JKSCI

한국컴퓨터정보학회논문지
Journal of The Korea Society of Computer and Information
Vol. 27 No. 12, pp. 17-27, December 2022
https://doi.org/10.9708/jksci.2022.27.12.017

An Extended ED-H Real-Time Scheduling Algorithm for Supporting an Intelligent PMU-Based Energy Harvesting System

Sangsoo Park*

*Professor, Dept. of Computer Science & Engineering, Ewha Womans University, Seoul, Korea

[Abstract]

In this paper, ED-H algorithm, an optimal real-time scheduling algorithm dealing with the characteristics of the integrated energy harvester system with a capacitor, is extended to satisfy the time constraint under the blackout state which is a deliberate power-off state by an intelligent power management unit adopted in the system. If the power supply system does not have enough energy, it temporarily shuts off the power supply to protect the circuit and capacitor and resumes the supply again when the capacitor is fully charged, which may delay the task execution during these blackout states by calculating the time according to the occurrence of the events. To mitigate the problem, even if task execution is delayed by the original ED-H algorithm, the remaining time of the subsequent time units no longer can afford to delay the execution of the task is predicted in the extended algorithm and the task is forced to be scheduled to meet the time deadline. According to the simulation results, it is confirmed that the algorithm proposed in this paper has a high scheduling performance increase of 0.4% to 7.7% depending on the characteristics of the set of tasks compared to the ED-H.

▸ **Key words**: Real-time scheduling, Energy harvesting, Capacitor, ED-H, Intelligent Power Management Unit

[요 약]

본 논문은 커패시터와 에너지 하베스터의 특성이 적용된 최적의 실시간 스케줄링 알고리즘인 ED-H 알고리즘을 적용할 때 적용되는 지능형 전원 관리 유닛에 의한 의도적인 전원 공급 차단 상태인 블랙아웃에서 시간 제약성을 만족하도록 알고리즘을 확장한다. 전원 공급 시스템에서 생산되는 에너지가 충분하지 않은 경우 회로와 커패시터를 보호하기 위해 전력 공급을 일시적으로 차단하고 커패시터가 충분히 충전되었을 때 다시 공급을 재개하므로 이러한 블랙아웃 시간 동안 태스크의 수행이 지연될 수 있으므로 해당 이벤트의 발생에 따른 시간을 계산하였다. 이러한 문제점을 극복하기 위해 원형 ED-H 알고리즘에 의해 태스크 수행이 지연되는 경우라 하더라도 본 논문에서 제안된 알고리즘은 뒤에 이어지는 시간 유닛 중에서 더 이상 태스크의 수행을 지연시킬 여력이 없는 경우, 즉 태스크를 이번 시간 유닛에 수행하지 않으면 시간 제약성을 위배하게 되는 경우의 잔여 시간을 계산하여 잔여 시간이 부족한 경우 시간 제약성을 만족하도록 강제로 스케줄링한다. 시뮬레이션을 통해 본 눈문에서 제안한 알고리즘이 ED-H 알고리즘에 대해 태스크 집합의 유형 특성에 따라 0.4%~7.7%까지 스케줄링 성능이 높은 것을 확인하였다.

▸ **주제어**: 실시간 스케줄링, 에너지 하베스팅, 커패시터, ED-H, 지능형 전원 관리 유닛

• First Author: Sangsoo Park, Corresponding Author: Sangsoo Park
*Sangsoo Park (sangsoo.park@ewha.ac.kr), Dept. of Computer Science & Engineering, Ewha Womans University
• Received: 2022. 11. 14, Revised: 2022. 11. 30, Accepted: 2022. 12. 05.

http://www.ksci.re.kr pISSN:1598-849X | eISSN:2383-9945

I. Introduction

저전력 반도체 및 소형 하드웨어의 발전에 따라 무선 통신을 통해 소량의 데이터를 송수신하는데 특화된 소물인터넷(IoST: Internet-of-Small-Things) 기술 기반의 원격 센싱 기술이 보안 관제, 지능형 교통망, 헬스케어 등 다양한 분야에 적용되고 있다[1-3].

소물인터넷을 위한 하드웨어는 전원 공급을 위해 주로 리튬이온 기반의 충전식 배터리를 탑재하고 있으나, 일시적으로 전원 공급 출력이 높아지는 경우 전압 강하가 발생하여 동작이 멈추거나 오동작할 수 있으며, 충전과 방전의 횟수가 늘어날수록 수명이 단축되는 특성이 있다[4]. 이러한 단점을 극복하기 위해 자연계에 존재하는 운동(kinetic), 태양광, 전파 에너지 등을 전원 공급의 소스로 사용하는 에너지 하베스팅 기술이 전원 소스의 물리적인 특성인 지속적으로 안정적인 품질의 전원 공급이 어려운 점을 보완하기 위해 보조 전원장치인 슈퍼 커패시터와의 결합이 확산되고 있다[5][6]. 슈퍼 커패시터는 충전식 배터리의 성능을 보완하는 전력 저장 장치로 빠른 충전과 방전이 가능하며 높은 전력 효율과 반영구적인 충방전 사이클 수명을 갖고 있어 에너지 하베스팅 기기와 결합 되어 사용되는 전력 장치로서 주목받고 있다[7].

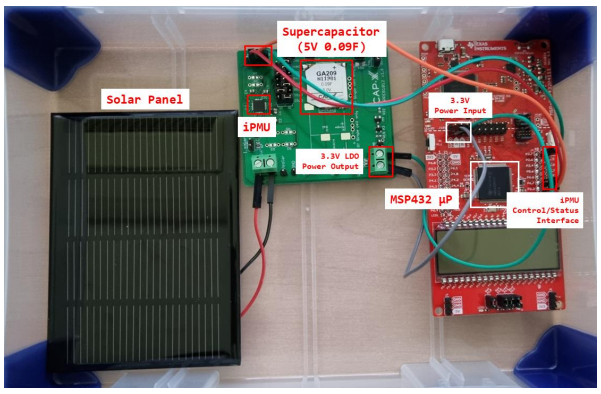


Fig. 1. A system example for a supercapacitor integrated solar energy harvesting system equipped with iPMU [8]

다양한 형태의 에너지 하베스트 시스템 중에서 가장 널리 쓰이는 태양광의 경우 물리적인 특성과 태양광 패널에 의한 전기 에너지의 생산이 비선형 전기적 특성을 갖기 때문에 이를 제어하기 위한 디지털 회로와 마이크로컨트롤러에서 요구하는 전압으로 변환하는 레귤레이터가 적용되는 것이 일반적이다.

슈퍼 커패시터와 결합되는 경우 기후의 영향으로 전력 공급에 변동이 생길 때 전력을 공급받는 마이크로컨트롤러에 상태 정보를 제공하는 인터페이스를 제공해야 한다[9]. 특히 전력 공급이 불충분할 경우 센서 등 일부 하드웨어가 동작하지 않거나, 소프트웨어 코드의 수행이 일시적으로 지연되고 메모리의 데이터가 유실될 수 있기 때문에 그림 1의 예와 같이 이를 지능적으로 제어할 수 있는 지능형 전원 관리 유닛인 iPMU(intelligent Power Management Unit)와 결합 되어 사용되는 경우가 일반적이다[10].

iPMU는 태양광 패널에 의해 공급되는 전기 에너지를 슈퍼 커패시터에 충전하면서 동시에 마이크로컨트롤러에 필요한 전력을 공급하도록 제어해주고 이것이 동시에 가능하지 않을 경우 회로와 슈퍼 커패시터를 보호하기 위해 마이크로컨트롤러에 대한 전력 공급을 일시적으로 차단하고 이후 슈퍼 커패시터가 충분히 충전되었을 때 마이크로컨트롤러에 전력 공급을 재개하는 지능화된 전원 관리 기법을 제공한다[11]. 본 논문에서는 이와 같이 iPMU에 의해 의도적으로 그리고 일시적으로 마이크로컨트롤러에 대한 전원이 차단되는 상태를 블랙아웃 상태라고 명명한다.

