reliability of hazard-aware CPSs while meeting the timing and energy constraints. To this end, we develop the first hazard-aware CPS model and study the impacts of surrounding environments (i.e., physical side) to the network infrastructure of a hazard-aware CPS (i.e., cyber side) with respect to reliability. We also propose a new scheme that adaptively tunes the fault tolerance strategies and admission strategies for real-time messages, to increase the reliability of hazard-aware CPSs under the energy constraint. Extensive simulation results demonstrate that our proposed scheme is capable of increasing system reliability by up to 4.21× with a lower deadline miss rate and runtime overhead compared with the state-of-the-art approaches.

Index Terms—Cyber-physical systems (CPSs), hazard-aware CPSs, reliability, real-time messages, sustainability

- *Peijin Cong and Weiming Jiang are with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, Jiangsu 210094, China. E-mail: {pjcong, jwm}@njust.edu.cn.*
- *Junlong Zhou is with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, Jiangsu 210094, China, and also with the State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100045, China. E-mail: jlzhou@njust.edu.cn.*
- *Mingsong Chen is with the Ministry of Education Engineering Research Center of Software/Hardware Co-design Technology and Application, East China Normal University, Shanghai 200062, China. E-mail: mschen@sei.ecnu.edu.cn.*
- *Shiyan Hu is with the School of Electronics and Computer Science, University of Southampton, SO17 1BJ Southampton, U.K. E-mail: S.Hu@soton.ac.uk.*
- *Keqin Li is with the Department of Computer Science, State University of New York, New York, NY 12561 USA. E-mail: lik@newpaltz.edu.*

Manuscript received 21 October 2022; revised 6 December 2022; accepted 8 December 2022. Date of publication 15 December 2022; date of current version 5 June 2024.
This work was supported in part by the National Natural Science Foundation of China under Grants 62172224 and 62272170, in part by the Natural Science Foundation of Jiangsu Province under Grant BK20220138, in part by the China Postdoctoral Science Foundation under Grants BX2021128, 2021T140327, and 2020M680068, in part by the Postdoctoral Science Foundation of Jiangsu Province under Grant 2021K066A, in part by the Fundamental Research Funds for the Central Universities under Grants 30922010318 and 30922010406, in part by the Open Research Fund of the State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences under Grant CARCHA202105, in part by the Future Network Scientific Research Fund Project under Grant FNSRFP-2021-YB-6, and in part by the Open Research Fund of Engineering Research Center of Software/Hardware Co-Design Technology and Application, Ministry of Education (East China Normal University) under Grant OP202203. Recommended for acceptance by Z. Xu, S. Ren, and O. Rana.
(Corresponding author: Junlong Zhou.)
Digital Object Identifier no. 10.1109/TSUSC.2022.3229310

1 Introduction

The hazardous environments include locations where natural disasters (e.g., earthquakes, floods, hurricanes) or man-made hazards (e.g., wars, terrorist attacks, nuclear and radiation accidents) may exist, as well as special places such as underground spaces, research laboratory or test centers that would be dangerous for humans to work in. In such environments, most of disasters and accidents are usually unpredictable, and their occurrences could induce massive destruction of infrastructures and loss of human lives [1]. Once disasters or accidents happen, the top priority is to save the survivors through providing emergency aid. However, during disasters or accidents, the telecom infrastructure may very often be totally or partially destroyed, increasing the difficulty in ensuring connectivity among the rescue teams. As a result, quickly deploying an emergency network infrastructure in the affected area to restore connectivity and provide communication services for rescue workers is crucial to post-disaster/accident management [2].

The network system deployed in hazardous environments is a key component of hazard-aware cyber-physical systems (CPSs) and its performance highly depends on surrounding environments. Many challenges exist in designing such a hazard-aware CPS [2], [3], [4]. First, the network must be adaptive and scalable to cope with unknown dynamical environments in which the network nodes (e.g., portable IoT devices used by humans) are mobile as well as the network topology and links are frequently changing. Second, message transmission in hazardous environments is more vulnerable to transient faults and hence more likely to suffer a soft error, which may lead to serious consequences. Third, the timeliness of the network system is

particularly important to real-time message transmission since the delay may result in unexpected results. Last but not least, portable IoT devices are typically battery-powered or energy-harvested, which necessitates an energy-efficient message transmission for ensuring the sustainability of hazard-aware CPSs. Therefore, this paper aims to tackle the above challenges through developing an adaptive, reliable, and sustainable hazard-aware CPS for real-time message transmission.

Facing these challenges, a great number of studies [1], [5], [6], [7] have been conducted to build critical network infrastructure in hazardous environments. However, these studies ignore the interactions between surrounding environments and networks. The CPS approaches [8], [9], [10], [11], [12], [13] could be applied to cope with the interactions. But they are not specially designed for hazardous environments and do not consider the reliability of network systems. In this paper, we focus on maximizing the reliability of CPS network infrastructure considering hazardous environments and energy harvesting under energy, utilization, and real-time constraints. To formulate the reliability maximization problem, we develop a hazard-aware CPS model that investigates the impacts of surrounding environments on the CPS network infrastructure from the reliability perspective. Based on this model, we present a new approach that determines the replication and admission of real-time messages for increasing the reliability of hazard-aware CPSs. Our contributions are summarized as follows.

- We develop a hazard-aware CPS model that specially considers the impacts of hazardous environments on the network infrastructure. The hazard-aware CPS uses the ring topology and consists of a master and multiple slaves. The master and slaves are all considered as computing nodes and they are connected by cables and powered by renewable energy.
- We give the first formulation in the literature for the energy and deadline-constrained reliability optimization problem in hazard-aware CPSs. Since the messages transmitted in hazard-aware CPSs can suffer transient faults at both network nodes and links, we consider the reliability for both in the problem formulation. To solve this problem, we propose a reliability enhancement framework that is designed based on the proportional-integral-derivative (PID) feedback control scheme and uses four controllers to adaptively adjust the number of messages and replicas for accommodating varying environments.
- We perform extensive simulation experiments to verify the proposed reliability enhancement framework. The experimental results of three message sets running on two simulated network systems reveal that our framework can increase system reliability by up to 4.21× when compared with state-of-the-art approaches regardless of the scale of network systems and message sets.

The rest of this paper is organized as follows. Section 2 reviews related work and Section 3 introduces the preliminaries used in this paper. Section 4 presents the details of the proposed methodology for reliability modeling and optimization of hazard-aware CPSs. Section 5 describes the simulation experiments and analyzes the results. A brief summary of this work and a short discussion on future work are given in Section 6.

2 Related Work

Considerable research efforts have been directed toward establishing critical network infrastructure in hazardous environments. Li et al. [5] attempted to solve the emergency communication problem in the hazardous scenario and they concentrated on the problem of maximizing the data traffic throughput of wireless mesh networks under limited energy and link capacity constraints and formulated it as a mixed integer linear programming problem. Zhang et al. [1] investigated the network performance gains derived through deploying two cooperative unmanned aerial vehicles as base stations to serve those vehicles for disaster response. Device-to-device communication is a promising backup solution for a network failure or disaster since in such a network, mobile devices can connect to each other directly without using a base station to tackle and route the traffic. Hourani et al. [6] quantified the disaster alleviation benefits of device-to-device communication. In [7], the authors developed a vehicle-assisted resilient network system with the objective of completing as many sensing, data collection, and message dissemination tasks as possible for post-disaster management. Although the above approaches are useful to help establish network infrastructure in hazardous environments, they do not consider the interactions between surrounding environments and networks.

On the other hand, there are numerous CPS methodologies that have been successfully deployed to handle the interactions. For example, Lima et al. [8] introduced a new approach to defend against man-in-the-middle attacks in the sensors and the network channels of CPSs. Gai et al. [9] focused on resource management in CPSs to minimize task execution costs by assigning computing resources to heterogeneous clouds. Kawamoto et al. [10] proposed a mobility-aware CPS method to enhance connectivity and reduce the power consumption of mobile ad-hoc networks. A perception-oriented scheme is presented in [11] to quantify the trustworthiness of networked CPSs from the aspects of ability, benevolence, and integrity. Zhou et al. [12] designed a decomposition-based task scheduling algorithm for addressing the security-energy trade-off optimization problem in mobile CPSs. Burg et al. [13] summarized the properties and requirements of wireless communication from CPSs' connectivity perspective and reviewed the state-of-the-art approaches toward designing secure CPSs. However, these CPS approaches are not specially designed for hazardous environments and they in general neglect the reliability of network systems. There exists a latest work [14] that develops a framework to conduct an automated hazard analysis for CPS security and safety. Again, CPS reliability is not taken into account.

