IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received June 29, 2019, accepted July 18, 2019, date of publication July 22, 2019, date of current version August 7, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2930242

Throughput Maximization for Multicore Energy-Harvesting Systems Suffering Both Transient and Permanent Faults

JUNLONG ZHOU[1], (Member, IEEE), PEIJIN CONG[2], JIN SUN[1], (Member, IEEE), XIUMIN ZHOU[1], TONGQUAN WEI[2], (Senior Member, IEEE), AND MINGSONG CHEN[3], (Senior Member, IEEE)

[1]School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China
[2]School of Computer Science and Technology, East China Normal University, Shanghai 200062, China
[3]Shanghai Key Laboratory of Trustworthy Computing, East China Normal University, Shanghai 200062, China

Corresponding author: Junlong Zhou (jlzhou@njust.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61802185, Grant 61872185, and Grant 61872147, in part by the Natural Science Foundation of Jiangsu Province under Grant BK20180470, in part by the National Key Research and Development Program of China under Grant 2018YFB2101301, in part by the Fundamental Research Funds for the Central Universities under Grant 30919011233, and in part by the Shanghai Municipal Natural Science Foundation under Grant 16ZR1409000.

ABSTRACT Harvesting renewable generation (e.g., solar energy) from the ambient environment to achieve a near perpetual operation for embedded systems is being paid more and more attention by academia and industry. However, an immediate problem along with the utilization of renewable energy is the degraded system throughput caused by the intermittent characteristic of renewable generation. On the other hand, energy-harvesting systems (EHSs) deployed in harsh environment are more vulnerable to transient and permanent faults. This paper aims at scheduling dependent tasks on a multicore platform for throughput maximization under energy and reliability constraints. The target of this paper is to design algorithms that optimize system throughput under the energy, reliability, as well as task precedence constraints. To achieve this goal, we propose a mixed-integer linear programming (MILP) approach for allocating and scheduling precedence constrained tasks on the multicore to maximize the throughput of EHS. However, the MILP may find the optimal solution in an exponential time. To overcome this difficulty, we propose a polynomial-time heuristic algorithm to solve the MILP-based throughput maximization problem. In this heuristic algorithm, the uncertainty in energy sources is considered and the allocation and scheduling of tasks are determined based on system energy state. The extensive simulation experiments are carried out to validate our MILP approach and throughput-aware heuristic algorithm. The simulation results justify that the MILP approach achieves an up to 92.9% improvement of system throughput when compared with a baseline method, and the proposed heuristic improves system throughput by up to 32.1% on average when compared with the four representative existing approaches.

INDEX TERMS Energy-harvesting systems, lifetime reliability, soft-error reliability, throughput.

I. INTRODUCTION

Power and energy are both critical design concerns of embedded applications, especially for battery-powered systems that are deployed in harsh environment [1]–[3]. Human beings either have no access to these systems or have difficulty in replacing a battery for these systems since these systems are in general deployed in extreme environments. Therefore, it is desirable for embedded systems deployed in such an environment to harvest energy from ambient environment to sustain their perpetual operation. For this reason, energy-harvesting systems (EHSs) in which energy is provided by external sources, e.g., ambient vibration, heat, or light, have been popularly used as suitable alternatives to traditional battery-powered systems. EHSs are expected to address both energy shortage and environmental challenges to a great extent. However, affected by the intermittent issue in renewable generation resources, EHS may fail to complete all the arrival tasks, leading to the degradation in system throughput, which is defined as the number of tasks being

The associate editor coordinating the review of this manuscript and approving it for publication was Michele Magno.

completed during a scheduling horizon [5]. In this paper, we are interested in maximizing the throughput of EHS running on multicore platforms.

As CMOS feature size continues to shrink down, multicore processors are now seriously under the threat of transient faults and permanent faults that would in turn result in soft errors and hard errors, respectively. The situation becomes even more severe for EHS deployed in harsh environments where the integrated circuits are easier to suffer high-energy neutron and alpha particle strike-induced transient faults and the cost of repairing or replacing the disable hardware is generally prohibitive. Thus, handling both transient and permanent faults to improve soft-error reliability (SER) as well as lifetime reliability (LTR) is of great importance for low-power, energy-harvesting embedded systems.

With that in mind, this paper focuses on solving the problem motivated by a common application of EHS, in-situ data processing systems (*InS* systems). These systems are deployed in harsh environments such as oil/gas exploration [6], astronomy observing in remote area [7], rural geographical surveying [8], and video surveillance for behavioral studies of wildlife [9]. The *InS* systems are powered by renewable energy [10], which may be insufficient to support the processing of all in-situ workloads. Under this circumstance, a high system throughput is preferred to support in-situ workloads to the most extent. Besides, SER and LTR optimization are imperative for *InS* systems since the systems are very vulnerable to transient and permanent faults due to their hash operating environment. Taking all the above into consideration, we aim to address the problem of maximizing throughput for multicore EHS suffering both transient and permanent faults. We make the following major contributions:

- We formulate the concerned problem as MILP model that determines an optimal schedule of dependent tasks with energy as well as reliability constraints on a multicore EHS to maximize system throughput.
- Since MILP may take an exponential time in solution space exploration, we design a polynomial-time algorithm to maximize the throughput of EHS, which determines task allocation and scheduling strategies based on system energy states.
- We develop an earliest-finish-time based list scheduling algorithm for EHS with plentiful energy supply and a cross entropy based task scheduling algorithm for EHS with insufficient energy supply.
- We carry out a series of simulations to validate the efficacy of our proposed MILP and heuristic algorithms by comparing their performance with that of a baseline method as well as four peer approaches.

The remainder of the paper is organized as follows. We review related work in Section II discusses existing methods relevant to this work. Section III presents system models and further formulates our concerned throughput maximization problem. Section IV presents an MILP approach to solve the throughput maximization problem. Section V describes the proposed throughput-aware task scheduling algorithm. Sections VI shows our experimental setups and results, and Section VII concludes the paper.

II. RELATED WORK

A substantial number of research efforts have been developed towards solving the problem of resource management and task scheduling for embedded systems with energy harvesting. Most of these research efforts are made from two perspectives of improving the harvesting efficiency and utilization of renewable energy. One perspective is concerned about how to maximize the power output harvested from a renewable energy source [11], [12]. The other perspective concentrates on how to exploit the fluctuating energy generated by the source efficiently [13], [14]. Different from [11]–[14] that do not consider energy savings, Chen et al. [4] proposed a dynamic frequency selection scheme to improve energy efficiency under the deadline miss rate constraint for real-time applications in EHS, and Liu *et al.* [15] presented an adaptive dynamic programming based algorithm to improve electricity efficiency of the residential grid. However, all the above works do not explore how to maximize the throughput of EHS under the intermittent renewable energy.

Numerous studies have discussed the approaches for addressing the throughput maximization problems. However, the existing techniques are either designed for energy harvesting networked systems [16]–[19] and energy harvesting power systems [20], or embedded systems [21] and data centers [22] that ignore the uncertainty in energy sources. For example, Yuan *et al.* [22] proposed a workload-aware task scheduling scheme that wisely decides the optimal combination of virtual machine and routing path for tasks. Unlike [22], the approaches proposed in [23] and [24] both utilize renewable power and are designed for green data centers. Specifically, the scheme [23] considers the temporal variation in grid price and renewable energy source, and intelligently schedules user requests under the delay bounds. The method [24] schedules all the arrival tasks cost-efficiently to meet user requests' delay-bound constraints by exploiting the spatial diversity in distributed green cloud data centers. These approaches are effective but do not consider SER and LTR.

Several recent works [25]–[29] have addressed the SER and LTR co-optimization problem. Zhou *et al.* [25] proposed a SER and LTR-balanced task frequency and replication selection strategy for maximizing system availability. Ma *et al.* [26] presented a reliability improvement framework which exploits the power features of the Big-Little type cores to maximize SER under LTR, power, and real-time constraints. Kim *et al.* [27] introduced DVFS and Q-learning techniques to optimize lifetime as well as energy for many-core microprocessors in the presence of both transient and permanent faults. Based on the impacts of hardware-as well as application-level variations on SER, a variation-aware task scheduling scheme [28] is developed to maximize SER while meeting a constraint on LTR. Unlike the literature [26]–[28] that either optimize SER or LTR,

TABLE 1. A comparison summary of related works from multiple aspects, such as optimization goal, constraint, utilization of renewable energy, target system, and application. * indicates that the corresponding reference does not provide specific discussions on design constraint, renewable energy type, or application.