또한 소물인터넷이 적용되는 응용에서 스트리밍 데이터가 사용되는 경우 데이터의 수집, 처리, 전송 혹은 그 반대로 데이터의 수신, 처리, 동작하는 기능을 담당하는 프로그램 태스크 사이에 시간 제약성이 존재하며 데이터를 처리하는 태스크들은 정해진 주기 이내에 수행이 완료되어야 하므로 여러 태스크를 우선순위에 따라 처리하는 태스크 스케줄링 알고리즘이 필요하다[12].

전원 공급에 대한 고려 없이 싱글코어 프로세서를 탑재한 마이크로컨트롤러를 위한 실시간 스케줄링 알고리즘은 EDF(Earliest Deadline First) 알고리즘 등 여러 최적화된 알고리즘이 존재하고 이를 실제 하드웨어에 적용할 때 발생할 수 있는 오버헤드 및 시스템 제약 등 다양한 관련 연구가 존재한다[13][14]. 그러나, 현재까지 알려진 바에 의하면 커패시터와 태양광 기반 에너지 하베스트 시스템을 위한 최적의 실시간 스케줄링 알고리즘은 EDF 알고리즘을 기반으로 한 ED-H 알고리즘이 유일하다[15-17].

ED-H 알고리즘은 전원 공급 장치의 특성을 단순화하여 커패시터의 에너지 용량 C에 대해 단위 시간 당 충전량과 각 태스크의 전력 소비에 따른 방전량을 파라미터로 하여 알고리즘을 수립하였고 최적화된 알고리즘임을 증명하였다[16].

본 논문은 ED-H 알고리즘이 기반하고 있는 단순한 전원 공급 장치 모델에 지능형 전원 관리 유닛에 적용되었을 때 발생할 수 있는 블랙아웃 상태를 처리함으로써 원형의 ED-H 알고리즘이 태스크 스케줄링에 실패하는 경우인 블랙아웃으로 현재 시점에 반드시 실행해야 하는 태스크가

수행되지 못하여 스케줄링에 실패하는 경우가 발생할 때를 예측하여 태스크의 수행 시기를 조정하는 확장된 ED-H 알고리즘을 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 ED-H 알고리즘 및 에너지 하베스팅 시스템을 고려한 실시간 스케줄링 알고리즘과 관련된 기존 연구를 살펴보고 본 논문에서 제안된 확장된 ED-H 알고리즘과 비교한다. 그리고 ED-H 알고리즘과 본 논문의 시스템 모델과 실시간 스케줄링 알고리즘에 대한 전원 공급 모델의 필요성을 스케줄링 예를 통하여 기술한다. 3장에서는 본 논문에서 제안한 확장된 ED-H 알고리즘의 세부 설계와 구현 방법을 기술하며 시뮬레이션을 통해 원형 ED-H 알고리즘과 스케줄링 성능을 비교한다. 마지막으로 4장은 본 논문의 결론과 향후 연구 과제를 기술한다.

II. Preliminaries

1. Related Works

실시간 스케줄링 알고리즘은 마이크로컨트롤러에서 수행되는 태스크의 작업 실행을 관리하여 성능을 극대화하도록 하는데, 제한된 저장된 에너지에서 소비를 최소화하면서 태스크의 시간 제약을 만족하도록 해야 한다. 에너지 하베스팅 시스템을 실시간 스케줄링 알고리즘에 적용하는 다양한 연구는 [15]에서 정리되었다. 이러한 스케줄링 알고리즘은 첫째 동적 전압 및 동작 주파수 조정을 적용한 방법, 둘째 가용한 에너지에 따른 태스크의 분할 방법인 가용한 에너지 만큼 태스크를 일부만 수행하거나 여러 태스크를 병합하는 방법, 셋째 시간에 따른 에너지 가용 여부에 따라 태스크를 연속적이 아니라 간헐적으로 수행하는 듀티 사이클링(duty cycling) 방법으로 구분한다.

듀티 사이클링은 원격 센싱 응용에서 가장 일반적으로 사용되는 방법으로 마이크로컨트롤러가 유용한 태스크를 수행하지 않을 때 절전 모드로 전환되고 데이터 스트리밍 태스크의 수행이 필요한 시간 동안 깨어나 태스크들을 수행하는 방식으로 에너지 소비를 최소화한다. 이러한 듀티 사이클링은 ED-H 알고리즘과 본 논문에서 제안하는 확장된 알고리즘의 기본 방식으로 정의된다.

대부분의 연구들은 원격 센서 하드웨어의 듀티 사이클을 위해 충전을 통해 에너지 소스의 수학적 모델을 수립함으로써 시스템 성능을 극대화하는 방법을 제안한다[18-19].

[17]은 주기적으로 수행되는 태스크만으로 대상이 제한되는 알고리즘을 확장하여 비주기적으로 발생할 수 있는 이벤트를 주기적인 태스크와 함께 처리할 수 있는 알고리즘을 제안한다. 예측하지 못한 시기에 이벤트가 발생하면 이 이벤트를 처리했을 때 소요되는 전력과 이로 인해 지연될 수 있는 주기적인 태스크가 여전히 시간 제약성을 만족하는지 검증하는 방법을 사용한다.

[10]은 시간 제약성 준수를 위한 실시간 스케줄링 알고리즘이 아닌 듀티 사이클링과 iPMU에 중점을 두어 마이크로프로세서가 태스크를 수행 중에 iPMU에 의해 전력이 부족한 경우 혹은 충전 중인 경우 등의 상태 정보를 수신하여 전력 공급이 차단 될 때 데이터가 손실될 수 있는 휘발성 메모리의 데이터를 비휘발성 메모리로 상호 간 전환하는 스케줄링 방법을 제안하였으며, 커패시터 용량과 태스크의 소비 전력에 따른 실험 결과를 제시한다.

본 논문은 기존 연구에서 제안된 알고리즘과 달리 슈퍼커패시터와 iPMU 적용으로 인해 커패시터에 에너지가 남아 있는 경우라도 회로와 커패시터의 보호 및 듀티 사이클링 상태를 제공하기 위해 발생할 수 있는 블랙아웃 상태에서도 주기적 태스크의 시간 제약성 보장할 수 있는ED-H 알고리즘의 확장된 알고리즘을 제안하여 기존 연구와 차별성을 갖는다.

2. Background and System Model

본 논문의 알고리즘은 ED-H 알고리즘에 기반하므로, ED-H의 시스템 모델을 동일하게 사용한다. 에너지 하베스팅 시스템은 마이크로컨트롤러에서 코드의 수행을 담당하는 컴퓨팅 요소, 태스크 집합, 에너지 저장 장치인 슈퍼 커패시터, 그리고 에너지 생산 장치인 태양광 패널로 구성된다.

마이크로컨트롤러에 내장된 프로세서는 하나의 동작 주파수 혹은 동작 속도로 고정되며, 듀티 사이클링의 비활성 상태에서 무시할 수 있는 소비전력을 갖는다고 가정한다.

시스템은 n개의 서로 독립적인 주기적인 실시간 태스크로 구성되며, 태스크 집합 $\tau=\tau_i(C_i, R_i, D_i, T_i, E_i);\ i=1, \ldots, n$으로 표기한다, 이때 C_i는 태스크 τ_i의 최악의 경우 수행 시간, R_i는 τ_i의 첫 인스턴스의 시작 시간 혹은 릴리즈 시간, D_i는 마감 시간 혹은 데드라인의 절대 시간, T_i는 수행 주기, E_i는 최악의 경우 소비하는 에너지량으로 정의한다.