3 Preliminaries

This paper focuses on optimizing the reliability of an energy-harvesting hazard-aware CPS suffering transient

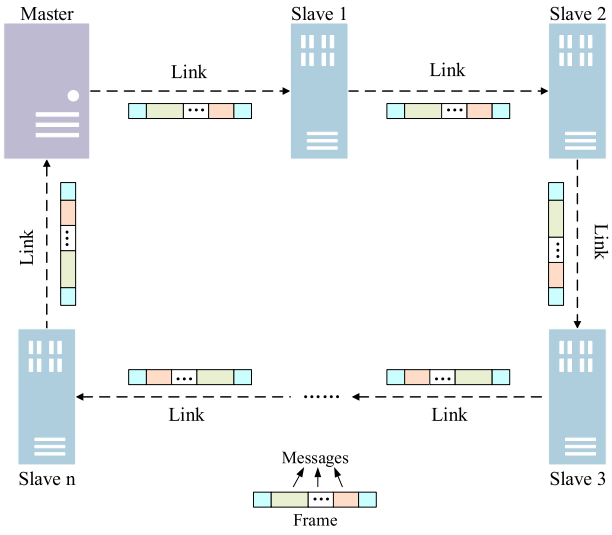


Fig. 1. An illustration of the network topology.

faults. We introduce the preliminaries including the models of network infrastructure in CPS, energy supply, energy consumption, and the problem definition as follows. Our modeling of a hazard-aware CPS's reliability is presented in Section 4.

3.1 Network System

Consider a network in CPS that adopts the ring topology and consists of one master node and n slave nodes connected by the cable, as shown in Fig. 1. During the communication process, the master cyclically sends a frame to all slaves, which is formed with multiple sub-telegrams (or called messages) in a standard structure. Concretely, each slave distinguishes the message sent to itself, completes an operation (read, process and/or write data) specified by command parameters without buffering the frame, and finally forwards other messages in the frame. Once the last slave sends the frame back to the master, the next transmission cycle starts again. For simplicity, we do not consider the non-IT components of the master and slaves. Thus, the master and slaves in the network are all referred to as computing nodes, represented by $\{C_1, C_2, \ldots, C_{n+1}\}$. The $n+1$ computing nodes are heterogeneous since their processing elements demonstrate varying computing capability. In other words, the time of computing nodes processing a message in unit length, denoted by $\{T_1, T_2, \ldots, T_{n+1}\}$, are different.

The independent messages transmitted in the network can be represented by a set $\mathcal{M} = \{\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_g\}$. Each message $\mathcal{M}_i$ $(1 \leq i \leq g)$ is modeled as a triple (P_i, D_i, L_i) where P_i, D_i, and L_i represent the period, relative deadline, and length of message $\mathcal{M}_i$. Let $T(\mathcal{M}_i, C_k)$ be the processing time of message $\mathcal{M}_i$ on computing node C_k $(1 \leq k \leq n+1)$. The message processing time is expressed as

$$T(\mathcal{M}_i, C_k) = L_i \times T_k. \tag{1}$$

To satisfy the constraint of real-time transmission, the message processing time should be less than the relative deadline, i.e., $T(\mathcal{M}_i, C_k) \leq D_i$. Let Υ_{miss} denote the deadline missing rate (DMR) that is derived as the ratio between the number of messages violating their deadline constraints and the total number of messages transmitted in the network.

3.2 Energy Supply

The computing nodes in the CPS all contain an energy source module that automatically scavenges renewable generation (e.g., solar energy), which is then converted into electrical energy. There is also an energy storage module (i.e., battery) and an energy dissipation module that processes messages. Let $Pow_{\text{harv}}(t)$ represent the harvesting power and $E_{\text{harv}}(t_1, t_2)$ represent the energy harvested from the environment in time interval $[t_1, t_2]$, then we have

$$E_{\text{harv}}(t_1, t_2) = \int_{t_1}^{t_2} Pow_{\text{harv}}(t)dt. \tag{2}$$

The energy source and storage modules both can supply the energy to the dissipation module. Let $E_{\text{sup}}(t_1, t_2)$ be the energy available during time interval $[t_1, t_2]$ and $E_{\text{store}}(t_1)$ be the energy stored into the storage module at time instance t_1, then we can derive the energy supply as

$$E_{\text{sup}}(t_1, t_2) = E_{\text{harv}}(t_1, t_2) + E_{\text{store}}(t_1). \tag{3}$$

3.3 Energy Consumption

The energy consumption of the dissipation module for processing messages is the product of power consumption and processing time. Let $E_{\text{cons}}(\mathcal{M}_i, C_k)$ denote the energy consumed by the dissipation module of computing node C_k for processing message $\mathcal{M}_i$, then it is formulated as

$$E_{\text{cons}}(\mathcal{M}_i, C_k) = Pow_{\text{cons}}(\mathcal{M}_i, C_k) \times T(\mathcal{M}_i, C_k), \tag{4}$$

where $Pow_{\text{cons}}(\mathcal{M}_i, C_k)$ is the power consumed by node C_k for processing $\mathcal{M}_i$. The power consumed by a CMOS-based computing node is the sum of static power and dynamic power. Static power is usually deemed as a processor-dependent constant since it is independent of the switching activities of circuits. On the contrary, the dynamic power is not constant. It is highly related to circuits' charging and discharging activities and it can be formulated as a function of processor voltage and frequency. Details on the calculation of power consumption can be found in the literature [15].

3.4 Problem Definition

Consider a scenario that real-time messages transmitted in an energy-harvesting hazard-aware CPS are periodic and independent as well as forward error control and replication techniques are adopted in the network to ensure fault tolerance. Assuming that the network in CPS follows a ring topology containing $n+1$ nodes (one master node and n slave nodes) and the characteristics of g messages to be transmitted are known, our objective is to determine the number of admitted messages in the network system and the number of replicas for each message so that the network system's reliability is maximized under the energy, deadline, and utilization constraints. We formulate the studied problem as

$$\max \quad R_{\text{sys}} \tag{5}$$

$$\text{s.t.} \quad E_{\text{cons}} \leq E_{\text{sup}} \tag{6}$$

$$\Upsilon_{\text{miss}} \leq \epsilon \tag{7}$$

$$U_{\text{net}} \leq 1. \tag{8}$$

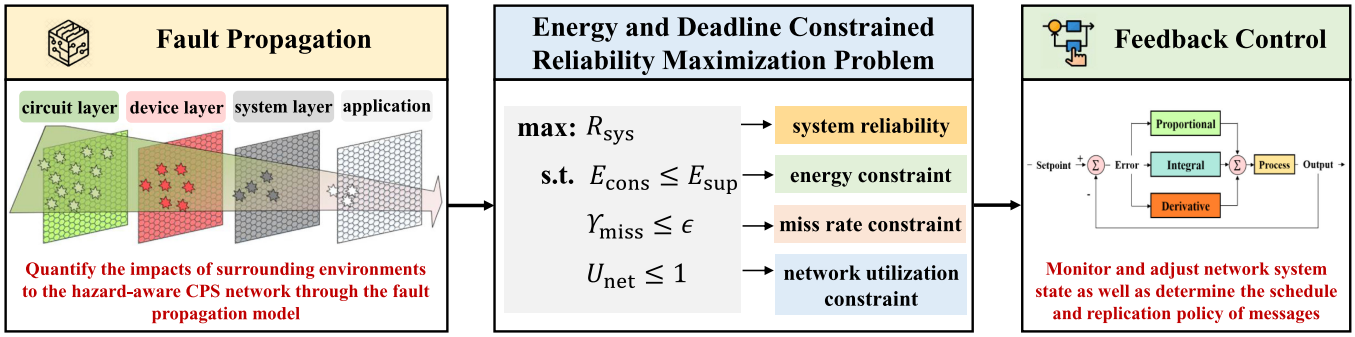


Fig. 2. An overview of our proposed scheme.

R_{sys} given in Eq. (5) is the system's overall reliability that will be detailedly described in the next section. Eqs. (6), (7), and (8) denote the constraints that should be satisfied by the network system. The 1st constraint indicates that system energy consumption E_{cons} cannot exceed supply energy E_{sup}. The 2nd constraint[1] requires the messages' DMR Υ_{miss} during a scheduling period should be less than a pre-defined threshold ϵ. The last constraint ensures that the network utilization U_{net} of processing and transmitting these messages should be lower than 1. The network utilization is determined by the number of messages and replicas transmitted through the communication channel such that it can be also called channel utilization. We use network utilization and channel utilization interchangeably in the following sections.