References	Optimization Goal	Constraint	Renewable Energy	Target System	Application
Chen et al. [4]	Energy efficiency	Deadline miss rate	Yes (solar power)	Embedded systems	Real-time tasks
Esram et al. [11]	Output power	*	Yes (solar power)	Photovoltaic systems	*
Kobayashi et al. [12]	Output power	*	Yes (solar power)	Photovoltaic systems	*
Sharma et al. [13]	Throughput and delay	Energy	Yes (*)	Sensor networks	Network packets
Abdeddaim et al. [14]	Feasibility	Energy	Yes (*)	Embedded systems	Real-time tasks
Liu et al. [15]	Electricity efficiency	*	Yes (solar power)	Smart grid systems	Housing units
Tutuncuoglu et al. [16]	Throughput	Battery storage	Yes (*)	Relay channels	Transmit powers
Gupta et al. [17]	Throughput	Causality and overflow of energy and data	Yes (*)	Successive relaying networks	Energy arrival events
Mehrabi et al. [18]	Throughput	Energy	Yes (*)	Wireless sensor networks	Mobile sinks
Wu et al. [19]	Throughput	Energy and capacity	Yes (*)	Communication systems	Communication channels
Wang et al. [20]	Spinning reserve cost	Transmission capability	Yes (wind power)	Power systems	Spinning reserves
Huang et al. [21]	Throughput	Temperature	No	Embedded systems	Real-time tasks
Yuan et al. [22]	Revenue	Round-trip time	No	Cloud data centers	User requests
Yuan et al. [23]	Profit	Delay	Yes (hybrid power)	Green data centers	User requests
Yuan et al. [24]	Cost	Delay	Yes (hybrid power)	Green cloud data centers	User requests
Zhou et al. [25]	Availability	Throughput	No	Embedded systems	Frame-based tasks
Ma et al. [26]	SER	LTR	No	Embedded systems	Real-time tasks
Tang et al. [27]	LTR+energy	Temperature	No	Embedded systems	Real-time tasks
Zhou et al. [28]	SER	Deadline	No	Embedded systems	Real-time tasks
Zhou et al. [29]	SER+LTR	Deadline	No	Embedded systems	Real-time tasks

an evolutionary-based algorithm is designed to optimize SER and LTR simultaneously [29]. However, the above-mentioned approaches are not developed for systems with uncertain energy supply, and fail to take throughput into consideration.

For a better understanding, we summarize related works from multiple aspects in TABLE 1 such as optimization goal, constraint, utilization of renewable energy, target system, and application. From the table readers can easily find that none of these existing works considers throughput and reliability (SER and LTR) simultaneously for energy harvested embedded systems. Unlike the existing works, this paper concentrates on optimizing the throughput of EHS running on the multicore embedded systems suffering both transient and permanent faults. In this paper, we present an allocation and scheduling scheme that uses an MILP solver to maximize system throughput under the energy, reliability, as well as task precedence constraints. To efficiently solve this NP-hard allocation and scheduling problem, we also propose an energy uncertainty-aware task scheduling heuristic algorithm in which the allocation, frequency, and execution order of tasks are determined to maximize system throughput.

III. SYSTEM MODEL AND PROBLEM DEFINITION

Below we introduce system models and define the problem that we are trying to solve.

A. ARCHITECTURE AND APPLICATION MODEL

The EHS is mainly composed of three modules (as shown in Fig. 1): energy source module, storage module, as well as dissipation module. The energy source module

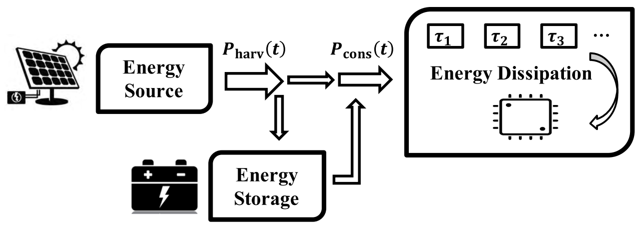


FIGURE 1. The diagram of the system architecture.

scavenges solar energy automatically at the rate of $P_{\text{harv}}(t)$. The scavenged energy will be further transformed into electrical energy. The energy storage module, usually implemented by a battery or super-capacitor, works as a buffer against the uncertainty of harvested energy. An embedded system running on the multicore C is used as the energy dissipation module. The multicore C consists of M homogeneous cores $\{C_1, C_2, \cdots, C_M\}$. All the cores are dynamic voltage and frequency scaling-enabled and support multiple discrete supply voltage and frequency levels. Let $V_{\min}/F_{\min}$ and $V_{\max}/F_{\max}$ be the minimum and maximum voltage/frequency equipped with the multicore, respectively. The kth ($1 \leq k \leq K$) voltage/frequency level supported by the multicore then satisfies $V_{\min}/F_{\min} \leq V_k/F_k \leq V_{\max}/F_{\max}$, where K is the number of voltage/frequency levels.

Suppose the multicore C hosts multiple applications with precedence constraints, each of which can be modeled as a directed acyclic graph (DAG) $G = (\mathcal{V}, \mathcal{E})$ [29]–[31]. Each graph contains a set $\mathcal{V}$ of vertices representing the set of task nodes and a set $\mathcal{E}$ of edges representing the partial order of tasks. The edge $(\tau_i, \tau_j) \in \mathcal{E}$ ($1 \leq i, j \leq |\mathcal{V}|$) imposes the precedence constraint that task τ_j cannot execute until its

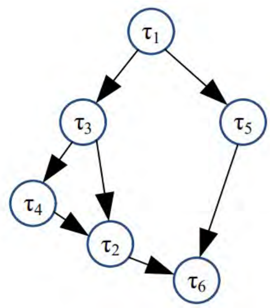


FIGURE 2. Example of applications modeled as DAG.

predecessor τ_i has finished execution. The communication time between tasks τ_i and τ_j is denoted by $CMT(\tau_i, \tau_j)$. Fig. 2 illustrates an example of DAG applications. Considering that a task may have multiple predecessors and successors, $Pre(\tau_i)$ and $Succ(\tau_i)$ are used to represent the set of task τ_i's immediate predecessors and successors. Let wc_i represent the worst-case execution cycles of task τ_i, the execution time of τ_i running at frequency F_k is then calculated as

$$ET(\tau_i, F_k) = \frac{wc_i}{F_k}. \tag{1}$$

B. ENERGY MODEL

We discuss the energy model for the concerned EHS from supply and demand perspectives separately.

1) ENERGY SUPPLY

We use $P_{\text{harv}}(t)$ to denote the harvesting power, and $E_{\text{harv}}(t_1, t_2)$ to denote the energy harvested from the external environment during time interval $[t_1, t_2]$, $E_{\text{harv}}(t_1, t_2)$ is then formulated as

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

As illustrated in Fig. 1, the multicore system consumes a portion of the harvested energy and the energy storage module stores the residuals. Both of the two modules are able to supply the energy to the energy dissipation module. Let $E_{\text{sup}}(t_1, t_2)$ represent the energy available in time interval $[t_1, t_2]$, and $E(t_1)$ represent the energy stored into the storage module at time point t_1, the supply energy can be derived as

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

2) ENERGY DEMAND

The power consumed by a CMOS device can be decomposed into dynamic and static portions, i.e.,

$$P_{\text{cons}} = P_{\text{dyn}} + P_{\text{sta}}. \tag{4}$$

The dynamic power P_{dyn} is associated with core's switching activity. A general way is to model dynamic power as a convex function of frequency. The static power P_{sta} is the power dissipated by the CMOS circuit itself, and is independent of switching activity. As the dynamic power is only consumed for executing tasks and the static power is dissipated to maintain circuit state, the total energy demanded by executing tasks in set $\mathcal{V}$ on the multicore C during a scheduling horizon $\mathcal{H}$ is

$$\begin{aligned} E_{\text{cons}}(\mathcal{V}, C, \mathcal{H}) &= \sum\nolimits_{m=1}^{M} P_{\text{sta}}(C_m) \times \mathcal{H} + \sum\nolimits_{m=1}^{M} \sum\nolimits_{\tau_i \in \mathcal{V}_m} P_{\text{dyn}}(\tau_i, C_m) \\ &\quad \times ET(\tau_i, F_k), \end{aligned} \tag{5}$$

where $P_{\text{sta}}(C_m)$ is the static power of core C_m and $P_{\text{dyn}}(\tau_i, C_m)$ is the dynamic power of executing task τ_i on core C_m. $\mathcal{V}_m$ is the set of tasks allocated to core C_m and $ET(\tau_i, F_k)$ is τ_i's execution time running at frequency F_k.

C. RELIABILITY MODEL

We discuss the reliability model for the concerned EHS from SER and LTR perspectives separately.