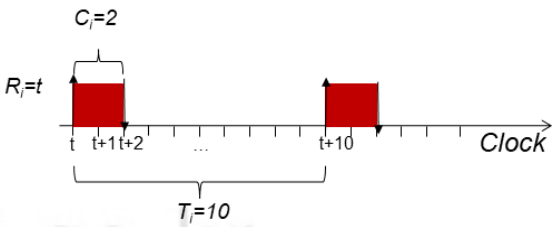


Fig. 2. A real-time task model example

시스템에서 모든 태스크는 절대 시간을 *[t, t+1)*의 범위로 나눈 시간 유닛 단위로 수행된다고 가정한다. 각 실시간 태스크 τ_i는 최초 인스턴스가 절대 시간 *t*에 릴리즈(그림 2에서 위쪽 화살표로 표기)되며 시간 유닛의 배수 단위인 수행 시간 C_i와 수행 주기 T_i를 갖는다. 매 수행 주기마다 수행 시간 C_i를 수행하는 태스크 τ_i의 새로운 태스크 인스턴스가 릴리즈 되고 이 인스턴스는 D_i 이내에 수행이 완료(그림 2에서 아래쪽 화살표로 표기)되어야 한다.

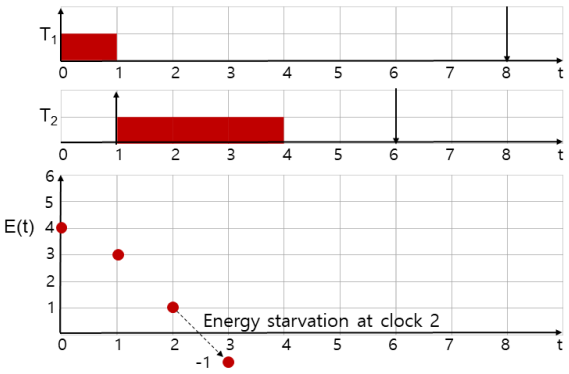
예를 들어, 그림 2에서 $C_i=2$, $R_i=t$, $T_i=10$의 시간 제약성을 갖는 태스크 τ_i는 태스크 인스턴스가 *t*, *t+10*에서 릴리즈 되어 수행을 시작한 것과 같이, 매 *10* 시간 유닛마다 릴리스 되어 다음 수행 주기 이전에 *2* 시간 유닛을 수행해야 시간 제약성을 만족한다. 단, 임의의 시간 *t*에 릴리즈 된 인스턴스가 *t*에서 바로 수행이 시작될 필요는 없다. 예를 들어, 릴리즈 된 인스턴스가 *t+5*부터 수행을 시작하여 데드라인인 *t+10*이전에 수행이 완료되면 시간 제약을 만족한다.

2.1 EDF Scheduling

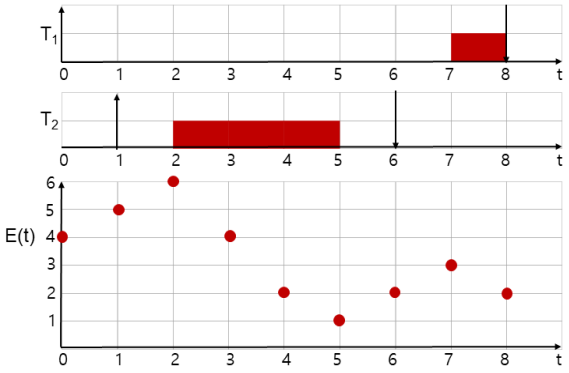
단일 프로세서 기반의 마이크로컨트롤러에서 실시간 태스크 집합 *τ*에서 임의의 태스크 τ_i의 비중을 $W_{Ti} = C_i / T_i$라고 정의했을 때 전체 실시간 태스크 집합 *τ*의 이용률은 $U = \Sigma_n W_{Ti}$로 정의된다.

EDF 스케줄링 알고리즘은 이론적으로 이용률이 *U=1*인 경우에 유휴 시간이 없이 마이크로프로세서가 항상 태스크를 수행하는 경우에도 모든 태스크의 시간 제약성을 만족하는 최적의 알고리즘이다. 시스템에서 태스크 스케줄러는 매 시간 *t*에 수행 가능한 상태인 모든 태스크의 우선순위를 결정하고 그 중에서 가장 우선 순위가 높은 태스크를 시간 유닛 *1* 만큼 수행하게 되는데, 우선 순위는 현재 시간 *t*로부터 데드라인이 가장 가까운 태스크가 가장 높은 우선순위를 갖고 반대로 데드라인이 가장 먼 태스크가 가장 낮은 우선순위를 갖는다.

이때 슈퍼커패시터와 에너지 하베스터에 의한 일시적으로 제한될 수 있는 전원 공급을 고려하면 추가적인 정의와 가정이 필요한데, 시스템에서 커패시터는 최대 *C* 만큼의 에너지를 저장할 수 있고, 에너지 하베스터는 매 시간 유닛마다 E_p의 에너지를 생산하며 절대 시간 *t*에 커패시터에 남은 에너지양을 *E(t)*라고 정의한다. 각 태스트 τ_i가 스케줄링 되어 수행되면 매 시간 유닛 동안 e_{Max}까지 에너지를 소비하고 해당 태스크의 최대 수행 시간인 C_i동안 E_i를 나누어 소비한다.



(a) Task scheduling and energy consumption by EDF



(b) Task scheduling and energy consumption by ED-H [16]

Fig. 3. An energy-aware scheduling example

예를 들어, 그림 3(a)의 예에서 *C=6*, *E(0)=4*, $E_p=1$, $\tau_1(C_1, R_1, D_1, T_1, E_1)=\tau_1(1,0,8,8,2)$, $\tau_2(3,1,6,5,8)$의 두 실시간 태스크의 첫 번째 인스턴스를 EDF 스케줄링 알고리즘으로 스케줄링 할 경우 시간 *0*에 τ_1이 릴리즈 되고 τ_2는 아직 릴리즈 되지 않았으므로 τ_1이 가장 우선순위가 높은 수행 가능한 태스크가 되고 $C_1=1$ 만큼 스케줄링 되어 첫 인스턴스의 수행을 마친다. 이때 $E_1=2 < e_{Max}$이므로 [0,1) 시간 유닛 동안 에너지 하베스터는 $E_p=1$만큼 에너지를 생산하고 τ_1는 $E_1=2$ 만큼 소비하므로 $E(1) = E(0) - (E_1-E_p) = 4 - (2-1) = 3$이 된다.

반면 τ_2는 절대 시간 *1*에 릴리즈 되어 수행을 시작하는데, $E_2=8 > e_{Max}$이므로 태스크는 시간 유닛 *[1,2)*와 *[2,3)*에서 각각 에너지 소비 상한인 $e_{Max} = 3$을 소비하고 마지막 시간 유닛인 *[3,4)*에 잔여 에너지인 $(E_2=8)-3-3=2$를 소비하게 된다. 그런데, EDF 스케줄링 알고리즘은 그림 3(a)에서 볼 수 있듯이 생산되는 에너지와 소비하는 에너지를 전혀 고려하지 않기 때문에, τ_2가 시간 유닛 *[1,2)*에서 하베스터에 의해 *1*만큼 에너지가 생산되고 태스크는 소비 상한인 $e_{Max} = 3$까지 소비하기 때문에 $E(2) = E(1) - (3-1) = 3 - (3-1) = 1$이 되고 같은 방식으로 τ_2가 시간 유닛 *[2,1)*

동안 *2*만큼의 에너지를 커패시터에서 소비하기 때문에 *E(3) = E(2) - 2 = -1*로 커패시터의 에너지가 고갈되어 태스크의 수행이 불가능해져 실시간 스케줄링에 실패하게 되므로 슈퍼커패시터와 에너지 하베스터에 의한 일시적으로 제한될 수 있는 전원 공급 환경에는 적용이 불가하다.

