Constraint-Based Dynamic Energy Management in Energy Harvesting Embedded Systems

WeiYing Yi
SID: 1155026053
Department of Computer Science and Engineering
The Chinese University of Hong Kong, Hong Kong
Email: wyyi@cse.cuhk.edu.hk

I. Introduction

In [1], a tablet-based home energy cost saving and appliance scheduling system was developed. Such system calculates the behavior change suggestions of the appliance that save occupants on their energy bills while avoiding disruption on their regular routines. The objective of the proposed system is to propose changes to a user's behavior which achieve their energy goals, while still being acceptable for their health, comfort and enjoyment.

The idea described above can be adapted to many energy constrained systems, and has large amount of practical interests in application and research. For example, we can adapt the idea into embedded systems powered by renewable energy. Then we want to develop a energy management system such that it is able to maximize the usage of the renewable energy and minimize the amount of wasted energy, while still being able to provide good enough services.

However, the energy management solution provided by [1] needs to be reconsidered when energy harvesting devices are being utilized instead of power lines. This is because the stochastic nature of renewable energy, which is usually unstable and varies along time. Clearly, one may just use renewable energy to recharge a primary battery that provides static DC power. In such situation, surplus energy is wasted as the battery has charging efficiency around 0.8. In fact, the most efficient way is that the embedded system accesses to the renewable energy directly without any energy device. But then the operation time of the embedded system is totally depend on the source of renewable energy and this is usually the most to avoid. Moreover, when the capacity of the battery is small, surplus of harvested energy is wasted if the battery is full. In order to address these issues, we propose a constraint-based dynamic energy management system especially for energy harvesting embedded systems. The proposed energy management system is able to fully utilize the harvested energy and minimize the wasted energy while providing good enough services, even when the energy harvesting device and the capacity of battery are small.

As a possible example scenario, in this project, we focus on wireless sensing nodes that are powered by solar cells. Each sensor node consists of a main processor, several environmental condition sensors, a small battery, a small solar cell and a wireless communication module (as shown in Figure 1). The proposed energy management system is divided into two separated subsystems, namely the Application Rate Layer and the Real-time Scheduling Layer. Promising results with energy saving are achieved by the proposed energy management system.

This report is organized as follows: In Section II, the system concepts and the used models and methods are described. Also, the design objectives for the energy management system are listed. In Section III, the detail CSP models for the Application Layer and the Real-time Scheduling Layer are introduced and implemented. Then, in Section IV, we demonstrate the improvements of the proposed energy management system compared with several naive energy management strategies. Last but not least, we conclude the advantages and disadvantages of the proposed energy management system and discuss the limitations and future works.