1) SOFT-ERROR RELIABILITY

SER is determined by the average fault arrival rate that is calculated as the expected failure number occurring per second. Using the exponential model proposed by Zhu *et al.* [32], the core raw fault rate at frequency F_k is

$$\lambda(F_k) = \lambda_{F_{\max}} \times 10^{\frac{F_{\max}-F_k}{\Omega}}, \tag{6}$$

where $\lambda_{F_{\max}}$ is core's fault rate when it is operating at the maximum frequency, and parameter Ω indicates the trend of fault rate increase with regard to the reduced frequency.

The SER of a task is defined as the probability of successfully executing the task while suffering no transient faults, and can be modeled using the exponential failure law [32]. Given task τ_i running on the multicore at frequency F_k, the SER of the task is then calculated as

$$SER(\tau_i, F_k) = e^{-\lambda(F_k)\times VF_i \times \frac{wc_i}{F_k}}, \tag{7}$$

where $\lambda(F_k) \times VF_i$ is the ultimate fault rate considering the task error probability and wc_i/F_k is task τ_i's execution time running at frequency F_k. Since a system's correct operation relies on the successful execution of all tasks, we formulate system SER as

$$SER_{\text{sys}} = \prod\nolimits_{\tau_i \in \mathcal{V}} \prod\nolimits_{F_k \in F} SER(\tau_i, F_k), \tag{8}$$

where F is the frequency set supported by the multicore.

2) LIFETIME RELIABILITY

LTR is decided by multiple wear-out effects such as electromigration, time dependent dielectric breakdown, stress migration, and thermal cycling [25]. For sake of simplicity, in this work we consider electromigration (EM) as the primary source of permanent faults for simplicity. EM is the dislocation of metal atoms due to momentum imparted by electrical current in wires as well as vias [33]. It is worth emphasizing that, we can easily extend this reliability model to incorporate other wear-out effects by the sum-of-failure-rate model [26].

LTR is generally evaluated upon the mean time to failure (MTTF) metric. According to the MTTF model for EM [33], core C_m's MTTF is calculated as

$$MTTF(C_m) = \int_0^\infty R_{\mathrm{LTR},m}(t)dt = \int_0^\infty e^{-(A_m t)^\alpha} dt, \quad (9)$$

where $R_{\mathrm{LTR},m}$ is the LTR of core C_m following the Weibull distribution, A_m is the aging rate of core C_m, and α is the slope coefficient of the Weibull distribution. A_m is determined by the hardware as well as the thermal profile of core C_m. The calculation method of A_m is provided in [34]. As in [34], [35], the system is deemed as failed as long as an arbitrary core in the system fails, the system MTTF is thereby obtained as

$$MTTF_{\mathrm{sys}} = \min_m MTTF(C_m), \quad \forall m = 1, 2, \cdots, M. \quad (10)$$

D. PROBLEM DEFINITION

We focus on solving the problem motivated by in-situ data processing applications deployed in special operating environments. These systems are powered by renewable energy and have throughput and reliability requirements. The problem that we aim to solve is described as follows. Given an application represented by a DAG $G = (\mathcal{V}, \mathcal{E})$ to be scheduled on the multicore system $C = \{C_1, C_2, \cdots, C_M\}$, and an estimated harvested energy budget E_{bgt} during a scheduling horizon $\mathcal{H}$, design a task scheduling scheme to maximize system throughput under the limited supply energy while satisfying the constraints on SER, LTR, and task precedence.

To solve the problem, we provide an MILP approach and two heuristic algorithms for allocating and scheduling dependent tasks with energy as well as reliability constraints on the multicore system. The details of our MILP approach and heuristic algorithms are introduced in the following sections.

IV. MILP-BASED APPROACH

This section presents our MILP approach to solving the throughput maximization problem described in Section III-D and discusses shortcomings of the MILP approach. The approach maximizes the system throughput under the constraints of reliability, energy, as well as task dependency by determining i) on which cores should the tasks be executed, ii) what voltages/frequencies should be used for the tasks, and iii) when should the tasks start. Before showing the MILP formulation, the binary variables used in the formulation are defined first below.

$$\eta(\tau_i, C_m) = \begin{cases} 1 & \text{if } \tau_i \text{ is allocated to } C_m \\ 0 & \text{otherwise.} \end{cases} \quad (11)$$

$$\varphi(\tau_i, \tau_j) = \begin{cases} 1 & \text{if } \tau_i \text{ starts before } \tau_j, \\ 0 & \text{otherwise.} \end{cases} \quad (12)$$

$$\delta(\tau_i, F_k) = \begin{cases} 1 & \text{if } \tau_i \text{ is executed at frequency } F_k, \\ 0 & \text{otherwise.} \end{cases} \quad (13)$$

A. OBJECTIVE

The goal of the MILP formulation is to maximize the system throughput in a scheduling horizon using the harvested energy, which is calculated as the number of task instances executed by the multicore EHS system. Given the input task set $\mathcal{V}$ and the multicore C, the system throughput during the scheduling horizon $\mathcal{H}$ is then expressed as

$$Tru_{\mathrm{sys}} = sizeof(\mathcal{V}_{\mathrm{exe}}) \quad (14)$$

where $\mathcal{V}_{\mathrm{exe}}$ is the set of tasks executed on the multicore and $\mathcal{V}_{\mathrm{exe}} \subset \mathcal{V}$ holds.

B. CONSTRAINTS

Let $\mathcal{N} \triangleq \{1, 2, \cdots, |\mathcal{V}_{\mathrm{exe}}|\}$, $\mathcal{M} \triangleq \{1, 2, \cdots, M\}$, and $\mathcal{K} \triangleq \{1, 2, \cdots, K\}$, the constraints that must be satisfied are then formulated as below.

1) *Constraint on Task-to-Core Allocation:* every task is allocated to exactly one core.

$$\sum_{m=1}^{M} \eta(\tau_i, C_m) = 1, \quad \forall i \in \mathcal{N}. \quad (15)$$

2) *Constraint on Frequency-to-Task Assignment:* every task is exactly executed at one frequency level.

$$\sum_{k=1}^{K} \delta(\tau_i, F_k) = 1, \quad \forall i \in \mathcal{N}. \quad (16)$$

3) *Constraint on Energy Consumption:* the total energy consumed by executing tasks cannot exceed the energy supply.

$$E_{\mathrm{cons}}(\mathcal{V}_{\mathrm{exe}}, C, \mathcal{H}) \leq E_{\mathrm{sup}}(\mathcal{H}). \quad (17)$$

4) *Constraint on SER and LTR:* the system SER and LTR should be no less than the thresholds.

$$SER_{\mathrm{sys}} \geq SER_{\mathrm{th}}, \quad (18)$$
$$MTTF_{\mathrm{sys}} \geq MTTF_{\mathrm{th}}. \quad (19)$$

5) *Constraint on Task Dependency and Non-Preemption:* the order of task executions cannot violate task dependency and no task executed on the same core can overlap with each other.

$$\forall i, j \in \mathcal{N}, \quad i \neq j,$$
$$\varphi(\tau_i, \tau_j) + \varphi(\tau_j, \tau_i) \geq 0, \quad (20)$$
$$\varphi(\tau_i, \tau_j) + \varphi(\tau_j, \tau_i) \leq 1, \quad (21)$$
$$t_{\mathrm{start}}(\tau_i) \leq t_{\mathrm{start}}(\tau_j) + \big(1 - \varphi(\tau_i, \tau_j)\big) \times \Delta \times \mathcal{H}, \quad (22)$$
$$t_{\mathrm{start}}(\tau_j) \leq t_{\mathrm{start}}(\tau_i) + \varphi(\tau_i, \tau_j) \times \Delta \times \mathcal{H}, \quad (23)$$
$$\forall i, j \in \mathcal{N}, \quad i \neq j, \ \forall m \in \mathcal{M}, \ \forall k \in \mathcal{K}$$
$$t_{\mathrm{finish}}(\tau_i) \leq t_{\mathrm{start}}(\tau_j) + (3 - \eta(\tau_i, C_m) - \eta(\tau_j, C_m) - \varphi(\tau_i, \tau_j)) \times \Delta \times \mathcal{H}, \quad (24)$$
$$t_{\mathrm{finish}}(\tau_j) \leq t_{\mathrm{start}}(\tau_i) + (2 - \eta(\tau_i, C_m) - \eta(\tau_j, C_m) + \varphi(\tau_i, \tau_j)) \times \Delta \times \mathcal{H}, \quad (25)$$

where $t_{\mathrm{start}}/t_{\mathrm{finish}}$ is the start/finish of a task and Δ ($\Delta \geq 1$) is a constant. Eq. (22) indicates that task τ_i must start before τ_j if $\varphi(\tau_i, \tau_j) = 1$ and Eq. (24) ensures that task τ_i completes its execution before task τ_j if τ_i and τ_j are running on the same core as well as τ_i starts before τ_j. Similar conditions hold for Eqs. (23) and (25).