4 The Proposed Methodology

This paper focuses on solving the reliability optimization problem of energy-harvesting hazard-aware CPSs. Our objective is to design a fault-tolerant message transmission scheme for optimizing the overall reliability of the CPS powered by harvested energy under various constraints. Fig. 2 shows the high-level overview of our proposed scheme for solving this reliability optimization problem. The proposed scheme first quantifies the impacts of surrounding environments on the hazard-aware CPS for estimating system reliability through a fault propagation model. The proposed scheme then maximizes system reliability under the energy, timing, and utilization constraints by using a feedback control approach which decides the number of messages and replicas adaptively. In the following, we describe our modeling of hazard-aware CPSs' reliability and discuss our proposed reliability-aware feedback control scheme in detail.

4.1 Reliability of Hazard-Aware CPSs

Since messages transmitted in the CPS may suffer from faults at both nodes and links, we model the reliability for both considering the impacts of the surrounding environment on the fault rate.

4.1.1 Reliability Model of Computing Nodes

Transient faults may cause errors that appear shortly and then disappear without damaging the device or shortening its lifetime [16]. Thus, the resultant errors are called soft errors that may prevent tasks from completing successfully. Approximately 95% of the particles at the Earth's surface capable of inducing transient faults are energetic neutrons [17]. For an electronic device, neutrons could hit its transistors and hence generate an electrical charge. When the accumulated charge exceeds a threshold, the transistor would be activated, leading to unexpected results (e.g., single-event upsets [18]). The errors produced at the circuit level would propagate from the circuit level to the component level and further to the system level, especially in hazardous environments. The modeling of fault propagation and reliability is described as follows.

Hazucha et al. [19] propose an environment-oriented fault model that especially considers the impacts of the environment (i.e., geographic locations) on fault rate, to estimate the raw fault rate at the circuit level. The raw fault rate is calculated as

$$\phi_{cir} = \alpha \times F \times A \times e^{\vartheta}, \tag{9}$$

where α is a circuit technology-dependent constant, F is the flux of neutrons determined by geographic locations (i.e., altitude, latitude, longitude) of the environment, A is a circuit's area sensitive to faults, and ϑ is a ratio of the circuit's critical charge to the charge collection efficiency.

As reported in [20], the fault rate at the component level can be derived approximately using key components such as SRAM cells, latches, and logic gates. Using the approximate approach [20], the fault rate of a basic component is then calculated as

$$\phi_{com} = \alpha' \times F \times A' \times e^{\vartheta'}, \tag{10}$$

where α' is a constant depending on the component type, F is the flux of neutrons depending on geographic locations, A' is the component's area sensitive to faults, and ϑ' is a ratio of the component's critical charge to the charge collection efficiency. The fault rate at the system level is determined by the fault rates at different components. Let ϕ_{sys} represent the system level fault rate and then it is expressed as

$$\phi_{sys} = \sum_{j=1}^{M} r_j \times V_j \times \phi_{com_j}, \tag{11}$$

where M denotes the number of components in the system, r_j denotes the ratio between the number of components of type j and the total number of components, V_j is the vulnerability (i.e., the probability that a transient fault ultimately results in

1. This is a soft real-time constraint that allows a small portion of messages to miss their deadlines.

a failure at the system level) of components of type j, and $\phi_{\text{com},j}$ denotes the fault rate of components of type j.

An exponential distribution that characterizes the average fault rate is in general utilized to model soft error [15]. Let ϕ be a computing node's average fault rate that captures the expected number of failure occurrences per unit of time. The fault rate of computing node C_k is derived as

$$\begin{aligned}\phi(C_k) &= \phi_k \times e^{(-\varphi_k \times s_k)} \\ &= \phi_k \times e^{(-\varphi_k \times \frac{1}{T_k})},\end{aligned} \tag{12}$$

where ϕ_k is the fault rate of node C_k evaluated by Eq. (11), φ_k is a node dependent constant, and s_k is node C_k's processing speed that is derived as $\frac{1}{T_k}$. The reliability, which is defined as the probability that no faults occur on node C_k when processing message $\mathcal{M}_i$, can be written as

$$R(\mathcal{M}_i, C_k) = e^{(-\phi(C_k) \times T(\mathcal{M}_i, C_k))}, \tag{13}$$

where $\phi(C_k)$ is the fault rate of node C_k and $T(\mathcal{M}_i, C_k)$ is the time of processing message $\mathcal{M}_i$ on node C_k. Given the probability of successfully processing message $\mathcal{M}_i$ on a node, the probability of successfully processing message $\mathcal{M}_i$ on all nodes is derived by

$$R_{\text{node}}(\mathcal{M}_i) = \prod_{k=0}^{n} R(\mathcal{M}_i, C_k). \tag{14}$$

Plugging Eqs. (4) and (12) into Eq. (14), we have

$$\begin{aligned}R_{\text{node}}(\mathcal{M}_i) &= \prod_{k=0}^{n} R(\mathcal{M}_i, C_k) \\ &= \prod_{k=0}^{n} e^{(-\phi(C_k) \times T(\mathcal{M}_i, C_k))} \\ &= \prod_{k=0}^{n} e^{\left(-\phi_k \times e^{(-\varphi_k \times \frac{1}{T_k})} \times L_i \times T_k\right)}.\end{aligned} \tag{15}$$

4.1.2 Reliability Model of Links

To provide fault-tolerance capacity for the CPS, we adopt a forward error control technology [21] that sends and processes the original and replicated messages simultaneously, rather than simply re-sending messages when a failure occurs. During the transmission, messages may suffer bit errors caused by noise and interference. Suppose that the transmission time of message $\mathcal{M}_i$ through all the links is tt_i and the probability of transmitting message $\mathcal{M}_i$ over links successfully is $R_{\text{link}}(\mathcal{M}_i)$. According to the links' bit error model [22], we have

$$R_{\text{link}}(\mathcal{M}_i) = e^{(-\omega \times tt_i)}, \tag{16}$$

where ω is the constant bit error rate.

4.1.3 System Overall Reliability

Although reliability modeling on the nodes and links has been studied before as described above, this work gives the first model on system overall reliability of a hazard-aware CPS. Let $R(\mathcal{M}_i)$ denote the probability of processing and transmitting message $\mathcal{M}_i$ successfully in the CPS network without using replicated messages for fault tolerance, and it is formulated as

$$R(\mathcal{M}_i) = R_{\text{node}}(\mathcal{M}_i) \times R_{\text{link}}(\mathcal{M}_i). \tag{17}$$

A message's reliability is typically defined as the probability that the message is successfully processed by the slaves and transmitted back to the master through the links within the consideration of transient faults and bit errors. Supposing there are ϖ_i replicas for message $\mathcal{M}_i$, the reliability of message $\mathcal{M}_i$ is enhanced and it is expressed as

$$R(\mathcal{M}_i, \varpi_i) = 1 - (1 - R(\mathcal{M}_i))^{\varpi_i + 1}. \tag{18}$$

The system reliability, which depends on the successful processing and transmission of all messages in the network, can be therefore derived as

$$R_{\text{sys}} = \prod_{i=1}^{g} R(\mathcal{M}_i, \varpi_i). \tag{19}$$

Plugging Eqs. (15), (16), (17), and (18) into Eq. (19), we have

$$\begin{aligned}R_{\text{sys}} &= \prod_{i=1}^{g} R(\mathcal{M}_i, \varpi_i) \\ &= \prod_{i=1}^{g} 1 - \Big(1 - R(\mathcal{M}_i)\Big)^{\varpi_i + 1} \\ &= \prod_{i=1}^{g} 1 - \Big(1 - R_{\text{node}}(\mathcal{M}_i) \times R_{\text{link}}(\mathcal{M}_i)\Big)^{\varpi_i + 1} \\ &= \prod_{i=1}^{g} 1 - \Big(1 - \prod_{k=0}^{n} e^{\left(-\phi_k \times e^{(-\varphi_k \times \frac{1}{T_k})} \times L_i \times T_k\right)} \\ &\quad \times e^{(-\omega \times tt_i)}\Big)^{\varpi_i + 1}.\end{aligned} \tag{20}$$

4.2 Feedback Control for Reliability Optimization

As a powerful general technique, control theory is widely adopted in various applications of computing systems and communication networks. In contrast to open-loop control which requires knowing everything accurately to work right, feedback (close-loop) control does not need to know everything and allows estimation and prediction errors [23]. In this sense, feedback control is more suitable for managing systems that deeply interact with uncertainty in environments. Therefore, we attempt to use feedback control to optimize the reliability of hazard-aware CPSs. There are a few feedback control-based schemes [24], [25] proposed for reliability optimization. However, these approaches cannot be directly applied to this work due to the inherent difference in problem natures.