II. System Model and Design Objectives

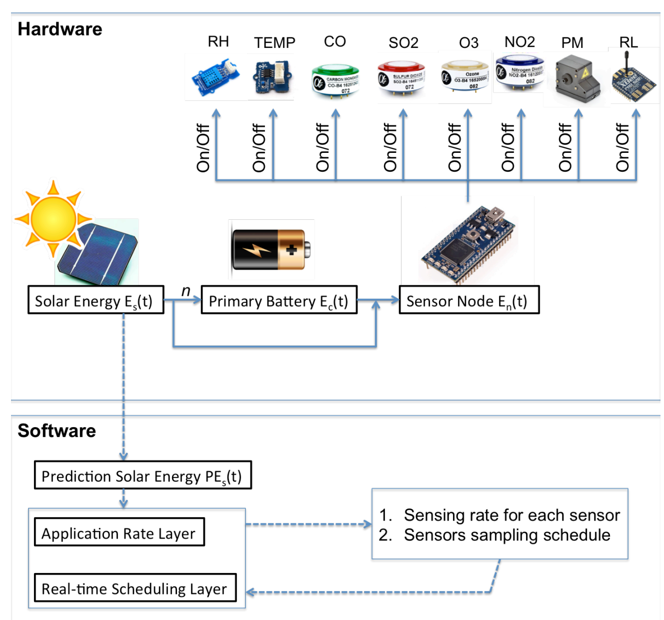

Fig. 1. Illustration of the system model

The system model is shown in Figure 1. The whole hardware and software systems are powered by the energy harvesting device, i.e. the solar cell, which delivers energy $E_s(t)$. On one hand, the harvested energy is then loaded into the primary battery with capacity C and the stored energy available at time t is $E_C(t)$. The primary battery has charging efficiency n. On the other hand, the harvested energy is consumed directly by the sensor node, with energy consumption $E_n(t)$, without storing into the primary battery. The capability to bypass the primary battery offers the opportunity to save substantial energy by using the solar energy directly when available because the charing efficiency n is definitely less than 100%. Also, the capacity of the primary battery is C and when it is charged up to C, if the energy consumption of the sensor node $E_n(t)$ is less than the harvested energy $E_s(t)$, surplus energy is wasted. Our goal is to develop a energy management strategy such that the wasted energy is minimized while the services provided by the sensor node is still good enough.

A. Assumptions

The most fundamental part of the proposed method is the prediction solar energy $PE_s(t)$, which is the predicted solar energy at time t. In this project, for simplicity reason, we assume the prediction correctness is always 100%. In fact, there are lost of researches are working on the prediction algorithms of solar energy and they are demonstrated to perform well [2], [3]. It is reasonable for us to assume the prediction correctness is 100% for simplicity reason.

In order to simplified the system model described above such that we are able to model the system behavior into a constraint satisfaction/optimization problem easily, we model the continuous time series into discrete intervals T. For any interval T_i, the harvested solar energy equals to the prediction solar energy, i.e., $E_s(T_i) = PE_s(T_i)$. The energy stored in the primary battery and the energy consumption of the sensor node can be expressed as $E_C(T_i)$ and $E_n(T_i)$ respectively. Note that, all expressions are describing the average energy in interval T_i.

For Application Rate Layer, we assume there is a function that returns the predicted solar energy for the next K Application Rate Layer intervals. And for Real-time Scheduling layer, we assume there is an other function that returns the predicted solar energy for the next L Real-time Scheduling intervals.

B. Application Rate Layer

In Application Rate Layer, there are 8 devices that are dynamically consuming energy, namely the CO sensor, the SO_2 sensor, the O_3 sensor, the NO_2 senor, the particulate matter (PM) sensor, the temperature (TEMP) sensor, the relative humidity (RH) sensor and the radio link (RL). The energy consumption expressions for the devices are e_{CO}, e_{SO2}, e_{O3}, e_{NO2}, e_{PM}, e_{TEMP}, e_{RH} and e_{RL} respectively, and they are independent of Application Rate Layer interval T_i. Also, the rewards of taking one sample of these 7 types of environmental condition are expressed as $reward_{CO}$, $reward_{SO2}$, $reward_{O3}$, $reward_{NO2}$, $reward_{PM}$, $reward_{TEMP}$ and $reward_{RH}$, respectively. They are also independent of Application Rate Layer interval T_i.

Our objective is to find the optimal operational rate of each type of devices for each interval T_i, i.e. $rate_{CO}(T_i)$, $rate_{SO2}(T_i)$, $rate_{O3}(T_i)$, $rate_{NO2}(T_i)$, $rate_{PM}(T_i)$, $rate_{TEMP}(T_i)$, $rate_{RH}(T_i)$ and $rate_{RL}(T_i)$, such that the wasted energy is minimized while respecting the limited and time-varying among of solar energy.

C. Real-time Scheduling Layer

After we have found the optimal operational rate for each type of devices for Application Rate interval T_i from the Application Rate Layer, we define each operation of the devices as a task and further split the Application Rate interval T_i into Real-time Scheduling interval T_j. One Application Rate interval T_i is evenly divide into several Real-time Scheduling interval T_j. There are 8 types of tasks, i.e. $task_{CO}$, $task_{SO2}$, $task_{O3}$, $task_{NO2}$, $task_{PM}$, $task_{TEMP}$, $task_{RH}$ and $task_{RL}$. The number of each type of task depends on the operational rate of the device. For example, if the operational rate of CO sensor is 10, then, the number of CO tasks are 10.