2.2 ED-H Scheduling

에너지 하베스팅 시스템을 위한 실시간 스케줄링 알고리즘은 온라인 즉 수행 시간 중에 태스크의 우선순위만을 고려하여 다음 시간 유닛 동안 수행할 태스크를 구하는 것이 아니라 에너지 하베스트로부터 생산되는 에너지와 태스크가 소비하는 에너지를 모두 고려하여 실시간 태스크의 시간 제약성을 만족하면서도 태스크가 수행될 때 커패시터에 저장된 에너지가 고갈되어 수행되지 못하는 경우가 없도록 실시간 스케줄링 알고리즘을 수행해야 한다.

ED-H 스케줄링 알고리즘은 2.1절의 스케줄링 예와 같이 시간 제약성 뿐이 아니라 태스크가 스케줄링 되는 동안 커패시터의 잔여 에너지를 사전에 계산하여 시간 제약성을 여전히 만족하지만 의도적으로 태스크를 지연시켜 에너지 하베스터에 의해 생산되는 에너지와 태스크에 의해 소비되는 에너지가 최악의 경우에도 고갈되지 않도록 태스크를 스케줄링한다.

ED-H 스케줄링 알고리즘은 이를 위해 임의의 태스크 τ_i가 임의의 시간 유닛 *[t,t+1)*에 스케줄링 되기 위한 규칙을 그림 4와 같이 정의하였으며 이러한 규칙을 만족하는 어떠한 태스크 스케줄링도 커패시터의 에너지가 고갈되지 않으면서 모든 시간 제약성을 만족한다는 것을 증명하였다.

그림 4의 규칙 3은 현재 수행 가능한 태스크가 있음에도 불구하고 수행하였을 때 에너지가 고갈되는 경우 어떠한 태스크도 스케줄링 하지 않고 에너지 하베스터에 의해 에너지를 생산하여 커패시터를 충전하도록 한다. 규칙 4는 커패시터에 에너지가 가득 찼음에도 불구하고 태스크를 수행하지 않았을 때 이로 인해 태스크의 시간 제약성을 만족시키지 못하는 경우를 예측하고 가장 우선순위가 높은 태스크를 스케줄링하여 시간 제약성을 만족시키도록 한다.

그림 3(b)는 [16]에서 ED-H 스케줄링 알고리즘의 규칙에 따른 동작 원리를 설명하기 위한 태스크 인스턴스 예제로 EDF 스케줄링과 다르게 시간 유닛 *[0,2), [5,7)* 동안 수행 가능한 태스크가 있음에도 불구하고 태스크를 수행시키지 않아 커패시터를 충전할 수 있고 이로 인해 에너지 고갈을 방지하면서도 동시에 모든 태스크 인스턴스의 시간 제약성을 만족시키는 스케줄링을 생성한다.

$L_r(t)$ = list of ready tasks at clock t

$PSE_\tau(t)$ = remaining energy after executing tasks released and finished within (t,d), where d = latest deadline among ready tasks

$ST_\tau(t)$ = remaining time slots after executing tasks release and finished within (t,d)

- Rule 1: EDF priority order is used to select the running job in $L_r(t)$ in [t, t+1)
- Rule 2: The processor is idle in [t, t+1) if $L_r(t)$ is empty.
- Rule 3: The processor is idle in [t, t+1) if $L_r(t)$ is not empty and one of the following conditions is satisfied:
  1) $0 \leqq E(t) < e_{Max}$
  2) $0 \leqq PSE\tau\ (tc) < e_{Max}$
- Rule 4: The processor is busy (or running a task) in [t, t+1) if $L_r(t)$ is not empty and one of the following conditions is satisfied:
  1) $C \leqq E(t) < C + e_{Max}$
  2) $ST\tau(t) = 0$
- Rule 5: The processor can equally be idle or busy in [t, t+1) if $L_r(t)$ is not empty, $0 < E(t) < C$, $ST_\tau(t) > 0$ and $PSE_\tau\ (t) > 0$

Fig. 4. Rules to select a running task at clock *t* in ED-H scheduling algorithm [16]

III. The Proposed Scheme

1. Scheduling algorithm implementation

ED-H 스케줄링 알고리즘은 2.2절에서 기술한 바와 같이 그림 4의 규칙을 만족하는 어떠한 태스크 스케줄링에 대해서도 시간 제약성을 만족시키고 동시에 에너지를 고갈시키지 않는다.

반면 실제 구현에 있어서는 그림 4의 규칙 5에서 시간 유닛 *t*에 태스크를 스케줄링을 해도 되고 안해도 되는 경우에 대한 처리가 필요한데, [16]에서는 이를 ASAP (As Soon As Possible) 정책과 ALAP (As Late As Possible) 정책으로 나누어 스케줄링 가능성을 분석하였지만 이에 대한 세부적인 구현 방법에 대한 정의하지 않았다.

본 논문에서는 해당 규칙을 두 정책에 대해 구현하고 이를 프로그램화할 때 규칙이 적용되는 순서를 그림 5와 같이 구현하였다.

```
ED-H_schedule(t)
{
// Rule 2
if (Lr(t) is empty)
   return;

// Rule 1
Tτ ← the highest priority task in Lr(t)

// Rule 4
if (C ≦ E(t) < C + eMax || STτ(t) = 0)
{
   execute_one_time_slot(Tτ);
   return;
}

// Rule 5
if (0 < E(t) < C && STτ(t) > 0 && PSEτ (t) > 0)
{
   if (policy = ALAP)
      return;
   else { // policy = ASAP
         execute_one_time_slot(Tτ);
         return;
   }
}

// Rule 3
if (0 ≦ E(t) < eMax || 0 ≦ PSEτ (tc) < eMax)
{
   return;
}

// Failed to schedule
exit;
}
```

Fig. 5. ED-H algorithm implementation for ALAP and ASAP policy in this paper

규칙 3의 경우 커패시터의 용량이 프로세서가 *1* 시간 유닛 동안 최대한 소비하는 에너지보다 적은 경우 혹은 수행 가능한 태스크를 수행하였을 때 잔여 에너지가 적은 경우에 태스크를 스케줄링하지 않는 경우를 정의하는데, 해당 조건이 규칙 5의 첫 번째 조건과 세 번째 조건과 겹치는 경우가 발생하므로 본 논문의 구현에서 규칙 3을 먼저 적용하는 경우 무조건 태스크를 스케줄링하지 않고 유휴 시간으로 남기는 경우가 발생할 수 있으므로 가장 마지막에 규칙 3을 먼저 적용하고 그 이후 규칙들을 순서대로 적용했을 때 다른 모든 규칙에 해당되지 않으면 스케줄링이 실패한 것이므로 이를 별도로 처리하였다.

규칙 5의 경우 태스크를 수행하여도 수행하지 않아도 스케줄링이 실패하지 않는 경우이므로 수행 가능할 때 가장 빠르게 태스크를 수행하는 ASAP 정책에서는 규칙 5에 해당하므로 태스크를 바로 수행하도록 구현하였고. ALAP 정책에서는 규칙 5에서 태스크를 수행하지 않도록 구현하였다. 이때 규칙 5에 따라 태스크가 수행되지 않는 경우라 하더라도 뒤에 이어지는 시간 유닛 중에서 규칙 4에 의해 더 이상 태스크의 수행을 지연시킬 여력이 없는 경우, 즉 태스크를 이번 시간 유닛에 수행하지 않으면 시간 제약성을 위배하게 되므로 규칙 4에 의해 수행되도록 스케줄링하였다.