C. LIMITATION OF MILP APPROACH

Our target is to find the optimal solution to our problem from the solution space. The problem can be optimally solved for systems of a small granularity using an MILP solver. However, since the MILP problem is NP-hard, the complexity may increase exponentially when solving systems of a large granularity. In this case, MILP solvers could not serve the purpose even for design time exploration. Therefore, developing a time-efficient algorithm to find a sub-optimum solution becomes a necessity. In the subsequent section, we present a polynomial-time scheduling strategy to maximize system throughput under the energy, reliability, and task precedence constraints.

V. THROUGHPUT-AWARE TASK SCHEDULING SCHEME

This section describes the proposed throughput-aware task scheduling scheme in detail. The scheme features the consideration of uncertainty in renewable energy sources and handles the uncertainty by dividing the system operation into high energy state as well as low energy state. Two heuristic algorithms are provided in the scheme to maximize the throughput of systems in the two energy states, respectively.

A. SOLVE THE ENERGY UNCERTAINTY

The available energy used to support system operation varies in the scheduling horizon (i.e., $\mathcal{H}$) because of the intermittent nature of renewable energy sources. To handle this uncertainty, we first divide system operation into two states with respect to energy: high energy state as well as low energy state. If there is sufficient energy available for the EHS system to complete all the tasks in the scheduling horizon at high speed, the system is in high energy state; whereas in low energy state in the opposite case. We then propose an energy state-aware approach (ESA) to solve our studied problem.

Algorithm 1 Energy State-Aware Approach

1 calculate the supply energy $E_{\text{sup}}(\mathcal{H})$ using Eqs. (2)-(3);
2 compute the energy $E_{\text{cons}}^{\max}(\mathcal{V}, C, \mathcal{H})$ consumed by all tasks running at the maximum frequency using Eq. (5);
3 **if** $E_{\text{cons}}^{\max}(\mathcal{V}, C, \mathcal{H}) \leq E_{\text{sup}}(\mathcal{H})$ **then**
  `/* high energy state */`
4   call Alg. 2 (i.e., earliest-finish-time based list scheduling algorithm);
5 **end**
6 **else**
  `/* low energy state */`
7   call Alg. 3 (i.e., cross-entropy based task scheduling algorithm);
8 **end**

Alg. 1 summarizes the algorithmic flow of ESA. The algorithm first estimates the energy $E_{\text{sup}}(\mathcal{H})$ supplied for the system in the scheduling horizon $\mathcal{H}$ using Eqs. (2)-(3) and the energy $E_{\text{cons}}^{\max}(\mathcal{V}, C, \mathcal{H})$ demanded by executing all tasks at the maximum frequency using Eq. (5) (lines 1-2). In case that there is sufficient energy to meet the system's energy demand, i.e., $E_{\text{cons}}^{\max}(\mathcal{V}, C, \mathcal{H}) \leq E_{\text{sup}}(\mathcal{H})$, indicating the system is in high energy state, earliest-finish-time based list scheduling (EFT-LS) heuristic is called to maximize system throughput (lines 3-5). Otherwise, the system is in low energy state and cross entropy based task scheduling (CE-TS) heuristic is called to maximize system throughput (lines 6-8). The details of EFT-LS and CE-TS are presented in Algs. 2 and 3, respectively.

B. EARLIEST-FINISH-TIME BASED LIST SCHEDULING FOR HIGH ENERGY STATE

We apply list scheduling (LS) [36] to solve our throughput maximization problem for systems operating at high energy state. Following the design of LS, EFT-LS is composed of a task prioritization phase (TPP) that determines the scheduling order (priority) of all tasks, as well as a core selection phase (CSP) that selects tasks in the order of their priorities and allocates the tasks onto most appropriate cores. It has been observed in [21] that system throughput is maximized if the latency of executing all tasks in the application is minimized in case of sufficient energy supply. In addition, the execution latency of a DAG application is in fact the finish time of the exit task, which is minimized if the finish time of its predecessors are minimized. Motivated by these observations, EFT-LS maximizes system throughput by minimizing the finish time of tasks.

TPP of EFT-LS: EFT-LS sets the priorities of tasks using the rank value $rank_{\text{EFT}}$ that is calculated based on the task execution time and communication time. Given task τ_i and its successors $Succ(\tau_i)$, the rank of τ_i is recursively defined as

$$rank_{\text{EFT}}(\tau_i) = \frac{\sum_{k=1}^{K} ET(\tau_i, F_k)}{K} + \max_{\tau_j \in Succ(\tau_i)} \big(CMT(\tau_i, \tau_j) + rank_{\text{EFT}}(\tau_j)\big). \tag{26}$$

As can be seen from Eq. (26), the rank is calculated recursively by traversing the task graph upward from the exit task τ_{exit}, of which the rank value is derived as

$$rank_{\text{EFT}}(\tau_{\text{exit}}) = \sum_{k=1}^{K} \frac{ET(\tau_{\text{exit}}, F_k)}{K}. \tag{27}$$

After obtaining the rank values of all tasks, the task scheduling list is then generated by sorting the tasks in the non-increasing order of $rank_{\text{EFT}}$. According to the definition of $rank_{\text{EFT}}$, we can easily deduce that the non-ascending order of $rank_{\text{EFT}}$ ensures a topological task order preserving the precedence constraints among tasks.

CSP of EFT-LS: Given the task scheduling list, EFT-LS tentatively puts the task under scheduling on all the cores and selects the core that delivers the shortest EFT. The key of EFT-LS is to minimize the EFT of tasks. To achieve this goal, EFT-LS assumes all the tasks run at the maximum frequency $F_{\max}$, which is a safe operation since the energy supply is plentiful. In addition, EFT-LS utilizes a task insertion policy that inserts a task in the idle time slot between two consecutively scheduled tasks on the same core without violating the task precedence constraint. The length of the idle time slot,i.e.,the timespan between the start time and finish time of the two consecutively scheduled tasks, should be longer than the execution time of the task to be inserted. By exploiting the idle time, the execution latency of the task set can be further reduced.

Algorithm 2 EFT-Based List Scheduling

1 compute $rank_{\mathrm{EFT}}$ for all tasks using Eq. (26);
2 sort the tasks in a scheduling list in the non-ascending order of $rank_{\mathrm{EFT}}$ values by $Rank = UpRank(\mathcal{V})$;
3 **while** the list *Rank* is not empty **do**
4   select the first task τ_i from the list for scheduling;
5   **for** each core C_m in set C **do**
6   derive $EFT(\tau_i)$ of task τ_i that runs on core C_m at frequency $F_{\max}$ and uses the insertion-based scheduling;
7   **end**
8   denote the core with the minimum *EFT* of task τ_i by C_r;
9   **while** ***true*** **do**
10   **if** $Temperature(\tau_i, C_r) \leq T_{\mathrm{th}}$ **then**
11   allocate task τ_i to core C_r;
12   **break**;
13   **end**
14   **else**
15   denote the core with the next minimum *EFT* of task τ_i by C_r;
16   **end**
17   **end**
18 **end**
19 calculate the system SER using Eq. (8);
20 **if** $SER_{\mathrm{sys}} < SER_{\mathrm{th}}$ **then**
21   **output** "Infeasible schedule";
22 **end**

Alg. 2 presents the psuedo code for EFT-LS. The algorithm first derives the $rank_{\mathrm{EFT}}$ for all tasks using Eq. (26) (line 1) and sorts these tasks in the non-ascending order of $rank_{\mathrm{EFT}}$ values using function $Rank = UpRank(\mathcal{V})$ (line 2), where *Rank* is the scheduling list of the sorted tasks and *UpRank*() computes the task rank values as well as sorts tasks based on the rank values. It then iteratively decides the allocation of each task to cores (lines 3-18). During each iteration, the algorithm calculates the EFTs of the currently-scheduled task if it is executed at frequency $F_{\max}$ and scheduled on M cores (lines 4-8), and allocates the task to the core with a minimum EFT if not violating the MTTF constraint (lines 10-13). It has been shown in [37] that the system LTR constraint can be ensured by checking whether the operating temperature of multicore system exceeds a corresponding threshold. Thus, if the operating temperature of executing task τ_i on core C_r, represented by $temperature(\tau_i, C_r)$, does-not exceed a threshold T_{th}, task τ_i is then allocated to core C_r. Otherwise, the algorithm attempts to allocate the task to the core with the next minimum EFT and checks the temperature constraint. The outer while-loop is terminated if the allocation of all tasks in the list *Rank* have been determined. The algorithm finally checks the system SER constraint. If the system SER SER_{sys} estimated by Eq. (8) is lower than the SER requirement SER_{th}, the derived task schedule is deemed as infeasible (lines 19-22).