In this article, we design a feedback control technique termed reliability enhancement framework (REF) that incrementally solves the optimization problem in Eqs. (5), (6), (7), and (8) through exploiting the history of system states in the previous profiling window. As illustrated in Fig. 3, REF mainly consists of five components and two queues: a

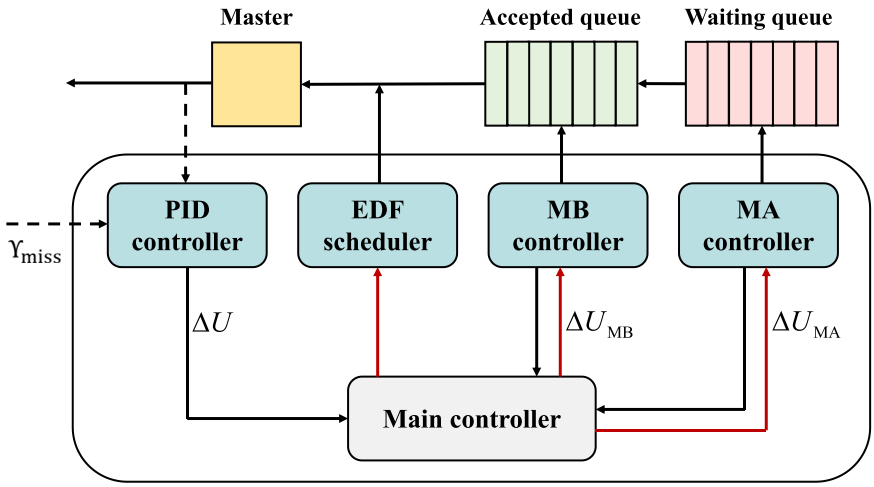


Fig. 3. Reliability enhancement framework REF.

main controller, a proportional-integral-derivative (PID) controller, a message access (MA) controller, a message backup (MB) controller, an earliest deadline first (EDF) scheduler, an accepted (ACC) queue, and a waiting (WAT) queue. These controllers are utilized to monitor and adjust the network system state as well as decide the schedule and replication policy of messages. Specifically, the PID controller periodically captures the messages' DMR Υ_{miss} and sends the channel utilization control action ΔU to the main controller. ΔU is the amount of channel utilization that should be adjusted (i.e., increased when $\Delta U > 0$ or decreased when $\Delta U < 0$) from the system and it depends on the system's current DMR. The accommodation of channel utilization is realized by the MB and MA controllers. Concretely, the MB controller is first utilized to accommodate the channel utilization by adding or reducing the replicas of messages in the ACC queue. If the MB controller is not able to accomplish the required ΔU, the MA controller is then utilized to accommodate the rest of ΔU by controlling the messages allowed to enter the ACC queue. ΔU_{MA} and ΔU_{MB} represent the channel utilization accommodation assigned to MA and MB controllers respectively and hold for $\Delta U_{\text{MA}} + \Delta U_{\text{MB}} = \Delta U$. The EDF scheduler is in charge of scheduling the admitted messages along with their replicas and dispatching the messages to the master for transmission and processing. Since EDF scheduling [27] is a well-known scheduling algorithm, we do not elaborate on the EDF scheduler in the paper. The details on the above-mentioned controllers are presented in the followings.

4.2.1 PID Controller

The PID controller is a popular feedback-based control loop mechanism to enhance the robustness of a control process, which continuously computes the difference between an estimated variable and a measured process variable as well as deploys the procedure based on proportional, integral, and derivative terms to minimize this difference [26]. Following this philosophy, we design our PID control algorithm whose pseudo-code is presented in Algorithm 1.

The input to Algorithm 1 is a pre-defined threshold ϵ used for controlling messages' DMR. The algorithm first samples the messages periodically to derive the DMR Υ_{miss} (line 1). When Υ_{miss} exceeds the threshold ϵ, i.e., $\Upsilon_{\text{miss}} > \epsilon$ (line 2), the algorithm then derives the control variable ΔU (i.e., the channel utilization control action) via the basic PID control formula [27] (line 3) and sends the obtained ΔU to the main controller (line 4). The PID control formula is expressed as

$$\Delta U = -\Theta_P \times \mathcal{E}(t) - \Theta_I \times \sum_{T_{W1}} \mathcal{E}(t) - \Theta_D \times \frac{\mathcal{E}(t) - \mathcal{E}(t - T_{W2})}{T_{W2}}, \tag{21}$$

Algorithm 1. The PID Control Algorithm

Require: The threshold ϵ for controlling DMR Υ_{miss};
Ensure: The channel utilization ΔU to be accommodated;
1: sample messages in the system to obtain DMR Υ_{miss};
2: **while** $\Upsilon_{\text{miss}} > \epsilon$ **do**
3: derive the control variable ΔU using Eq. (21);
4: send the ΔU to the main controller;
5: sample messages in the system to obtain DMR Υ_{miss} when the next sampling window comes;
6: **end while**

where $\mathcal{E}(t) = \epsilon - \Upsilon_{\text{miss}}$ is the difference between the threshold and the current DMR. $\Theta_P, \Theta_I, \Theta_D, T_{W1}, T_{W2}$ are tunable parameters of the PID controller. Specifically, $\Theta_P, \Theta_I, \Theta_D$ are coefficients, T_{W1} is the time window used to sum the errors, and T_{W2} is the time window of derivative errors. After obtaining the control variable ΔU, the algorithm sends ΔU to the main controller which in turn assigns it to the MB and MA controllers for accommodating the channel utilization. Here we use ΔU_{MA} and ΔU_{MB} to represent the channel utilization accommodation assigned to MB and MA controllers respectively, which hold for $\Delta U_{\text{MA}} + \Delta U_{\text{MB}} = \Delta U$. If $\Delta U > 0$, indicating the channel utilization needs to be increased, then more messages and/or replicas are allowed to enter the system. Otherwise, channel utilization needs to be decreased by removing messages and/or replicas from the system. After sending out the ΔU for accommodation, the algorithm samples messages in the system to obtain (new) DMR when the next sampling window comes (line 5). The above process repeats until the DMR is controlled below the threshold.

Finally, the algorithm outputs the channel utilization ΔU to be accommodated by the MB and MA controllers.

4.2.2 MB Controller

The MB controller is used to increase/decrease channel utilization by adding/reducing replicas for messages transmitted through the communication channel. The reliability of a message can be derived by Eq. (18) once the number of the message's replicas is determined. Since the value of each message's reliability is positive, we can derive an inequality that is expressed as

$$\left(\frac{R(\mathcal{M}_1,\varpi_1)+R(\mathcal{M}_2,\varpi_2)+\cdots+R(\mathcal{M}_g,\varpi_g)}{g}\right)^g \geq R(\mathcal{M}_1,\varpi_1)\times R(\mathcal{M}_2,\varpi_2)\times\cdots\times R(\mathcal{M}_g,\varpi_g)=R_{\text{sys}}. \quad (22)$$

The equality holds iff $R(\mathcal{M}_1,\varpi_1)=R(\mathcal{M}_2,\varpi_2)=\cdots=R(\mathcal{M}_g,\varpi_g)$. Clearly, the system reliability R_{sys} is optimized if all the messages' reliabilities are equal and maximal. Based on this observation, we design the MB controller algorithm as shown in Algorithm 2.