2. Comparisons of ASAP and ALAP

그림 6은 본 논문에서 그림 5에 의해 구현된 ED-H 알고리즘에 대해 [16]의 스케줄링 예제인 $C=6$, $E(0)=4$, $E_p=1$, $e_{Max}=3$, $\tau_1(C_1, R_1, D_1, T_1, E_1) = \tau_1(1,0,8,8,2)$, $\tau_2(3,1,6,5,8)$를 ASAP 정책과 ALAP 정책에 따라 스케줄링한 결과이다. 그림 3의 예제는 τ_1과 τ_2의 첫 인스턴스까지만 수행한 결과였으므로 두 번째 인스턴스까지 수행을 확장하였다.

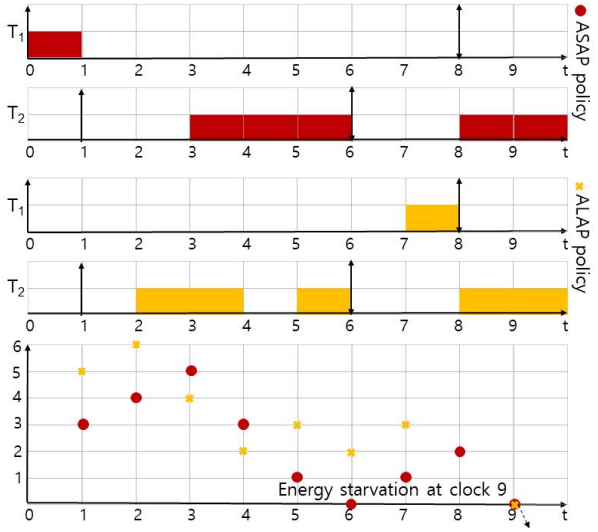


Fig. 6. Scheduling Example of ASAP and ALAP policies in ED-H

상단의 ASAP 정책의 경우 τ_1는 첫 번째 인스턴스가 수행이 최초 가능한 시점인 절대 시간 *0*에 바로 스케줄링 되었고, τ_2는 두 번째 인스턴스가 수행이 가능한 최초 시점인 절대 시간 *0*에 바로 스케줄링 되었다. 반면 하단의 ALAP 정책의 경우 두 태스크 모두 첫 번째 인스턴스가 데드라인에 이르러서야 수행을 마친 것을 볼 수 있다.

그런데, 두 정책 모두 절대 시간 *9*에 커패시터의 에너지가 고갈되면서 스케줄링이 실패하게 된다. [16]의 예제에서는 두 태스크의 첫 번째 인스턴스만을 제시하였기 때문에 에너지 고갈이 발생하지 않았으나 각 태스크의 주기를 적용하여 수행할 경우 예제의 태스크 집합 τ는 스케줄링에 실패하며 그 원인을 아래와 같이 확인하였다.

태스크 집합의 주기가 상이하더라도 태스크 스케줄링에 있어서 모든 태스크 주기의 최소공배수까지의 스케줄링이

계속해서 반복되는 특성이 있기 때문에 예제 태스크 집합 τ의 최소공배수인 절대 시각 *40*까지 스케줄링하였다. 이때 각 태스크가 소비하는 에너지를 계산해보면 τ_1은 주기가 *8*로 절대 시각 *40*까지 *5*번 릴리즈 되어 총 $5 \times 2 = 10$의 에너지를 소비하며, τ_2는 주기가 *5*로 절대 시각까지 약 *8*번 릴리즈 되어 총 $8 \times 8 = 64$의 에너지가 소비되어 태스트 집합 τ의 총 에너지 소비는 약 *74*가 된다.

반면 에너지 하베스터에 의한 에너지 생산 E_p는 각 시간 유닛당 *1*이므로 필요한 에너지 소비 약 *74*에 필요한 에너지 총량에 더해 τ_2의 첫 인스턴스가 절대 시간 1에 릴리즈 되므로 ED-H에 의한 스케줄링에 의하면 마지막 인스턴스의 마지막 수행 시간 중 소비되는 에너지양 *2*를 제외하고 약 *32*가 부족하게 된다. 따라서, (1) 부족한 에너지를 충분한 용량의 커패시터를 사전에 모두 채워 넣거나 (2) 단위 시간 유닛 당 생산하는 에너지의 총합이 소비되는 에너지보다 커야지만 예제의 태스크 집합 τ의 스케줄링이 가능하다는 점을 확인 할 수 있다.

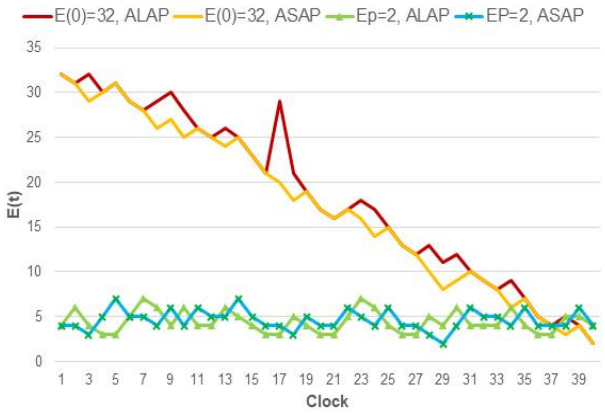


Fig. 7. Adjusting E(0) or E_p to prevent any energy starvation during the entire cycles

그림 7은 (1)의 경우인 $C=E(0)=32$로 설정한 스케줄링 결과 및 (2)의 경우인 $E_p=2$ (*절대 시각 40까지 에너지 생산량=40 × 2 =10 > 에너지 소비량 약 74*)에 대한 스케줄링 결과를 도식한 것이며, 본 논문에서 구현한 ASAP 정책과 ALAP 정책 모두에 대해 성공적으로 스케줄링하였음을 보여준다.

3. An extension to support iPMU

슈퍼 커패시터를 보조 전원장치로 사용하고 태양광 패널을 에너지 하베스터로 사용하는 전원 공급 시스템을 실제로 구현하기 위해서는 태양광 패널에 의해 공급되는 전기 에너지를 슈퍼 커패시터에 충전하면서 동시에 마이크로컨트롤러에 필요한 전력을 공급하도록 해야 하며, 생산되는 에너지가 충분하지 않은 경우 회로와 슈퍼 커패시터를 보호하기 위해 전력 공급을 일시적으로 차단하고 커패시터가 충분히 충전되었을 때 다시 전력 공급을 재개하는 역할을 하는 iPMU가 필요하다.

iPMU가 사용될 경우 전원 제어 회로에 의해 의도적으로 마이크로컨트롤러에 대한 전원 공급이 일시적으로 차단되는 상태인 블랙아웃이 발생할 수 있다.

iPMU는 특정 전압 이하가 되면 전원 공급을 차단하고 충분한 전압 이상이 된 후에 전원 공급을 재개하게 되는데, 본 논문에서는 특정 전압 이하가 되었을 때 즉 전원이 차단 될 때의 전압에 대응하는 커패시터의 용량을 C_{cut}이라고 정의하고 전원 공급이 재개 될 때의 전압에 대응하는 용량을 C_{rdy}라고 정의하였다[11].

이러한 블랙아웃 상태에 의한 스케줄링 가능 여부를 확인하기 위해 [16]의 태스크 집합 τ에 대해 스케줄링을 수행하였다. 실험에서 스케줄링에 필요한 커패시터의 용량과 초기 충전량에 대한 변경이 필요하므로 $C=10$, $E(0)=8$, $C_{cut}=6$, $C_{rdy}=8$로 설정하였다.