C. CROSS ENTROPY BASED TASK SCHEDULING FOR LOW ENERGY STATE

As introduced above, the proposed EFT-LS algorithm can maximize system throughput by minimizing the latency of executing tasks in the application. However, EFT-LS is only effective for systems with plentiful energy supply and thus cannot tackle the studied throughput maximization problem for systems with insufficient energy supply. Therefore, we develop a cross-entropy (CE) based approach to maximize the throughput of systems with insufficient energy supply. The CE approach is a versatile strategy used for solving NP-hard optimization problems [38].

For a deterministic optimization problem to be solved, the CE approach converts it into an associated stochastic optimization problem and considers the optimum solution to the stochastic problem as a rare event. The approach finds the optimum solution by continuously increasing the probability of the rare event using an iterative sampling scheme that gradually changes the sampling distribution. When the probability of the rare event approaches 1, the optimum solution to the original deterministic problem is found. During the iterative process, each solution is viewed as a sample. Solution samples are produced according to the probability density function (PDF) and then converged in a probabilistic manner to better solution samples. For more details of the CE approach, the readers are recommended to refer to the literature [38].

For a better understanding, we briefly illustrate the philosophy behind the CE approach in Fig. 3. During each iteration of the CE approach, solution samples are produced following the PDF and the quality of these solutions are evaluated. We identify those high-quality samples as elite samples, and rely on them to update the PDF's characterizing parameter. The updated PDF will be utilized during the next iteration for producing a new generation of samples.

Alg. 3 shows the pseudo code of our CE-based heuristic. In the initialization step, we initialize the mean value as well as standard deviation of Gaussian distribution which will be adopted to produce samples. In this step, iteration count is also initialized (lines 1-2). The algorithm iteratively derives

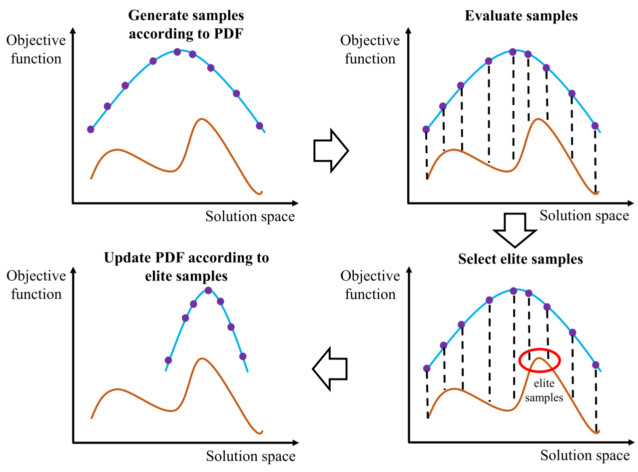


FIGURE 3. Workflow of cross entropy optimization approach [39].

Algorithm 3 CE-Based Task Scheduling

---

1 initialize the mean (ζ_1) and variation (ξ_1) of Gaussian distribution; `/* Gaussian distribution is adopted as the PDF */`

2 $g = 1$; `/* initialize the counter of iterations */`

3 **repeat**

4   produce $\mathcal{W}$ solution samples $[\mathcal{S}_1, \mathcal{S}_2, \cdots, \mathcal{S}_{\mathcal{W}}]$ using Latin hypercube sampling method according to distribution $N(\zeta_g, \xi_g)$;

5   select $\mathcal{J}$ ($\mathcal{J} < \mathcal{W}$) feasible solution samples meeting the constraints in Eqs. (15)-(25) using acceptance-rejection scheme;

6   calculate the Tru_{sys} value for each selected sample by Eq. (14);

7   select the top $\mathcal{Q}$ elite samples with respect to Tru_{sys};

8     $g++$ and update ζ_g and ξ_g;

9 **until** *iteration converges or iteration count reaches its maximum value*;

---

the best solution leading to shortest task schedule from lines 3 to 9. Specifically, in each iteration the algorithm generates $\mathcal{W}$ voltage samples following the Gaussian distribution $N(\zeta_g, \xi_g)$ (line 4) and chooses $\mathcal{J}$ feasible samples which meet all the design constraints in Eqs. (15)-(25) by using the acceptance-rejection scheme (line 5). The algorithm then evaluates the selected samples in terms of system throughput Tru_{sys} derived by Eq. (14), and choose the top $\mathcal{Q}$ elite samples to update the distribution PDF (lines 6-7). At the end of each iteration, g, ζ_g, and ξ_g are updated (line 8). The iteration is terminated if the predefined convergence criteria is met or the iteration count reaches a pre-specified limit (line 9). Since the evaluation procedure for the solution samples are independent of each other, we can strikingly accelerate the heuristic algorithm by running the algorithm in the parallel programming environment.

TABLE 2. Characteristics of real-world DAGs [40].

DAG Benchmark	Number of Nodes	Number of Edges
CyberShake_30	30	112
CyberShake_50	50	188
CyberShake_100	100	380
Inspiral_30	30	95
Inspiral_50	50	160
Inspiral_100	100	319
Montage_25	25	95
Montage_50	50	206
Montage_100	100	433
Sipht_30	30	91
Sipht_60	60	198
Sipht_100	100	335

VI. EVALUATION

This section first describes the simulation setups used for validating the proposed MILP and ESA approaches, then presents and analyzes the simulation results.

A. SIMULATION SETUPS

Four real-world DAG benchmarks including CyberShake, Inspiral, Montage, as well as Sipht [40] are utilized in the simulations to verify the effectiveness of the proposed scheme. These DAG benchmarks are widely used in evaluating the performance of task scheduling algorithms. TABLE 2 summarizes the key characteristics of these benchmarks. The simulations are performed based on 2×3 ($M = 6$) and 2×4 ($M = 8$) multicore systems. The multicore model is built upon a ARM Cortex platform. Each core supports three levels of frequency and voltage, i.e., 300MHz/1.06V, 600MHz/1.1V, and 900MHz/1.2V. We use solar energy as the renewable generation, since it is the most easy-to-access energy source and it can derived based on the harvesting power trace [4]

$$P_{\text{harv}}(t) = \left|\Lambda \times \Psi(t) \times \cos(\frac{t}{70\pi}) \times \cos(\frac{t}{100\pi})\right|, \quad (28)$$

where Λ is a constant coefficient, and $\Psi(t)$ is a zero-mean, unit-variance random variable. The raw failure rate at the maximum frequency $\lambda_{F_{\max}}$ is set to 1.0×10^{-4} [28]. The task vulnerability factor, VF_i, is randomly selected within the range (0, 1]. We use the same setups (e.g., $\alpha = 2$, the current density is $1.5 \times 10^6 \, A/cm^2$, the activation energy is $0.48eV$, derive the temperature using HotSpot [41]) as in [34] to predict core's aging rate A_m and in turn core's MTTF.

We perform two separate sets of comparative experiments to fully verify the proposed MILP approach and ESA scheme. In the first set of experiments, we compare the proposed MILP approach with the baseline method Rand and our scheme ESA in terms of improving system throughput. Rand is a method that randomly determines the allocation and frequency of tasks under the energy supply constraint. In the second set of experiments, we compare the proposed scheme ESA with the benchmarking methods TATS [23], WARM [22], TMTC [21], and AFTS [42] in terms of

TABLE 3. System throughput of 12 benchmarks achieved by the MILP approach, the proposed scheme ESA, and baseline method Rand.

DAG	6-cores					8-cores				
	Rand	MILP		ESA		Rand	MILP		ESA	
Benchmark	Tru_{sys}	Tru_{sys}	IMP	Tru_{sys}	IMP	Tru_{sys}	Tru_{sys}	IMP	Tru_{sys}	IMP
CyberShake_30	18	26	44.4%	24	33.3%	20	28	40.0%	26	30.0%
CyberShake_50	28	43	53.6%	40	42.9%	31	46	48.4%	43	38.7%
CyberShake_100	65	88	35.4%	85	30.8%	70	91	30.0%	88	25.7%
Inspiral_30	16	27	68.8%	25	56.3%	19	28	47.4%	26	36.8%
Inspiral_50	29	45	55.2%	41	41.4%	33	47	42.4%	44	33.3%
Inspiral_100	58	89	53.4%	83	43.1%	65	92	41.5%	87	33.8%
Montage_25	12	22	83.3%	20	66.7%	15	24	60.0%	22	46.7%
Montage_50	29	46	58.6%	41	41.4%	32	48	50.0%	45	40.6%
Montage_100	63	91	44.4%	88	39.7%	69	93	34.8%	90	30.4%
Sipht_30	14	27	92.9%	23	64.3%	18	28	55.6%	25	38.9%
Sipht_60	32	55	71.9%	52	62.5%	38	57	50.0%	54	42.1%
Sipht_100	49	90	83.7%	85	73.5%	57	93	63.2%	89	56.1%
Average	34.4	54.1	62.1%	50.6	49.7%	38.9	56.3	46.9%	53.3	37.8%

increasing system throughput and ensuring the schedule feasibility. The comparative algorithms are described below.