Algorithm 2. The MB Control Algorithm

Require: The channel utilization ΔU to be accommodated;
Ensure: The channel utilization accommodation ΔU_{MB} assigned to MB controller and the number of replicas for messages;
1: $\Delta U_{\text{MB}} \Leftarrow 0$;
2: calculate the mean value R_{mean} of messages' reliabilities in the ACC queue;
3: derive the number of messages whose reliability is lower or higher than R_{mean}, represented by n_L and n_H;
4: **if** $\Delta U > 0$ **then**
5: sort the n_L messages whose reliability is lower than R_{mean} in the non-decreasing order of their reliability;
6: $i \Leftarrow 1$;
7: **while** $\Delta U_{\text{MB}} < \Delta U$ **do**
8: **if** adding a replica for message $\mathcal{M}_i$ would not violate the energy constraint given in Eq. (6) **then**
9: add a replica for message $\mathcal{M}_i$;
10: **end if**
11: update ΔU_{MB} by $\Delta U_{\text{MB}} \Leftarrow \Delta U_{\text{MB}} + \Delta U(\mathcal{M}_i,1)$ where $\Delta U(\mathcal{M}_i,1)$ is derived by Eq. (23);
12: $i \Leftarrow i+1$;
13: **if** $i == n_L+1$ **then**
14: $i \Leftarrow 1$;
15: **end if**
16: **end while**
17: **else**
18: sort the n_H messages whose reliability is higher than R_{mean} in the non-decreasing order of their reliability;
19: $i \Leftarrow n_H$;
20: **while** $\Delta U_{\text{MB}} > \Delta U$ **do**
21: remove one backup for message $\mathcal{M}_i$;
22: update ΔU_{MB} by $\Delta U_{\text{MB}} \Leftarrow \Delta U_{\text{MB}} - \Delta U(\mathcal{M}_i,1)$ where $\Delta U(\mathcal{M}_i,1)$ is derived by Eq. (23);
23: $i \Leftarrow i-1$;
24: **if** $i == 0$ **then**
25: $i \Leftarrow n_H$;
26: **end if**
27: **end while**
28: **end if**

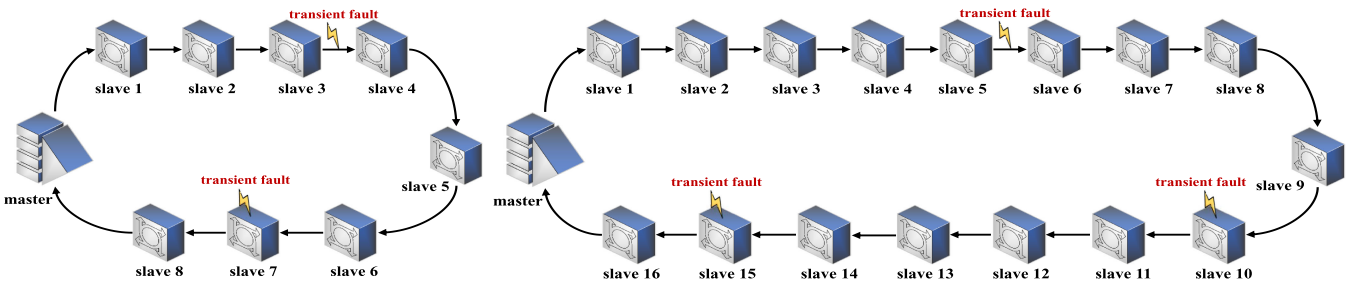
The input to Algorithm 2 is the channel utilization ΔU to be accommodated by the MB and MA controllers. For the MB controller, the algorithm first initializes ΔU_{MB} to 0 (line 1), calculates the mean value R_{mean} of messages' reliabilities in the ACC queue (line 2), and uses n_L and n_H to record the number of messages whose reliability is lower or higher than R_{mean} respectively (line 3). Then depending on the value of ΔU, the algorithm derives the channel utilization ΔU_{MB} accommodated by the MB controller. Specifically, when $\Delta U > 0$, the algorithm sorts the n_L messages whose reliability is lower than R_{mean} in the non-decreasing order of their reliability (line 5) and iteratively adds a replica for messages under the energy constraint (lines 8-12). Once a replica is added, ΔU_{MB} is updated by $\Delta U_{\text{MB}} \Leftarrow \Delta U_{\text{MB}} + \Delta U(\mathcal{M}_i,1)$ where $\Delta U(\mathcal{M}_i,1)$ is the utilization increment by adding a replica. The utilization increment by adding X replicas is

$$\Delta U(\mathcal{M}_i,X)=\frac{\left(\sum_{j=1}^{n+1}T(\mathcal{M}_i,C_k)+\zeta_i\right)(X+1)}{P_i}, \quad (23)$$

where X is the number of replicas added to message $\mathcal{M}_i$, $T(\mathcal{M}_i,C_k)$ is the time of processing message $\mathcal{M}_i$ on node C_k, and ζ_i is the time of transmitting message $\mathcal{M}_i$ over the network. Using Eq. (21), $\Delta U(\mathcal{M}_i,1)$ can be readily derived by setting X to 1. If all the n_L messages have been added with a replica (line 13) and ΔU is not used up yet, i.e., $\Delta U_{\text{MB}} < \Delta U$, the algorithm will go back to the first message (line 14) and repeat the above accommodation process until $\Delta U_{\text{MB}} \geq \Delta U$. The operation process of the other case ($\Delta U \leq 0$, lines 17-28) is similar to that of the case $\Delta U > 0$ (lines 4-16). The difference is that for the n_H messages whose reliability is higher than R_{mean}, the algorithm iteratively reduces messages' replicas for decreasing channel utilization. As a result, Algorithm 2 outputs the channel utilization accommodation ΔU_{MB} to be realized by the MB controller and the updated number of replicas for messages.

Algorithm 3. The MA Control Algorithm

Require: The channel utilization ΔU_{MA} to be accommodated by MA controller and the number N_{ACC} of messages in the ACC queue;
1: **if** $\Delta U_{\text{MA}} > 0$ **then**
2: **while** $\Delta U_{\text{MA}} > 0$ **do**
3: sort messages in the WAT queue following the EDF rule;
4: **for** head in the WAT queue **do**
5: **if** $\Delta U_{\text{MA}} - \Delta U(\mathcal{M}_i,0) > 0$ **then**
6: dequeue the head of the WAT queue and enqueue it to the ACC queue;
7: $N_{\text{ACC}} \Leftarrow N_{\text{ACC}}+1$;
8: $\Delta U_{\text{MA}} \Leftarrow \Delta U_{\text{MA}} - \Delta U(\mathcal{M}_i,0)$;
9: **end if**
10: **end for**
11: **end while**
12: **else**
13: reject newly submitted messages and keep them in the WAT queue;
14: **end if**



(a) Network 1. (b) Network 2.

Fig. 4. The topology of two simulated network systems where transient faults may occur.

4.2.3 MA Controller

If the MB controller cannot afford to realize all the channel utilization accommodation ΔU, then the MA controller is used to help complete the rest of ΔU by controlling the message flow into the network system. The pseudo-code of our MA control algorithm is presented in Algorithm 3. The input to the algorithm is the channel utilization accommodation ΔU_{MA} ($\Delta U_{\mathrm{MA}} \Leftarrow \Delta U - \Delta U_{\mathrm{MB}}$) left to the MA controller and the number of messages in the ACC queue. Given the required channel utilization accommodation ΔU_{MA}, the algorithm first checks the value of ΔU_{MA}. If $\Delta U_{\mathrm{MA}} > 0$, indicating the required channel utilization accommodation can be realized by admitting newly submitted messages (with no replicas) into the ACC queue (line 1). Thus, the algorithm first sorts messages in the WAT queue following the EDF policy, then iteratively dequeues the head message of the WAT queue and enqueues it to the ACC queue if $\Delta U_{\mathrm{MA}} - \Delta U(\mathcal{M}_i, 0) > 0$ holds, and finally updates N_{ACC} and ΔU_{MA} (lines 2-11). If $\Delta U_{\mathrm{MA}} \leq 0$, indicating there is no need to admit new messages into the ACC queue, then the algorithm rejects the newly submitted messages and keeps them in the WAT queue (lines 12-14).

Algorithm 4. The Main Control Algorithm

Require: The messages $\mathcal{M} = \{\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_g\}$ to be processed and transmitted;
1: **for** $\mathcal{M}_i \in \mathcal{M}$ **do**
2: initialize the number of replicas of $\mathcal{M}_i$ under the energy constraint (see Eq. (6)) via the approach [28];
3: **end for**
4: derive the total channel utilization accommodation ΔU using the PID control algorithm (i.e., Algorithm 1);
5: derive the channel utilization accommodation ΔU_{MB} using the MB control algorithm (i.e., Algorithm 2);
6: compute the channel utilization ΔU_{MA} accommodated by the MA controller as $\Delta U - \Delta U_{\mathrm{MB}}$;
7: **if** $\Delta U_{\mathrm{MA}} \neq 0$ **then**
8: call the MA control algorithm (i.e., Algorithm 3) to accommodate the channel utilization ΔU_{MA};
9: **end if**
10: schedule messages using the EDF policy;

4.2.4 Main Controller

The main controller is the core of our proposed reliability enhancement framework REF since it integrates the PID, MB, and MA controllers to construct a closed feedback loop which ensures a robust control process. The details on the main controller are presented in Algorithm 4. The main control algorithm takes the messages $\mathcal{M} = \{\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_g\}$ to be processed and transmitted as input. It first initializes the number of replicas for all messages under the energy constraint (see Eq. (6)) using the approach [28] (lines 1-3). It then computes the total channel utilization accommodation ΔU using the PID control algorithm (line 4) and takes ΔU as input of the MB control algorithm to derive the channel utilization accommodation ΔU_{MB} and the updated number of replicas for messages (line 5). After obtaining ΔU_{MB}, the channel utilization ΔU_{MA} accommodated by the MA controller can be readily obtained as $\Delta U - \Delta U_{\mathrm{MB}}$ (line 6). If $\Delta U_{\mathrm{MA}} \neq 0$, the MA controller needs to accommodate the channel utilization ΔU_{MA} (lines 7-9). Finally, the algorithm schedules the messages using the EDF policy (line 10).