Our objective is to properly schedule the tasks such that sufficient good services are provided by the sensor node given the available energy.

III. Implementation

In this section, the detail implementation of the Application Rate Layer and the Real-time Scheduling Layer are described. Each layer is modeled into a constraint satisfaction/optimization problem and solve using constraint solver.

A. CSP Model for Application Rate Layer

In the Application Rate Layer, we set the Application Rate interval to 10 minutes long in time, and there are total 144 intervals in one day. The solar energy $E_s(T)$ is set to a specific discrete function and the solar energy prediction function returns the next K intervals' solar energy $PE_s(T_i)$ ... $PE_s(T_{i+K})$. The energy consumption of each type of devices are predetermined and the reward values of each type of sensor are decided by the users. The capacity of the primary battery is also set and we require that the remain energy in the battery can not less than an emergency level for emergency events.

As each Application Rate interval is 10 minutes and most of the current monitoring systems have sensing rate about 1 minutes [4]. We define the maximum operational rate in one Application Rate interval to 10.

We model the Application Rate Layer in to a CSP as follows:

Variables:

$rate_{CO}$: An array with size K. Describing the operational rate of CO sensor of the next K Application Rate intervals.

$rate_{SO2}$: An array with size K. Describing the operational rate of SO2 sensor of the next K Application Rate intervals.

$rate_{O3}$: An array with size K. Describing the operational rate of O3 sensor of the next K Application Rate intervals.

$rate_{NO2}$: An array with size K. Describing the operational rate of NO2 sensor of the next K Application Rate intervals.

$rate_{PM}$: An array with size K. Describing the operational rate of PM sensor of the next K Application Rate intervals.

$rate_{TEMP}$: An array with size K. Describing the operational rate of TEMP sensor of the next K Application Rate intervals.

$rate_{RH}$: An array with size K. Describing the operational rate of RH sensor of the next K Application Rate intervals.

$rate_{RL}$: An array with size K. Describing the operational rate of the radio link of the next K Application Rate intervals.

Domains:

$D(rate_{CO}) = D(rate_{SO2}) = D(rate_{O3}) = D(rate_{NO2}) = D(rate_{PM}) = D(rate_{TEMP}) = D(rate_{RH}) = D(rate_{RL})$ = [0, ..., 10]

Constraints:

We define that each device should be operate at least once in an Application Rate interval while satisfying the energy constraint. Also, the operational rate of the CO, SO2, O3 and NO2 sensors should be the same while the operational rates of TEMP and RH senor are the same. The operational rate of the radio link makes sure that the data generated by the sensors are all transmitted to the server.

In order to minimized the wasted energy while still providing good enough services, we post a constraint that maximizes the totals rewards in the next K Application Rate intervals. The reason is detail discussed in Section IV. Also, please refer to the ApplicationLevel.cpp file in appendix for the detail implementation of the constraints. Note that for each Application Rate interval T_i, the operational rate of each devices in the next K intervals are calculated. However, the solution for current interval is the first element of the variable arrays.

Note that in Application Rate Layer, we do not consider the execution time of sensor sampling or data transmitting. This is because the sampling time and the transmitting can be ignored comparing to the time of Application Rate interval.

B. CSP Model for Real-time Scheduling Layer

In the Real-time Scheduling Layer, each Application Rate interval is divided into L smaller intervals. Now the execution time of the sensor sampling and the data transmitting tasks can not be ignored. And this is also one of the reason why we need the Real-time Scheduling Layer. We assume the executing time of sensor sampling or data transmitting task takes one Real-time Scheduling interval for simplicity reason.

For Real-time Scheduling Layer, the energy is from the harvesting device or the primary battery, and the energy is stored into capacitors in each Real-time Scheduling interval. We can assume there is no energy lost by storing energy into capacitors.