- TATS [23] is a particle swarm optimization and simulated annealing based algorithm that exploits the temporal variation to schedule tasks of green data center for maximizing profit under task delay constraints.
- WARM [22] is a throughput-aware approach that maximizes the revenue of green data center providers by reducing the scheduling cost of all arrival tasks.
- TMTC [21] is a temperature-constrained optimization method that maximizes system throughput by minimizing the execution latency of tasks.
- AFTS [42] is a method that achieves energy efficiency and fault-tolerance simultaneously for real-time EHS using the techniques of DVFS and primary backup.

B. VALIDATE THE PROPOSED MILP APPROACH

We use a common MILP solver, CPLEX with AMPL, to address the instances of the proposed MILP formulation for maximizing system throughput under the constraints of energy as well as reliability. As discussed in Section IV-C, the MILP solver may not address the optimization problem for systems of a larger granularity efficiently. Generally, MILP can generate the optimum throughput for small systems. However, for most small systems, MILP may fail in finding the optimum solutions in several hours. Therefore, we terminate the MILP solver after six hours and adopt the best results generated by the solver. Considering that the baseline method Rand doesnot consider the reliability and task precedence constraint and hence its produced solutions may violate the constraints, we remove these invalid results for conducting a fair comparison with the proposed MILP and ESA.

TABLE 3 demonstrates the comparison of system throughput obtained by our MILP, ESA, and baseline method Rand. In the comparison, twelve benchmarks and two multicore systems are used. In the table, the Tru_{sys} column indicates the system throughput achieved by the three approaches while the "IMP" column indicates the improvement of system throughput realized by MILP and ESA over Rand. For the 6-core system, the average throughput improvement achieved by MILP and ESA over Rand are 62.1% and 49.7%, respectively. The highest improvement achieved by MILP and ESA compared to Rand can be up to 92.9% and 73.5%, respectively. For the 8-core system, the average throughput improvement achieved by MILP and ESA over Rand are 46.9% and 37.8%, respectively. The highest improvement achieved by MILP and ESA compared to Rand can be up to 63.2% and 56.1%, respectively. We also observe that MILP can maximize the system throughput among the three approaches regardless of benchmarks and multicores. However, the MILP solver always derives the solutions in hours while ESA and Rand in minutes.

C. VALIDATE THE PROPOSED ESA APPROACH

Two simulation experiments are performed to justify the efficacy of the proposed ESA approach in terms of increasing system throughput and ensuring schedule feasibility. In the first experiment, we compare the system throughput achieved by the proposed scheme ESA and four peer approaches TATS [23], WARM [22], TMTC [21], and AFTS [42]. In the second experiment, we compare the schedule feasibility realized by the proposed scheme ESA and four peer approaches TATS [23], WARM [22], TMTC [21], and AFTS [42]. The schedule feasibility is defined as the ratio of the number of applications that can be successfully scheduled to the total number of applications adopted in the test.

Fig. 4 presents the throughput of executing 12 benchmarks on 6-core and 8-core systems achieved by the proposed scheme ESA and peer approaches TATS [23], WARM [22], TMTC [21], and AFTS [42]. The results clearly show that ESA is able to achieve the highest averaged system throughput among the five methods no matter which multicore system is adopted. Specifically, the averaged throughput of 6-core system achieved by ESA, TATS [23], WARM [22], TMTC [21], and AFTS [42] over the 12 benchmarks are

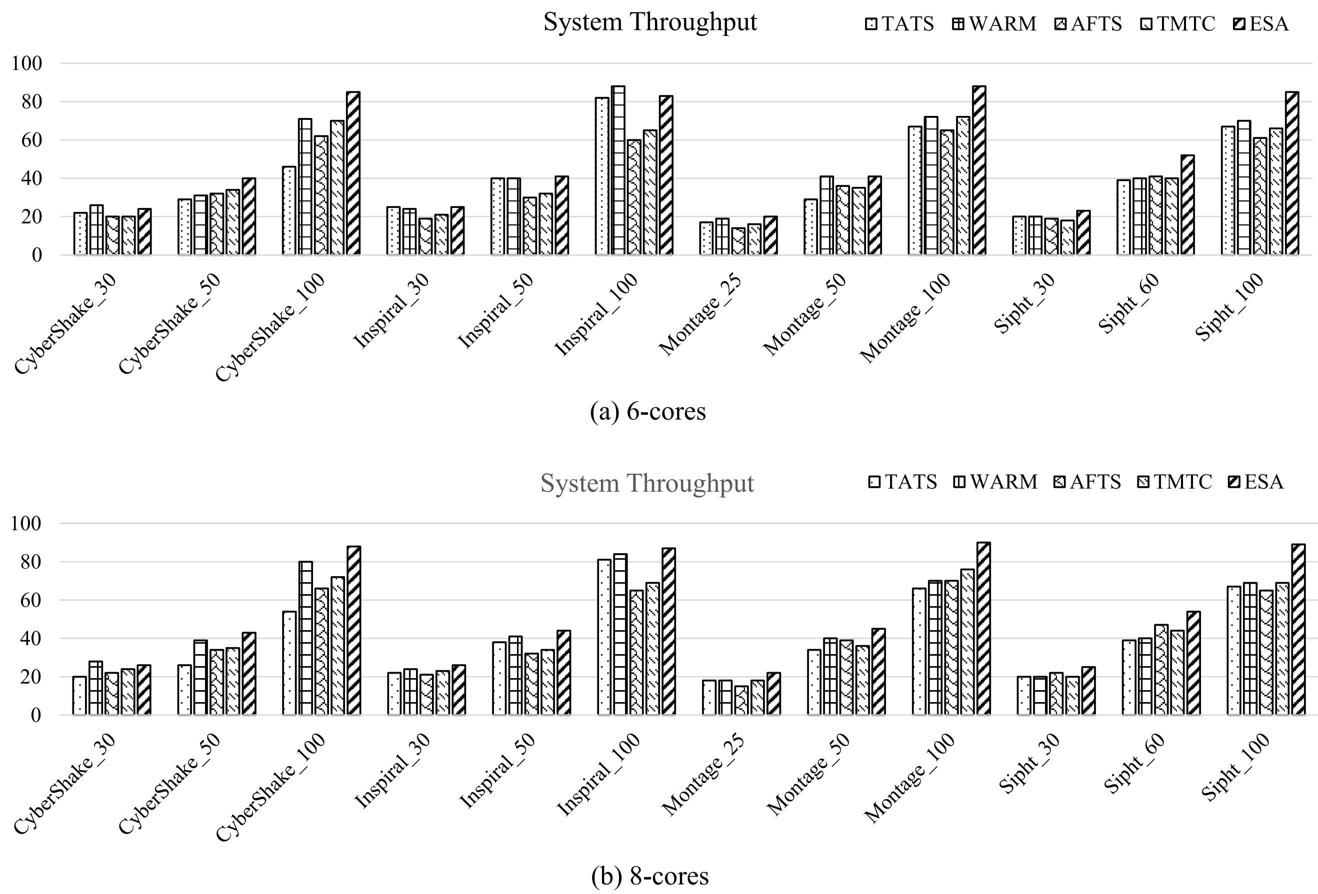


FIGURE 4. **The system throughput of executing 12 benchmarks using the proposed scheme ESA and peer approaches TATS [23], WARM [22], TMTC [21], and AFTS [42].**

50.6, 40.3, 45.2, 40.8, and 38.3, respectively. The averaged throughput of 8-core system using ESA, TATS [23], WARM [22], TMTC [21], and AFTS [42] are 53.3, 40.4, 46.1, 43.3, and 37.6, respectively. The averaged system throughput using ESA can be up to 32.1% higher than those of TATS [23], WARM [22], TMTC [21], and AFTS [42]. The reason why ESA outperforms the four peer approaches is that ESA considers the uncertainty in energy sources and determines the allocation and scheduling of tasks based on the system energy state. From the figure, we can also deduce that the throughput of 8-core system is almost higher than that of 6-core system. This is because that adding more cores is benefit to the scheduling of tasks in the benchmark.