5 Experiments

We carry out numerous simulation-based experiments to verify the proposed reliability enhancement framework REF. This section first introduces the simulation setups and then presents the experimental results and analysis in detail.

5.1 Experimental Setup

To evaluate the proposed framework REF, in the experiments we compare REF with three state-of-the-art approaches NFT, ART, and EA with respect to system reliability and DMR. The three approaches used in the comparison are described as follows.

- NFT (No-Fault-Tolerance) is a baseline method that does not take any extra fault-tolerant techniques when transient faults occur.
- ART (Automatic-Repeat-Transmission) is an automatic repeat transmission mechanism that re-sends messages when transient faults occur.
- EA (Evolutionary Algorithm) is a generic population-based meta-heuristic [29] which utilizes a series of evolution-inspired operations to search for the optimum solution in the solution space.

We implement the simulation experiments on a computer that is equipped with a 3.2GHz Intel i5 quad-core processor and 24GB DDR4 memory. The simulation experiments are written in Java and run on a Mac OS. We use an extensible and modular simulator, OMNeT++ [30], to build two network systems (see Fig. 4) with different ring topologies for validating the proposed framework REF. The first network system

TABLE 1
The Values of PID Controller Parameters

	Θ_P	Θ_I	Θ_D	T_{W1}	T_{W2}
Value	0.5	0.005	0.1	100	1

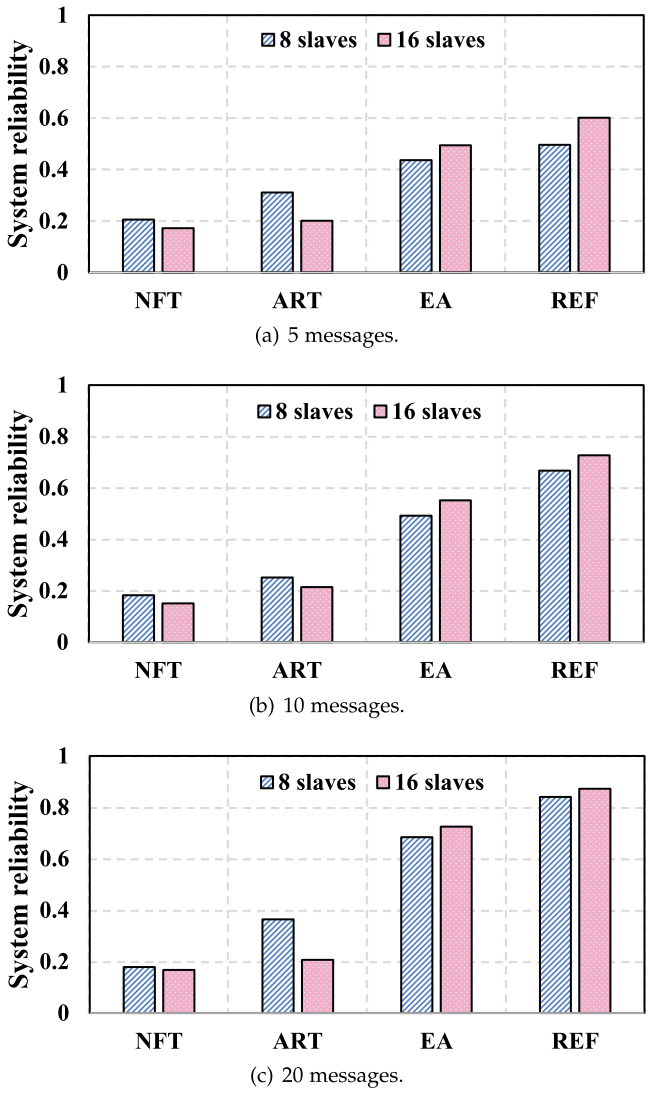
consists of 1 master and 8 slaves while the second network system consists of 1 master and 16 slaves. For the message transmission, three message sets with different scales (containing 5, 10, and 20 messages, respectively) are considered to be handled by the two network systems. During the transmission of messages at nodes and links, transient faults may occur due to the hazardous environment (see Fig. 4). For the environment-related parameters, we use the simulator SPICE to sample the data of neutron flux, supply voltage, and critical charge. For example, the neutron flux F takes the value from the range of $(0, 550]\text{cm}^{-2}\text{s}^{-1}$. We consider three types of components in the experiment. Referring to [32], the vulnerability V_j of three components (i.e., SRAM, latch, and logic gate) are set to 0.1, 0.15, and 0.08, respectively. The ratio of the number of components SRAM, latch, and logic gate to the total number of components are respectively set as 0.775, 0.025, and 0.2. We use the same parameters (i.e., Θ_P, Θ_I, Θ_D, T_{W1}, T_{W2}) [31] of PID controller in the simulation, the values of which are listed in Table 1. The relative deadline (in time units) and the length (in bytes) of messages are derived by a generator, in the range of [0.5,1] and [300,800], respectively. The DMR threshold ϵ varies for different simulation cases but it keeps the same value for different approaches in the same simulation case. Specifically, its value becomes higher along with the increase in the number of messages and slaves in the network system. For example, the value of ϵ is set as 10%, 20%, and 40% for three message sets, respectively. We select solar energy as the renewable generation. Similar to [33], the harvesting power trace is derived using

$$Pow_{\text{harv}}(t) = \left|\kappa(t) \times \cos\left(\frac{t}{70\pi}\right) \times \cos\left(\frac{t}{100\pi}\right)\right|,$$

where $\kappa(t)$ is a normally distributed random variable.

5.2 Experimental Results

Since our objective is to optimize system reliability, we first compare the system reliability achieved by the approaches NFT, ART, EA [29] and our proposed framework REF. As the results are clearly shown in Fig. 5, no matter which topology and message setting is adopted, our proposed framework REF always has the highest system reliability among the four methods. To be specific, when compared to the approaches NFT, ART, EA, the increase in system reliability realized by REF can be up to 2.49× in the case of 5 messages and 16 slaves; the increase in system reliability realized by REF can be up to 3.82× in the case of 10 messages and 16 slaves; the increase in system reliability realized by REF can be up to 4.21× in the case of 20 messages and 16 slaves. In addition, we can observe that the larger the system scale, the higher the reliability improvement can be achieved by our framework REF. The efficacy of our REF in increasing system reliability benefits from its consideration of both node and link failures as well as its reliability-aware feedback controllers.



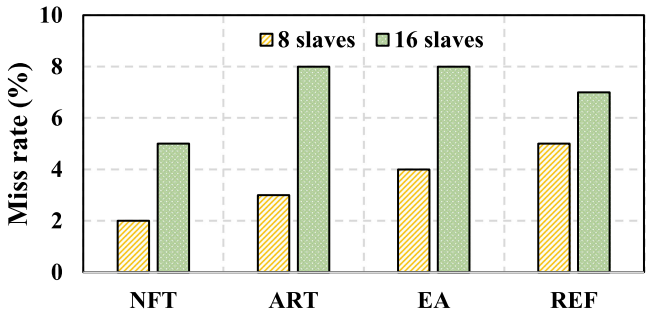
(a) 5 messages.

(b) 10 messages.

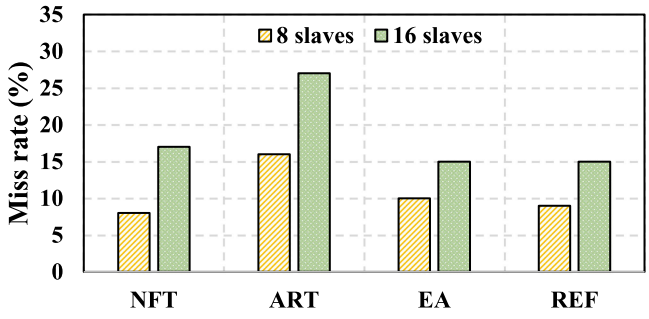
(c) 20 messages.

Fig. 5. The system reliability R_{sys} achieved by NFT, ART, EA [29] and our proposed framework REF in the case of (a) 5 messages (b) 10 messages and (c) 20 messages.