Our objective is to properly schedule the tasks such that sufficient good services are provided by the sensor node given the available energy. And in fact, we define the quality of services provided by the sensor node as the evenly of operating tasks of the devices in L intervals. For example, when there are 10 Real-time Scheduling intervals in one Application Rate interval, and the operational rate of the CO sensor is 5. Then we say the service of CO sampling is good if these 5 samples are evenly distributed into the 10 Real-time Scheduling intervals, like $\{1, 0, 1, 0, 1, 0, 1, 0, 1, 0\}$ or $\{0, 1, 0, 1, 0, 1, 0, 1, 0, 1\}$ where 1 means taking a sample and 0 means taking no sample in a specific Real-time Scheduling interval.

When there is no device is operating in Real-time Scheduling interval T_j, we say this is a *no_task_executing_interval*. In fact, we can put the whole system into sleep mode in the *no_task_executing_interval* and hence energy is saved. However, this is contradict to the evenly distribution of the tasks.

We model the Real-time Scheduling Layer in to a CSP as follows:

Variables:

$index_{CO}$: An array with size L. When value is 1, it means the specific task is executed in this Real-time Scheduling interval. Otherwise, no task is executed.

$index_{SO2}$: An array with size L. When value is 1, it means the specific task is executed in this Real-time Scheduling interval. Otherwise, no task is executed.

$index_{O3}$: An array with size L. When value is 1, it means the specific task is executed in this Real-time Scheduling interval. Otherwise, no task is executed.

$index_{NO2}$: An array with size L. When value is 1, it means the specific task is executed in this Real-time Scheduling interval. Otherwise, no task is executed.

$index_{PM}$: An array with size L. When value is 1, it means the specific task is executed in this Real-time Scheduling interval. Otherwise, no task is executed.

$index_{TEMP}$: An array with size L. When value is 1, it means the specific task is executed in this Real-time Scheduling interval. Otherwise, no task is executed.

$index_{RH}$: An array with size L. When value is 1, it means the specific task is executed in this Real-time Scheduling interval. Otherwise, no task is executed.

$index_{RL}$: An array with size L. When value is 1, it means the specific task is executed in this Real-time Scheduling interval. Otherwise, no task is executed.

Domains:

$D(index_{CO}) = D(index_{SO2}) = D(index_{O3}) = D(index_{NO2}) = D(index_{PM}) = D(index_{TEMP}) = D(index_{RH}) = D(index_{RL})$ = [0, 1]

Constraints:

First of all, the scheduling should satisfy the energy constraint. That is the total energy consumption of each Real-time Scheduling interval should not be larger than the energy harvested so far (The remain energy of current interval can be used in the next interval.). The distributing of 0s or 1s for a specific device should be as even as possible. This ensure the quality of services provided by the sensor node. Please refer to the RealtimeScheduling.cpp file in appendix for the detail implementation of the constraints.

IV. EVALUATION

In section, we compare our approach to 3 typical energy management strategies. Improvements in energy saving, operational endurance and quality of services are achieved. In the following experiments, each interval is 10 minutes and total 3 days of data are simulated.

A. Typical Approach 1

One of the most typical approach in energy management for embedded systems using harvesting energy is to average the long term harvested energy (e.g. 1 day) and evenly distribute the energy to each interval. All the harvested energy is first stored into the primary battery. In this approach, the operational endurance is maximized. As illustrated in Figure 2, all the devices are operated in all intervals (rate_CO = 2, rate_SO2 = 2, rate_O3 = 2, rate_NO2 = 2, rate_PM = 2, rate_TEMP = 1, rate_RH = 1, rate_RL = 1). However, such result is achieved when we assume the capacity of the battery is large enough. When the capacity of the battery is small or limited, the operational endurance reduces a lot while large amount of energy is wasted (as shown in Figure 3). The reason is that the energy consumption of the sensor node equals to the long term average harvested energy. In many intervals, the available solar energy is much larger than the energy consumed by the senor node. These residual amount of energy is stored to the battery. When the battery is full, all the residual amount of energy is wasted.