Fig. 5 shows the schedule feasibility of executing 12 benchmarks on 6-core and 8-core systems using the proposed scheme ESA and peer approaches TATS [23], WARM [22], TMTC [21], and AFTS [42]. As can be seen in the figure, ESA can ensure a much higher feasibility as compared to TATS [23], WARM [22], TMTC [21], and AFTS [42]. For example, the feasibility achieved by ESA, TATS [23], WARM [22], TMTC [21], and AFTS [42] for the 6-core system are 83.3%, 66.7%, 75.0%, 58.3%, and 66.7%, respectively. This is because that neither of TATS [23], WARM [22], TMTC [21], and AFTS [42] considers the energy and reliability constraints simultaneously. In addition, we can easily

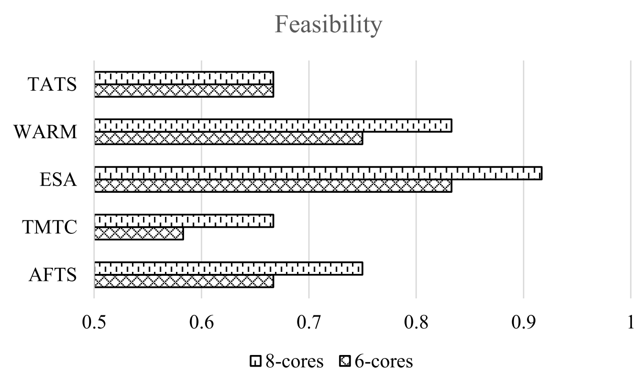


FIGURE 5. **The schedule feasibility of executing 12 benchmarks using the proposed scheme ESA and peer approaches TATS [23], WARM [22], TMTC [21], and AFTS [42].**

find that the feasibility of executing benchmarks on the 8-core system using the five methods are almost higher than that of the 6-core system, which benefits from the larger scheduling space brought by the increased core number.

VII. CONCLUSION

In this paper, we have solved the task allocation and scheduling problem for maximizing the throughput of multicore energy-harvesting systems. To optimize the throughput of systems powered by intermittent renewable energy, as well as to satisfy the reliability and task precedence constraints, we designed an MILP approach and an energy state-aware approach. The MILP approach is able to find the optimal solution but it may take exponential time to finish while the energy state-aware approach is a polynomial-time heuristic that can derive the sub-optimal solutions efficiently. We performed extensive simulations to validate the proposed MILP and energy state-aware approaches. Simulation results show that the proposed MILP approach has the best performance in increasing system throughput. The system throughput can be increased by up to 92.9% using the MILP approach as compared to a baseline method. The evaluation results also demonstrate that the proposed heuristic algorithm outperforms four benchmarking methods from the perspectives of system throughput and schedule feasibility.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their helpful suggestions.

REFERENCES

[1] K. Lampka, K. Huang, and J.-J. Chen, "Dynamic counters and the efficient and effective online power management of embedded real-time systems," in *Proc. 9th Int. Conf. Hardw./Softw. Codes. Syst. Synth.*, Oct. 2011, pp. 267–276.

[2] G. Xie, H. Peng, Z. Li, J. Song, Y. Xie, R. Li, and K. Li, "Reliability enhancement toward functional safety goal assurance in energy-aware automotive cyber-physical systems," *IEEE Trans. Ind. Informat.*, vol. 14, no. 12, pp. 5447–5462, Dec. 2018.

[3] G. Xie, J. Huang, Y. L. R. Li, and K. Li, "System-level energy-aware design methodology towards end-to-end response time optimization," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, to be published. doi: 10.1109/TCAD.2019.2921350.

[4] J. Chen, T. Wei, and J. Liang, "State-aware dynamic frequency selection scheme for energy-harvesting real-time systems," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 22, no. 8, pp. 1679–1692, Aug. 2014.

[5] J. Zhou, J. Chen, K. Cao, T. Wei, and M. Chen, "Game theoretic energy allocation for renewable powered *in-situ* server systems," in *Proc. 22nd Int. Conf. Parallel Distrib. Syst.*, Dec. 2016, pp. 721–728.

[6] "Offshore oil and gas supply," Work. Document Nat. Petroleum Council, 2011.

[7] *How Big Data is Changing Astronomy (Again)*, The Atlantic, Washington, DC, USA, 2012.

[8] "The changing geospatial landscape," Rep. Nat. Geospatial Advisory Committee, 2009.

[9] W. A. Cox, M. S. Pruett, T. J. Benson, S. J. Chiavacci, and F. R. Thompson, III, "Development of camera technology for monitoring nests," in *Video Surveillance of Nesting Birds* (Studies in Avian Biology), no. 43, C. A. Ribic, F. R. Thompson, III, and P. J. Pietz, Eds. Berkeley, CA, USA: Univ. of California Press, 2012, pp. 185–210.

[10] C. Li, Y. Hu, L. Liu, J. Gu, M. Song, X. Liang, J. Yuan, and T. Li, "Towards sustainable *in-situ* server systems in the big data era," in *Proc. Int. Symp. Comput. Archit.*, Jun. 2015, pp. 14–26.

[11] T. Esram and P. L. Chapman, "Comparison of photovoltaic array maximum power point tracking techniques," *IEEE Trans. Energy Convers.*, vol. 22, no. 2, pp. 439–449, Jun. 2007.

[12] K. Kobayashi, I. Takano, and Y. Sawada, "A study on a two stage maximum power point tracking control of a photovoltaic system under partially shaded insolation conditions," *Solar Energy Mater. Solar Cells*, vol. 90, nos. 18–19, pp. 2975–2988, 2006.

[13] V. Sharma, U. Mukherji, V. Joseph, and S. Gupta, "Optimal energy management policies for energy harvesting sensor nodes," *IEEE Trans. Wireless Commun.*, vol. 9, no. 4, pp. 1326–1336, Apr. 2010.

[14] Y. Abdeddaim and D. Masson, "Real-time scheduling of energy harvesting embedded systems with timed automata," in *Proc. Int. Conf. Embedded Real Time Comput. Syst. Appl.*, Aug. 2012, pp. 31–40.

[15] D. Liu, Y. Xu, Q. Wei, and X. Liu, "Residential energy scheduling for variable weather solar energy based on adaptive dynamic programming," *IEEE/CAA J. Autom. Sinica*, vol. 5, no. 1, pp. 36–46, Jan. 2018.

[16] K. Tutuncuoglu, B. Varan, and A. Yener, "Throughput maximization for two-way relay channels with energy harvesting nodes: The impact of relaying strategies," *IEEE Trans. Commun.*, vol. 63, no. 6, pp. 2081–2093, Jun. 2015.

[17] S. Gupta, R. Zhang, and L. Hanzo, "Throughput maximization for a buffer-aided successive relaying network employing energy harvesting," *IEEE Trans. Veh. Technol.*, vol. 65, no. 8, pp. 6758–6765, Aug. 2016.

[18] A. Mehrabi and K. Kim, "General framework for network throughput maximization in sink-based energy harvesting wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 16, no. 7, pp. 1881–1896, Jul. 2017.

[19] W. Wu, J. Wang, X. Wang, F. Shan, and J. Luo, "Online throughput maximization for energy harvesting communication systems with battery overflow," *IEEE Trans. Mobile Comput.*, vol. 16, no. 1, pp. 185–197, Jan. 2017.

[20] G. Wang, Q. Bian, H. Xin, and Z. Wang, "A robust reserve scheduling method considering asymmetrical wind power distribution," *IEEE/CAA J. Autom. Sinica*, vol. 5, no. 5, pp. 961–967, Sep. 2018.

[21] H. Huang, V. Chaturbedi, G. Quan, J. Fan, and M. Qiu, "Throughput maximization for periodic real-time systems under the maximal temperature constraint," *ACM Trans. Embedded Comput. Syst.*, vol. 13, no. 2s, p. 70, Jan. 2014.

[22] H. Yuan, J. Bi, M. Zhou, and K. Sedraoui, "WARM: Workload-aware multi-application task scheduling for revenue maximization in SDN-based cloud data center," *IEEE Access*, vol. 6, pp. 645–657, 2017.

[23] H. Yuan, J. Bi, M. Zhou, and A. C. Ammari, "Time-aware multi-application task scheduling with guaranteed delay constraints in green data center," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 3, pp. 1138–1151, Jul. 2018.