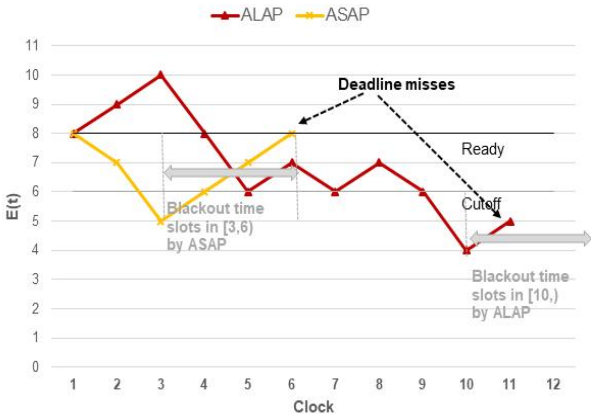


Fig. 8. Scheduling failure due to power mode changes managed by iPMU

그림 8에서 확인할 수 있듯이 ED-H 알고리즘은 커패시터의 최대 용량과 초기 용량을 높였음에도 불구하고 iPMU에 의한 블랙아웃 상태를 고려하지 않았다. 또한 각 정책에 따라 ED-H의 스케줄링 규칙을 그대로 적용하면 ASAP 정책에서 $E(3)=5 < C_{cut}=6$으로 블랙아웃 상태가 시작되어 태스크를 수행하지 못하게 되고 이후 매 시각 충전을 지속하여 $E(6)=8 \geqq C_{rdy}=8$에 이르러서야 태스크의 수행이 가능한 상태에 도달하지만, 시간 제약성을 위배하게 된다. ALAP 정책의 경우 $E(10)=4 < C_{cut}=6$으로 블랙아웃 상태가 시작되어 절대 시간 11에 시간 제약성을 위배하게 되어

두 정책 모두 스케줄링에 실패하게 된다는 점을 확인하였으므로 iPMU을 특성을 고려한 ED-H 알고리즘의 확장이 필요하다.

```
iPMU_schedule(t)
{
// Calculate time slots in blackout
// blackout occurs if remaining energy
//    is less than Ccut
// blackout = the number of time slots
//    E(t) becomes above Crdy
if ((Ccut-PSEτ(t))/Ep > STτ(t))
{
    blackout ← (Ccut-PSEτ(t))/Ep
}
else
{
    blackout ← 0
}

// Rule 2
if (Lr(t) is empty)
   return;

// Rule 1
Tτ = the highest priority task in Lr(t)

// Rule 4 (adjust STτ(t) to count 'blackout')
if (C≦E(t)<C+eMax || STτ(t)-blackout = 0)
{
   execute_one_time_slot(Tτ);
   return;
}

// Rule 5 (adjust STτ(t) to count 'blackout')
if (0<E(t)<C && STτ(t)-blackout>0 &&
PSEτ(t)>0)
{
   execute_one_time_slot(Tτ);
   return;
}

// Rule 3
if (0≦E(t)<eMax || 0≦PSEτ(tc) < eMax)
{
   return;
}

// Failed to schedule
exit;
}
```

Fig. 9. Extended ED-H for handling blackout time slots

본 논문은 블랙아웃 상태가 발생하였을 때 전원 공급이 차단되어 다시 충분한 에너지가 충전될 때까지 태스크의 수행이 지연되는 일종의 시간 제약성에 대한 패널티의 시간 유닛 수를 먼저 계산하여 이를 그림 9의 확장된 ED-H 알고리즘에서 규칙 4와 규칙 5의 STτ(t)에 보정하도록 하여 iPMU의 블랙아웃 상태가 발생함에도 불구하고 스케줄링의 시간 제약성을 만족하도록 ED-H 알고리즘을 확장하였다.

4. Simulation results

본 논문에서 제안한 iPMU를 지원하는 확장 ED-H 알고리즘의 정상적인 동작을 확인하기 위해 [16]의 예제 태스크 집합 τ에 τ_3 태스크를 추가하여 복잡도를 높인 $C=6$, $E(0)=4$, $E_p=2$, $e_{Max}=3$, $C_{cut}=3$, $C_{rdy}=4$ $\tau_1(C_1, R_1, D_1, T_1, E_1)=\tau_1(1,0,8,8,2)$, $\tau_2(3,1,6,5,8)$, $\tau_3(3,0,20,20,8)$에 대해 ED-H 알고리즘의 ALAP과 ASAP 정책과 본 논문의 iPMU 확장 알고리즘에 대해 수행하여 비교하였다.

그림 10은 모든 태스크 주기의 최소공배수인 절대 시간 40까지의 $E(t)$의 변화를 도식화하였으며, 그림 11은 각 시간 유닛마다 스케줄링 된 태스크 인스턴스를 도식화하였다. 그림에서 ED-H 알고리즘의 ALAP 정책은 절대 시간 21에 커패시터에 잔여 에너지가 있음에도 불구하고 시간 제약성 만족에 실패하여 스케줄링이 중단된 것을 확인할 수 있다. 반면 ED-H 알고리즘의 ASAP 정책은 본 논문에서 제안한 iPMU 확장 알고리즘과 동일한 $E(t)$ 변화를 갖으며 스케줄링에 성공하는 것을 확인할 수 있다.

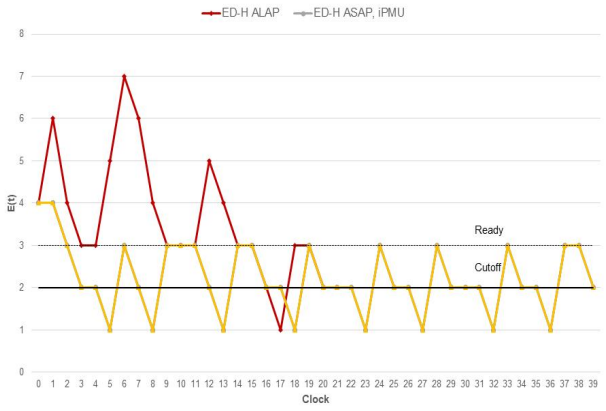
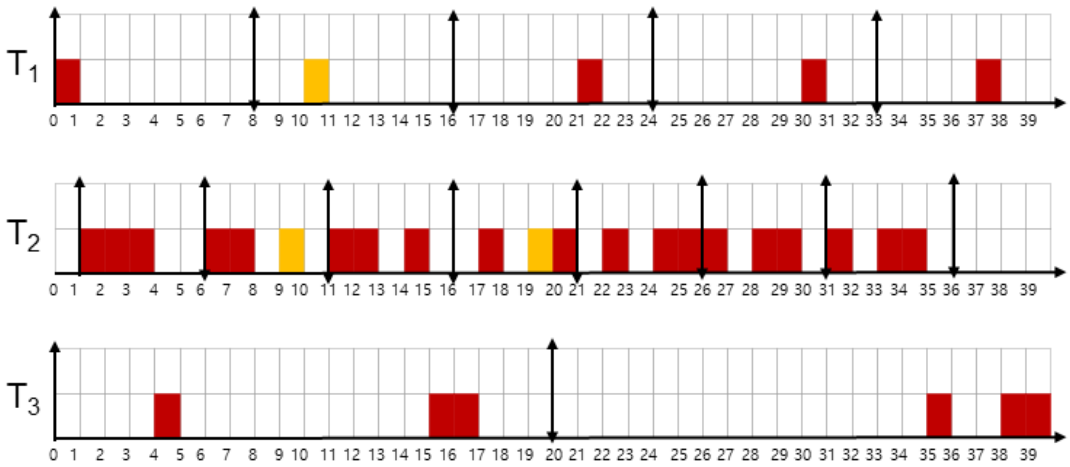