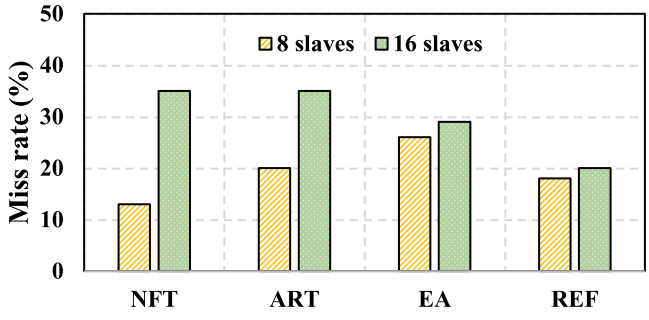
Figs. 6 and 7 compare the DMR Υ_{miss} and channel utilization U_{net} of the approaches NFT, ART, EA [29] and our proposed framework REF, respectively. As can be seen from Fig. 6, for given messages, the DMR of 16-slave systems is higher than that of 8-slave systems. This is due to the increased time used for transmitting messages over more slaves in the network. The results also indicate that ART has the highest DMR among the four methods. This is because that ART automatically re-sends messages when transient faults occur and it does not judiciously determine the number of replicas for messages. Our proposed framework REF is effective in controlling the DMR since in most cases of 16 slaves, the DMR of our REF is lower than that of NFT, ART, and EA. Fig. 7 shows the channel utilization of FT, ART, EA, and our REF. Obviously, our REF has the highest channel utilization among the four approaches regardless of network topology and message set. The reason is that our REF sends replicas for messages to improve system reliability while other approaches do not. We can also find that the channel utilization of all methods becomes higher along with the increase in the number of messages.



(a) 5 messages.

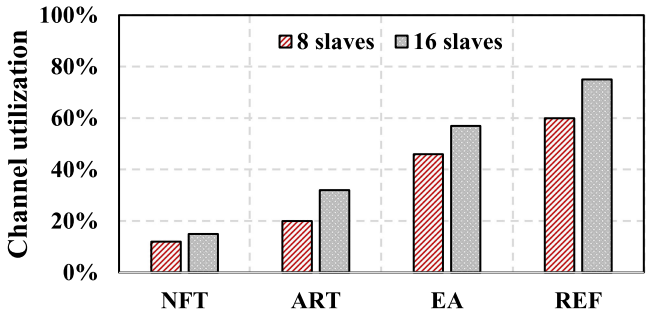


(b) 10 messages.

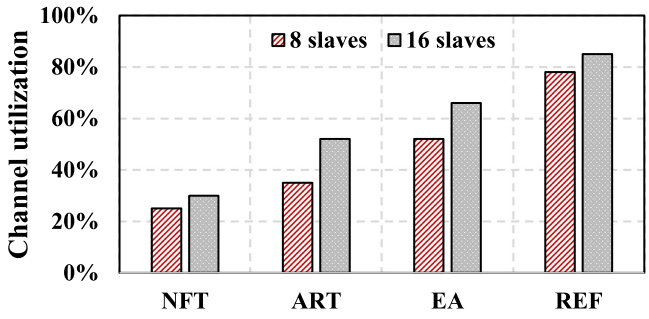


(c) 20 messages.

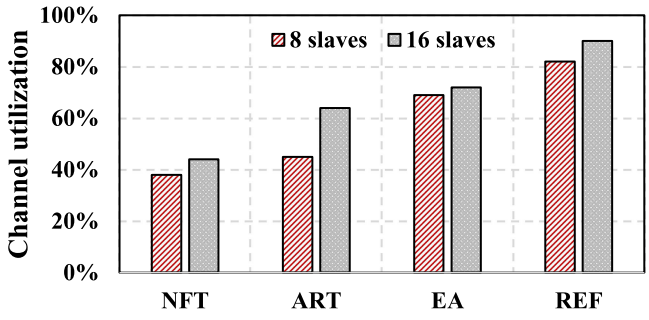
Fig. 6. The deadline miss rate Υ_{miss} of NFT, ART, EA [29] and our proposed framework REF in the case of (a) 5 messages (b) 10 messages and (c) 20 messages.



(a) 5 messages.



(b) 10 messages.



(c) 20 messages.

Fig. 7. The channel utilization U_{net} of NFT, ART, EA [29] and our proposed framework REF in the case of (a) 5 messages (b) 10 messages and (c) 20 messages.

Table 2 presents the CPU runtime (unit: ms) of the approaches NFT, ART, EA [29] and our proposed framework REF. The results show that the CPU runtime of these methods all grows with the increase in the number of messages and slaves. Among the four methods, EA always has the largest CPU runtime which becomes a few seconds for large-scale systems. This is because EA has a large number of time-consuming search operations. Although the runtime overhead of our proposed framework REF is higher than NFT and ART, it is still acceptable since even within the largest system scale (i.e., 20 messages and 16 slaves), the REF's CPU runtime is less than 0.5s.

TABLE 2
CPU Runtime (Unit: Ms) of NFT, ART, EA [29] and the Proposed Framework REF

	NFT	ART	EA	REF
5 messages, 8 slaves	40.0	41.0	283.0	53.0
5 messages, 16 slaves	66.0	42.0	500.0	87.0
10 messages, 8 slaves	75.0	54.0	456.0	99.0
10 messages, 16 slaves	123.0	84.0	774.0	169.0
20 messages, 8 slaves	221.0	150.0	1316.0	295.0
20 messages, 16 slaves	367.0	245.0	2210.0	483.0

6 Conclusion

In this article, we developed the first hazard-aware CPS model and presented a reliability enhancement framework that focuses on optimizing system reliability under energy, utilization, and timing constraints. Our framework is designed based on the PID feedback control technique to determine the message transmission and admission strategies used in hazard-aware CPSs. Extensive experiments were implemented to validate the efficacy of the proposed framework. The simulation results show that the proposed framework is effective in increasing reliability and decreasing the deadline miss rate of CPS network systems, with an acceptable runtime overhead. Compared to the three alternative approaches, the reliability improvement achieved by our framework can be up to 4.21×.

Inspired by [34], in the future we will extend our proposed methodologies to solve the lifetime reliability issue of

non-IT components in hazard-aware CPSs. This is because these non-IT components such as batteries may be a key part of CPS devices and they also may be prone to wear out in a hazardous environment due to their accelerated aging. The cross-layer control latency optimization problem considered in [35] is worth studying for CPSs. Therefore, we also plan to model and solve the latency optimization problem when deploying hierarchical control of CPSs.

REFERENCES

[1] S. Zhang and J. Liu, "Analysis and optimization of multiple unmanned aerial vehicle-assisted communications in post-disaster areas," *IEEE Trans. Veh. Technol.*, vol. 67, no. 12, pp. 12049–12060, Dec. 2018.

[2] K. Miranda, A. Molinaro, and T. Razafindralambo, "A survey on rapidly deployable solutions for post-disaster networks," *IEEE Commun. Mag.*, vol. 54, no. 4, pp. 117–123, Apr. 2016.

[3] G. C. Deepak, A. Ladas, Y. A. Sambo, H. Pervaiz, C. Politis, and M. A. Imran, "An overview of post-disaster emergency communication systems in the future networks," *IEEE Wirel. Commun.*, vol. 26, no. 6, pp. 132–139, Dec. 2019.

[4] J. Zhou, L. Li, A. Vajdi, X. Zhou, and Z. Wu, "Temperature-constrained reliability optimization of industrial cyber-physical systems using machine learning and feedback control," *IEEE Trans. Autom. Sci. Eng.*, early access, Mar. 12, 2021, doi: 10.1109/TASE.2021.3062408.

[5] M. Li, H. Nishiyama, N. Kato, Y. Owada, and K. Hamaguchi, "On the energy-efficient of throughput-based scheme using renewable energy for wireless mesh networks in disaster area," *IEEE Trans. Emerg. Top. Comput.*, vol. 3, no. 3, pp. 420–431, Sep. 2015.

[6] A. Al-Hourani, S. Kandeepan, and A. Jamalipour, "Stochastic geometry study on device-to-device communication as a disaster relief solution," *IEEE Trans. Veh. Technol.*, vol. 65, no. 5, pp. 3005–3017, May 2016.

[7] P. Li, T. Miyazaki, K. Wang, S. Guo, and W. Zhuang, "Vehicle-assist resilient information and network system for disaster management," *IEEE Trans. Emerg. Top. Comput.*, vol. 5, no. 3, pp. 438–448, Third Quarter 2017.