[24] H. Yuan, J. Bi, and M. Zhou, "Spatial task scheduling for cost minimization in distributed green cloud data centers," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 2, pp. 729–740, Apr. 2019.

[25] 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.*, Jan. 2016, pp. 685–690.

[26] Y. Ma, T. Chantem, R. P. Dick, S. Wang, and X. S. Hu, "An on-line framework for improving reliability of real-time systems on 'big-little' type MPSoCs," in *Proc. Design, Automat. Test Eur. Conf. Exhib. (DATE)*, Mar. 2017, pp. 446–451. doi: 10.23919/DATE.2017.7927031.

[27] T. Kim, Z. Sun, H.-B. Chen, H. Wang, and S. X.-D. Tan, "Energy and lifetime optimizations for dark silicon manycore microprocessor considering both hard and soft errors," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 9, pp. 2561–2574, Sep. 2017.

[28] J. Zhou, T. Wei, M. Chen, X. S. Hu, Y. Ma, G. Zhang, and J. Yan, "Variation-aware task allocation and scheduling for improving reliability of real-time MPSoCs," in *Proc. Design, Automat. Test Eur. Conf. Exhib. (DATE)*, Mar. 2018, pp. 171–176.

[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.*, to be published. doi: 10.1109/TCAD.2018.2883993.

[30] G. Xie, Y. Chen, R. Li, and K. Li, "Hardware cost design optimization for functional safety-critical parallel applications on heterogeneous distributed embedded systems," *IEEE Trans. Ind. Informat.*, vol. 14, no. 6, pp. 2418–2431, Jun. 2018.

[31] G. Xie, G. Zeng, Y. Chen, Y. Bai, Z. Zhou, R. Li, K. Li, "Minimizing redundancy to satisfy reliability requirement for a parallel application on heterogeneous service-oriented systems," *IEEE Trans. Service Comput.*, to be published. doi: 10.1109/TSC.2017.2665552.

[32] 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 Design*, Nov. 2004, pp. 35–40.

[33] J. Srinivasan, S. V. Adve, P. Bose, and J. A. Rivers, "The impact of technology scaling on lifetime reliability," in *Proc. Int. Conf. Dependable Syst. Netw.*, Jun. 2004, pp. 177–186.

[34] L. Huang, F. Yuan, and Q. Xu, "Lifetime reliability-aware task allocation and scheduling for MPSoC platforms," in *Proc. Conf. Design, Autom. Test Eur.*, Apr. 2009, pp. 51–56.

[35] T. Chantem, Y. Xiang, X. S. Hu, and R. P. Dick, "Enhancing multicore reliability through wear compensation in online assignment and scheduling," in *Proc. Design, Automat. Test Eur. Conf. Exhib. (DATE)*, Mar. 2013, pp. 1373–1378.

[36] H. Topcuoglu, S. Hariri, and M.-Y. Wu, "Performance-effective and low-complexity task scheduling for heterogeneous computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 13, no. 3, pp. 260–274, Mar. 2002.

[37] Y. Ma, J. Zhou, T. Chantem, R. P. Dick, S. Wang, and X. S. Hu, "On-line resource management for improving reliability of real-time systems on 'big–little' type MPSoCs," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, to be published. doi: 10.1109/TCAD.2018.2883990.

[38] R. Rubinstein and D. Kroese, "The cross entropy method: A unified approach to combinatorial optimization, monte-carlo simulation, and machine learning," in *Annals of Operations Research*, 2004 ed. Springer, Jul. 2004, p. 301.

[39] X. Chen, Y. Chen, A. Y. Zomaya, R. Ranjan, and S. Hu, "CEVP: Cross entropy based virtual machine placement for energy optimization in clouds," *J. Supercomput.*, vol. 72, no. 8, pp. 3194–3209, Aug. 2016.

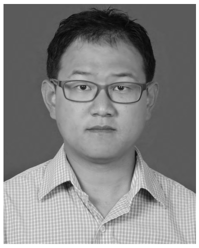
[40] G. Juve, A. Chervenak, E. Deelman, S. Bharathi, G. Mehta, and K. Vahi, "Characterizing and profiling scientific workflows," *Future Gener. Comput. Syst.*, vol. 29, no. 3, pp. 682–692, 2013.

[41] 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.

[42] L. Zhu, T. Wei, X. Chen, Y. Guo, and S. Hu, "Adaptive fault-tolerant task scheduling for real-time energy harvesting systems," *J. Circuits, Syst., Comput.*, vol. 21, no. 1, Feb. 2012, Art. no. 1250004.

JUNLONG ZHOU (S'15–M'17) received the Ph.D. degree in computer science 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 Assistant Professor with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China. He has published more than 40 refereed papers in his research areas, most of which are published in premium conferences and journals, including the IEEE/ACM DATE, the IEEE TPDS, the IEEE TCAD, the IEEE TCAS, the IEEE TR, the IEEE TAES, JSS (Elsevier), JSA (Elsevier), and FGCS (Elsevier). His research interests include real-time embedded systems, cloud computing, and cyber physical systems. He received the Reviewer Award from the *Journal of Circuits, Systems, and Computers*, in 2016. He has served as the Publication Chair, the Publicity Chair, the Section Chair, and the TPC member for numerous conferences. He has been an Associate Editor for the *Journal of Circuits, Systems, and Computers*, and serves as a Guest Editor for several special issues of the *ACM Transactions on Cyber-Physical Systems*, *IET Cyber-Physical Systems: Theory & Applications*, and the *Journal of Systems Architecture: Embedded Software Design*.

PEIJIN CONG received the B.S. degree from the Department of Computer Science and Technology, East China Normal University, Shanghai, China, in 2016, where she is currently pursuing the Ph.D. degree. She has published 10 papers in premium journals, such as the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, the IEEE TRANSACTIONS ON SUSTAINABLE COMPUTING, and the IEEE COMMUNICATIONS SURVEYS AND TUTORIALS. Her current research interests include the areas of cloud computing and the Internet of Things.



JIN SUN (M'17) received the B.S. and M.S. degrees in computer science from the Nanjing University of Science and Technology, Nanjing, China, in 2004 and 2006, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Arizona, in 2011. From January 2012 to September 2014, he was with Orora Design Technologies, Inc., as a member of Technical Staff. He is currently an Associate Professor with the School of Computer Science and Engineering, Nanjing University of Science and Technology. His research interests include integrated circuit modeling and analysis and computer-aided design. He has been an Associate Editor for the *Journal of Circuits, Systems, and Computers*, since 2018.

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 include the areas of embedded systems, cloud computing, and distributed systems.

TONGQUAN WEI (M'11–SM'19) received the Ph.D. degree in electrical engineering from Michigan Technological University, Houghton, MI, USA, in 2009. He is currently an Associate Professor with the Department of Computer Science and Technology, East China Normal University, Shanghai, China. He has published over 60 papers in these areas, most of which are published in premium conferences and journals, including the IEEE/ACM ICCAD, the IEEE/ACM DATE, the IEEE TPDS, the IEEE TC, the IEEE TCAD, the IEEE TVLSI, the IEEE TCAS, the IEEE TSG, the IEEE TDSC, the IEEE TAES, the IEEE TRL, the IEEE TSUSC, and the IEEE COMST. His current research interests include real-time embedded systems, green and reliable computing, parallel and distributed systems, and cloud computing. He has been a Regional Editor for the *Journal of Circuits, Systems, and Computers*, since 2012. He served as a Guest Editor for several special sections of the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, the *ACM Transactions on Embedded Computing Systems*, and the *ACM Transactions on Cyber-Physical Systems*.

MINGSONG CHEN (S'08–M'11–SM'17) received the B.S. and M.E. degrees from the Department of Computer Science and Technology, Nanjing University, Nanjing, China, in 2003 and 2006, respectively, and the Ph.D. degree in computer engineering from the University of Florida, Gainesville, FL, USA, in 2010. He is currently a Full Professor with the Department of Embedded Software and Systems, East China Normal University, Shanghai, China. He has published over 70 papers in these areas, most of which are published in premium conferences and journals, including ICCAD, DAC, DATE, ICSE, CODES+ISSS, the IEEE TPDS, the IEEE TC, the IEEE TCAD, the IEEE TCC, the IEEE TSC, the IEEE TRL, and ACM TODAES. His current research interests include cyber-physical systems, formal verification, and mobile cloud computing. He has served as the TPC Chair, the Publication Chair, the Publicity Chair, the Section Chair, and the TPC member for numerous conferences. He is also an Associate Editor of *IET Computers and Digital Techniques* and the *Journal of Circuits, Systems, and Computers*. He serves as a Guest Editor for a special section of the IEEE TRANSACTIONS ON RELIABILITY.