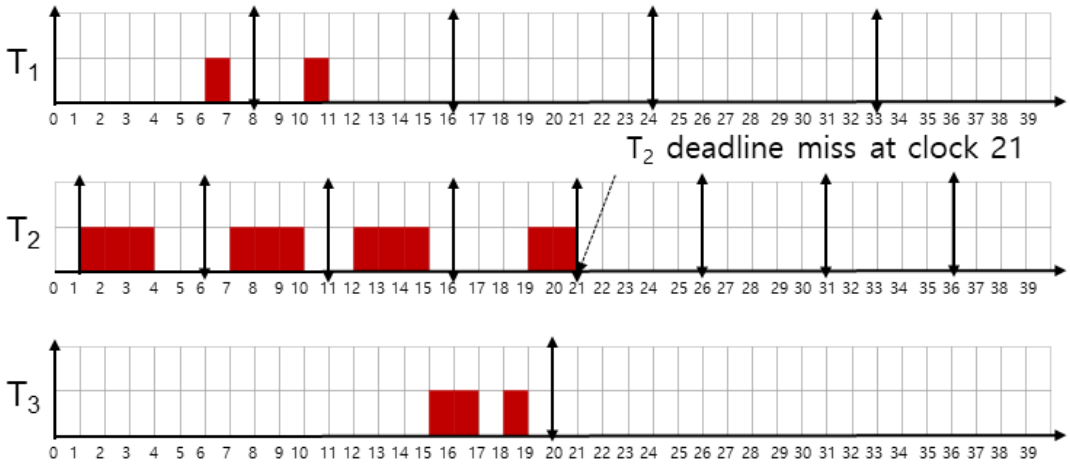
Fig. 10. Scheduling failure due to power mode changes managed by iPMU

그러나, ED-H 알고리즘의 ASAP 정책에 의한 스케줄링과 iPMU 확장 알고리즘에 의한 스케줄링의 $E(t)$ 변화가 동일하다고 해서 동일한 스케줄링을 생성하는 것은 아니다. 그림 11의 주황색으로 표기된 3개의 인스턴스의 실행 시간 유닛에 차이가 있어 동일한 스케줄링을 생성하는 것은 아니라는 것을 확인하였다. 따라서, iPMU 확장 알고리즘에서 스케줄링이 가능하더라도 블랙아웃 상태를 고려하지 않은 ED-H 알고리즘의 ASAP에서 스케줄링이 가능하지 않은 경우에 대한 추가 실험을 수행하였다.

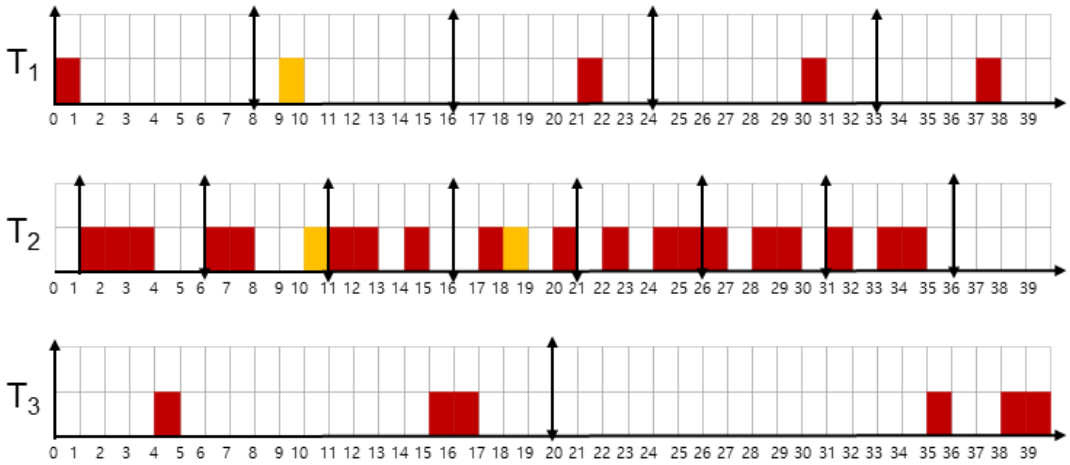
이를 위해 태스크 집합을 GNU 과학 라이브러리 (GSL: GNU Scientific Library) [20]의 난수 발생기를 사용하여 무작위로 생성하여 스케줄링을 수행하였다. 실험에서



(a) Successful task scheduling by ED-H ASAP policy



(b) Failed task scheduling by ED-H ALAP policy



(c) Successful task scheduling by iPMU extension

Fig. 11. Task scheduling comparison between ED-H and iPMU

태스크 집합은 크게 3 종류(category)로 분류되며 태스크 집합 유형 0은 낮은 실행 시간과 에너지 비중을 갖는 태스크 만으로 구성되고, 태스크 집합 유형 1은 높은 비중을 갖는 태스크 만으로 구성되며, 마지막으로 태스크 유형 2는 낮은 비중의 태스크와 높은 비중의 태스크를 혼합한 태스크 집합으로 구성하였다. 태스크 유형 0은 태스크의 비중이 [0.1, 0.5) 범위에, 태스크 유형 1은 [0.5, 0.9) 범위에 균일하게 분포하도록 하고, 태스크 유형 2는 낮은 비중의 태스크와 높은 비중의 태스크가 0.2대 0.8의 비율로 분포하도록 하였다.

이때 태스크 집합 인스턴스의 첫 100개를 스케줄링하였으며, 생성된 태스크 집합이 상기 기술한 커패시터의 최대 용량과 초기 용량 및 단위 시간당 에너지 생산량이 태스크 집합에 의해 소비되는 에너지보다 큰 경우에 대해서만 즉 스케줄링이 실현 가능한 경우에만 실험을 수행하였다.

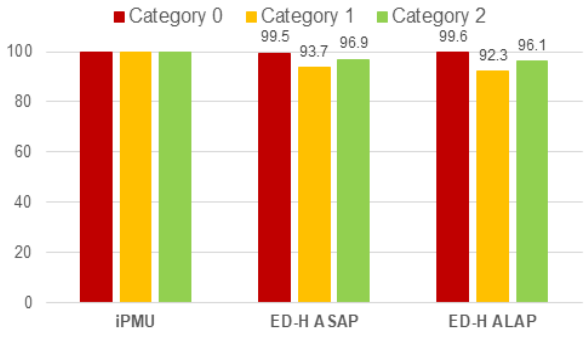


Fig. 12. Simulation results for randomly generated task sets

그림 12는 본 논문에서 제안한 iPMU 확장 알고리즘을 기준으로 스케줄링 성공 비율을 정규화한 것으로 본 논문에서 제안한 알고리즘이 원형 ED-H 알고리즘과 비교했을 때 태스크 집합의 유형 특성에 따라 0.4%~7.7%까지 스케줄링 성능이 높은 것을 확인하였다.

IV. Conclusions

EDF 스케줄링 알고리즘은 단일 프로세서 환경에서 이용률 100% 이하의 실시간 태스크에 대해 모든 시간 제약성을 만족하는 최적의 알고리즘이다. 에너지 하베스팅 시스템을 위한 스케줄링 알고리즘은 생산되는 에너지와 태스크가 소비하는 에너지를 모두 고려하여 실시간 태스크의 시간 제약성을 만족하면서도 태스크가 수행될 때 커패시터에 저장된 에너지가 고갈되어 수행되지 않는 경우가 없도록 실시간 스케줄링을 수행해야 하며, ED-H 스케줄링 알고리즘은 이러한 환경에서 동작하는 유일한 최적의 알고리즘이다.

ED-H 알고리즘의 실제 구현에 있어서 세부적인 구현 방법에 대한 정의가 되어 있지 않아 본 논문에서는 ED-H 알고리즘의 분석을 위해 제시된 규칙과 정책들을 구현하고 이를 프로그램화할 때 적용되는 순서를 구현하고 서로 다른 정책에 대해 비교하였다. 더불어 실제 커패시터와 결합 된 에너지 하베스터를 적용하기 위해 필수적인 지능형 전원 관리 유닛으로 인하여 발생하는 블랙아웃 상태를 ED-H 스케줄링 알고리즘의 규칙에 확장함으로써 실제 시스템에서의 스케줄링 성능을 향상 시켰다. 시뮬레이션을 통해 본 논문에서 제안한 알고리즘이 원형 ED-H 알고리즘과 비교했을 때 태스크 집합의 유형 특성에 관계없이 스케줄링 성능이 높은 것을 확인하였다.