[8] P. M. Lima, M. V. S. Alves, L. K. Carvalho, and M. V. Moreira, "Security of cyber-physical systems: Design of a security supervisor to thwart attacks," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 3, pp. 2030–2041, Jul. 2022.

[9] K. Gai, M. Qiu, H. Zhao, and X. Sun, "Resource management in sustainable cyber-physical systems using heterogeneous cloud computing," *IEEE Trans. Sustain. Comput.*, vol. 3, no. 2, pp. 60–72, Second Quarter 2018.

[10] Y. Kawamoto, H. Nishiyama, and N. Kato, "MA-LTRT: A novel method to improve network connectivity and power consumption in mobile ad-hoc based cyber-physical systems," *IEEE Trans. Emerg. Top. Comput.*, vol. 1, no. 2, pp. 366–374, Dec. 2013.

[11] Y. Wang, "Trust quantification for networked cyber-physical systems," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 2055–2070, Jun. 2018.

[12] J. Zhou, T. Wang, W. Jiang, H. Chai, and Z. Wu, "Decomposed task scheduling for security-critical mobile cyber–physical systems," *IEEE Internet Things J.*, vol. 9, no. 22, pp. 22280–22290, Nov. 2022.

[13] A. Burg, A. Chattopadhyay, and K. Lam, "Wireless communication and security issues for cyber-physical systems and the internet-of-things," *Proc. IEEE*, vol. 106, no. 1, pp. 38–60, Jan. 2018.

[14] A. Golabi, A. Erradi, and A. Tantawy, "Towards automated hazard analysis for CPS security with application to CSTR system," *J. Process Control*, vol. 115, pp. 100–111, 2022.

[15] D. Zhu, R. Melhem, and D. Mosse, "The effects of energy management on reliability in real-time embedded systems," in *Proc. IEEE/ACM Int. Conf. Comput. Aided Des.*, 2004, pp. 35–40.

[16] 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.

[17] 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.*, vol. 70, no. 4, pp. 581–594, Apr. 2021.

[18] T. Karnik and P. Hazucha, "Characterization of soft errors caused by single event upsets in CMOS processes," *IEEE Trans. Dependable Secur. Comput.*, vol. 1, no. 2, pp. 128–143, Second Quarter 2004.

[19] 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, Dec. 2000.

[20] M. Riera, R. Canal, J. Abella, and A. Gonzalez, "A detailed methodology to compute soft error rates in advanced technologies," in *Proc. IEEE Des. Automat. Test Europe Conf. Exhib.*, 2016, pp. 217–222.

[21] J. Alzubi, O. Alzubi, and T. Chen, *Forward Error Correction Based on Algebr.-Geometric Theory*, 1st ed., Berlin, Germany: Springer, 2014.

[22] Bit Error Rate, 2022. [Online]. Available: https://en.wikipedia.org/wiki/Bit_error_rate

[23] T. Abdelzaher, Y. Diao, J. L. Hellerstein, C. Lu, and X. Zhu, "Introduction to control theory and its application to computing systems," in *Performance Modeling and Engineering*, Boston, MA, USA: Springer, 2008.

[24] Y. Ma, T. Chantem, R. P. Dick, and X. S. Hu, "Improving system-level lifetime reliability of multicore soft real-time systems," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 25, no. 6, pp. 1895–1905, Jun. 2017.

[25] 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 Des. Integr. Circuits Syst.*, vol. 39, no. 1, pp. 88–100, Jan. 2020.

[26] PID Controller, 2022. [Online]. Available: https://en.wikipedia.org/wiki/PID_controller

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

[28] M. A. Haque, H. Aydin, and D. Zhu, "On reliability management of energy-aware real-time systems through task replication," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 3, pp. 813–825, Mar. 2017.

[29] J. Zhou, J. Sun, X. Zhou, T. Wei, M. Chen, S. Hu, and X. S. Hu, "Resource management for improving soft-error and lifetime reliability of real-time MPSoCs," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 38, no. 12, pp. 2215–2228, Dec. 2019.

[30] OMNeT++, 2022. Accessed: Jul. 31, 2018. [Online]. Available: https://omnetpp.org/

[31] L. Li, P. Cong, K. Cao, J. Zhou, T. Wei, M. Chen, and X. S. Hu, "Feedback control of real-time EtherCAT networks for reliability enhancement in CPS," in *Proc. Des. Automat. Test Eur.e Conf. Exhib.*, 2018, pp. 688–693.

[32] M. Ebrahimi, L. Chen, H. Asadi, and M. B. Tahoori, "CLASS: Combined logic and architectural soft error sensitivity analysis," in *Proc. IEEE 18th Asia South Pacific Des. Automat. Conf.*, 2013, pp. 601–607.

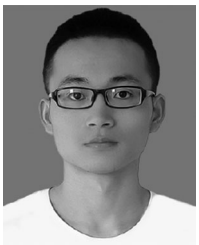
[33] J. Zhou, K. Cao, X. Zhou, M. Chen, T. Wei, and S. Hu, "Throughput-conscious energy allocation and reliability-aware task assignment for renewable powered in-situ server systems," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 41, no. 3, pp. 516–529, Mar. 2022.

[34] L. Liu, C. Li, H. Sun, Y. Hu, J. Gu, and T. Li, "BAAT: Towards dynamically managing battery aging in green datacenters," in *Proc. 45th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw.*, 2015, pp. 307–318.

[35] X. Hou, C. Li, J. Liu, L. Zhang, Y. Hu, and M. Guo, "ANT-Man: Towards agile power management in the microservice era," in *Proc. IEEE Int. Conf. High Perform. Comput. Netw. Storage Anal.*, 2020, pp. 1–14.

Peijin Cong received the BS and PhD degrees in computer science from East China Normal University, Shanghai, China, in 2016 and 2021. She is currently with the Nanjing University of Science and Technology, Nanjing, China. Her research interests include cyber-physical systems and cloud computing. She has published 20 refereed papers, including more than 10 in premier *IEEE/ACM Transactions*.

Junlong Zhou (Member, IEEE) is currently an associate professor with the Nanjing University of Science and Technology, Nanjing, China. He has authored or co-authored more than 100 refereed papers, including 35 *IEEE/ACM Transaction* papers in his research areas, which include embedded systems, edge computing, and CPS. He has been an associate editor for the JCSC and the IET CPS, a subject area editor for the JSA, and a guest editor for 7 Journals such as *ACM Transactions on Cyber-Physical Systems*. He was the recipient of the Best Paper Awards from IEEE iThings 2020 and IEEE CPSCom 2022.



Weiming Jiang received the BS degree in communication engineering from the Nanjing University of Science and Technology, Nanjing, China, in 2020, where he is currently working toward the MS degree. His research interest includes is cyber security.

Mingsong Chen (Senior Member, IEEE) received the BS and ME degrees from Computer Science and Technology Department, Nanjing University, Nanjing, China, in 2003 and 2006, respectively, and the PhD degree in computer engineering from the University of Florida, Gainesville, in 2010. He is currently a professor with the SEI, East China Normal University. His research interests include CPS and embedded systems. He has published more than 100 papers in premier conferences and journal including DAC, RTSS, *IEEE Transactions on Computers* and *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*.

Shiyan Hu (Senior Member, IEEE) received the PhD degree in computer engineering from Texas A&M University, in 2008. He is a professor and chair in Cyber-Physical System (CPS) Security with the University of Southampton. His research interests include CPS and CPS Security, where he has published more than 150 refereed papers in journals and conferences, about half of which appeared in *IEEE/ACM Transactions*. He is the editor-In-chief of IET CPS: Theory & Applications. He is an associate editor for *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, *IEEE Transactions on Industrial Informatics*, *IEEE Transactions on Circuits and Systems I: Regular Papers*, *ACM Transactions on Design Automation of Electronic Systems* and *ACM Transactions on Cyber-Physical Systems*. He is a member of the European Academy of Sciences and Arts, a fellow of IET, and a fellow of the British Computer Society.

Keqin Li (Fellow, IEEE) is a SUNY distinguished professor of computer science with the State University of New York. He is also a national distinguished professor with Hunan University, China. His research interests include cloud computing, edge computing, embedded systems and CPS, high-performance computing. He has published more than 840 journal articles, book chapters, and refereed conference papers, and has received several best paper awards. He has chaired many international conferences. He is currently an associate editor of the ACM CSUR. He has served on the editorial boards of *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Transactions on Computers*, *IEEE Transactions on Cloud Computing*, *IEEE Transactions on Services Computing*, and *IEEE Transactions on Sustainable Computing*.