향후 연구과제로는 본 논문의 iPMU를 지원하기 위해 확장된 ED-H 알고리즘을 실제 하드웨어에 구현하고 데이터 스트리밍 응용에 적용하여 성능을 분석할 예정이며, 추가적으로 실제 응용에서 필요한 비주기적 태스크들의 시간 제약성을 만족하도록 확장하는 알고리즘을 설계하고 구현할 예정이다.

ACKNOWLEDGEMENT

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education (NRF-2021R1F1A1050088).

REFERENCES

[1] K. Routh and T. Pal, "A survey on technological, business and societal aspects of Internet of Things by Q3, 2017," 2018 3rd International Conference On Internet of Things: Smart Innovation and Usages (IoT-SIU), pp. 1-4, Bhimtal, India, Feb. 2018, DOI: 10.1109/IoT-SIU.2018.8519898.

[2] G. Saha, R. Singh, and S. Saini, "A Survey Paper on the impact of "Internet of Things" in Healthcare," 2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA), pp. 331-334, Coimbatore, India, Jun. 2019, DOI: 10.1109/ICECA.2019.8822225.

[3] M. Gohar, S. H. Ahmed, M. Khan, N. Guizani, A. Ahmed, and A. Ur Rahman, "A Big Data Analytics Architecture for the Internet of Small Things," IEEE Communications Magazine, Vol. 56, No.

2, pp. 128-133, Feb. 2018, DOI: 10.1109/MCOM.2018.1700273.

[4] Y. Lei, C. Zhang, Y. Gao, and T. Li, "Charging Optimization of Lithium-ion Batteries Based on Capacity Degradation Speed and Energy Loss," Energy Procedia, Vol. 152, pp. 544-549, Nov. 2018, DOI: 10.1016/j.egypro.2018.09.208.

[5] M. Kabakulak and S. Arslan, "Design and Application of an Electromagnetic Energy Harvester for Wireless Sensor Network," 2020 International Conference on Electrical, Communication, and Computer Engineering (ICECCE), pp. 1-6, Istanbul, TurkeyJun, 2020, DOI: 10.1109/ICECCE49384.2020.9179409.

[6] J. S. Bestley, T. Gomathi, and R. B. R. Samuel, "Efficient low wind energy harvesters for wireless sensor networks," 2018 2nd International Conference on Inventive Systems and Control (ICISC), pp. 785-790, Coimbatore, India, Jan. 2018, DOI: 10.1109/ICISC.2018.8398906.

[7] M. Saliya, N. Kouvelas, R. V. Prasad and N. Hokke, "Characterizing and Optimizing Piezo Harvesters for Train Interiors," 2020 IEEE SENSORS, pp. 1-4 Oct. 2020, DOI: 10.1109/SENSORS47125.2020.9278637.

[8] S. Park, “Design and Evaluation of the Internet-Of-Small-Things Prototype Powered by a Solar Panel Integrated with a Supercapacitor,” Journal of the Korea Society of Computer and Information, Vol. 26, No. 11, pp. 11-19, Nov. 2021, DOI: 10.9708/JKSCI.2021.26.11.011.

[9] F. Deng, X. Yue, X. Fan, S. Guan, Y. Xu, and J. Chen, "Multisource Energy Harvesting System for a Wireless Sensor Network Node in the Field Environment," IEEE Internet of Things Journal, Vol. 6, No. 1, pp. 918-927, Feb. 2019, DOI: 10.1109/JIOT.2018.2865431.

[10] A. Sabovic, A. K. Sultania, C. Delgado, L. D. Roeck, and J. Famaey, “An Energy-Aware Task Scheduler for Energy-Harvesting Batteryless IoT Devices,” IEEE Internet of Things Journal, Vol. 9, No. 22, pp. 23097-23114, Jun. 2022, DOI: 10.1109/JIOT.2022.3185321.

[11] E-Peas, “Highly-Efficient, Regulated Dual-Output, Ambient Energy Manager for up to 7-cells solar panels with optional primary battery,” https://kamaka.de/en/e-peas-highly-efficient-regulated-dual-output-ambient-energy-manager-for-up-to-7-cells-solar-panels-with-optional-primary-battery/

[12] L. Yao, H. Zhao, J. Tang, S. Liu, and J. -L. Gaudiot, "Streaming Data Priority Scheduling Framework for Autonomous Driving by Edge," 2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC), pp. 37-42, Madrid, Spain, Jul. 2021, DOI: 10.1109/COMPSAC51774.2021.00017.

[13] J. Li, G. Zheng, H. Zhang and G. Shi, "Task Scheduling Algorithm for Heterogeneous Real-time Systems Based on Deadline Constraints," 2019 IEEE 9th International Conference on Electronics Information and Emergency Communication (ICEIEC), pp. 113-116, Beijing, China, Jul. 2019, DOI: 10.1109/ICEIEC.2019.8784641.

[14] J. Lu, W. Zheng and Y. Xiao, "A Task Scheduling Algorithm for Minimizing Worst- Case Execution Time of Real-Time Embedded Systems," 2021 International Conference on Computer Information Science and Artificial Intelligence (CISAI), pp. 479-482, Kunming, China, Sep. 2021, DOI: 10.1109/CISAI54367.2021.00097.

[15] M. M. Sandhu, S. Khalifa, R. Jurdak, and M. Portmann, "Task Scheduling for Simultaneous IoT Sensing and Energy Harvesting: A Survey and Critical Analysis," arXiv: Signal Processing, Apr. 2020. DOI: 10.48550/arXiv.2004.05728/.

[16] M. Chetto, "Optimal Scheduling for Real-Time Jobs in Energy Harvesting Computing Systems," IEEE Transactions on Emerging Topics in Computing, Vol. 2, No. 2, pp. 122-133, Jan. 2014, DOI: 10.1109/TETC.2013.2296537.

[17] R. E. Osta, M. Chetto, H. E. Ghor, and R. Hage, "Real-time scheduling of aperiodic tasks in energy harvesting devices," 2017 Sensors Networks Smart and Emerging Technologies (SENSET), pp. 1-4, Sep. 2017, DOI: 10.1109/SENSET.2017.8125053.

[18] A. Mishra and A. K. Ray, "Energy-efficient Design of Wireless Sensor Mote using Mobile-Edge Computing and Novel scheduling mechanism for self-sustainable Next-gen Cyber Physical System," 2021 Second International Conference on Smart Technologies in Computing, Electrical and Electronics (ICSTCEE), pp. 1-8, Bengaluru, India, Dec. 2021, DOI: 10.1109/ICSTCEE54422.2021.9708557.

[19] J. Leithon, L. A. Suárez, M. M. Anis, and D. N. K. Jayakody, "Task Scheduling Strategies for Utility Maximization in a Renewable-Powered IoT Node," in IEEE Transactions on Green Communications and Networking, Vol. 4, No. 2, pp. 542-555, Jun. 2020, DOI: 10.1109/TGCN.2019.2959730.

[20] GNU scientific library, http://www.gnu.org/software/gsl/

Authors

Sangsoo Park received the BS degree in Computer Science from Korea Advanced Institute of Science and Technology, Daejeon, Korea, in 1998, and the MS and PhD degrees in Computer Science & Engineering from Seoul National University, Seoul, Korea, in 2000 and 2006, respectively. Dr. Park joined the faculty of the Department of Computer Science & Engineering at Ewha Womans University, Seoul, Korea, in 2009. He is currently a Professor in the Department of Computer Science & Engineering, Ewha Womans University. He is interested in real-time embedded systems and system software.