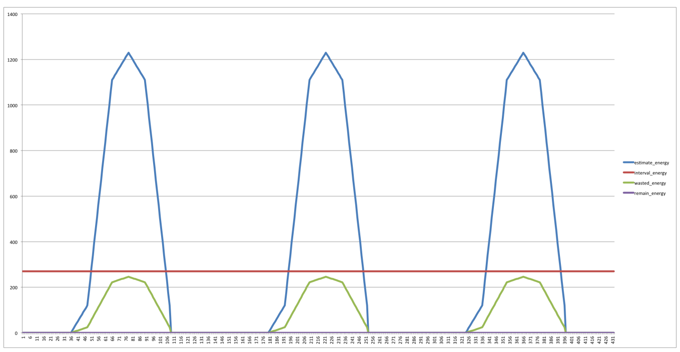

Fig. 2. Illustration of typical approach 1 with infinity battery capacity. X axis is intervals. Y axis is amount of energy. Blue line is the harvested energy. Red line is the energy consumed by devices. Green line is the wasted energy. Purple line the energy remained in battery. Average wasted energy: 70.83. Number of operating intervals per day: 144

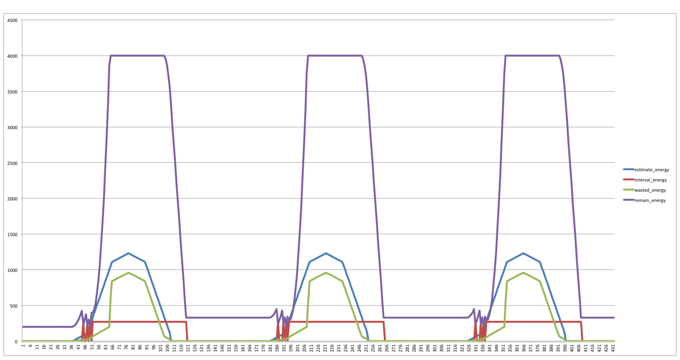

Fig. 3. Illustration of typical approach 1 with small battery capacity. X axis is intervals. Y axis is amount of energy. Blue line is the harvested energy. Red line is the energy consumed by devices. Green line is the wasted energy. Purple line the energy remained in battery. Average wasted energy: 216.42. Number of operating intervals per day: 73

B. Typical Approach 2

In order to reduce the amount of wasted energy, an other typical approach in energy management for embedded systems using harvesting energy is to fully utilize the harvested energy without any storage batteries. At each interval, all the harvested energy are fully utilize in the same interval, i.e. the sensor node consumes all the energy from the solar cell. However, as the energy consumption of the sensor node is not a continuous number, residual energy may be wasted as there is no battery for storage (as shown in Figure 4). Although the wasted energy is much less than that in Figures 2 and 3, the operational endurance of this typical approach is the minimal.

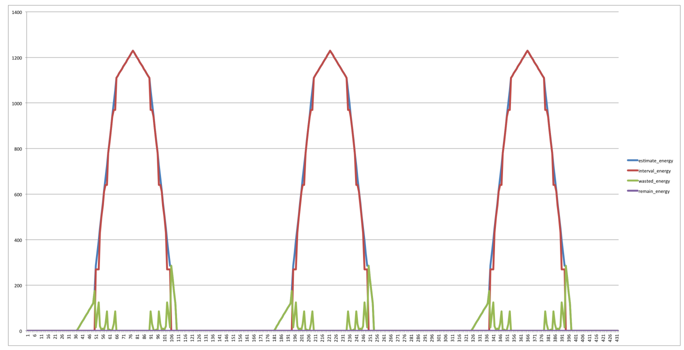

Fig. 4. Illustration of typical approach 2 without battery. X axis is intervals. Y axis is amount of energy. Blue line is the harvested energy. Red line is the energy consumed by devices. Green line is the wasted energy. Purple line the energy remained in battery. Average wasted energy: 19.06. Number of operating intervals per day: 55

C. Typical Approach 3

In this typical approach, the harvest energy is loaded into the battery all the time. The operational rates of all devices are adapted to the nature of the solar energy. That is when there is enough energy, the operational rates of all devices will be increase to fully utilize the energy and vice versa (as shown in Figure 5). Such approach has better performance in operational endurance but larger amount of wasted energy compared with that in Figures 4. This is because in this approach, the harvested energy is loaded into the battery first and the changing efficiency is less than 1. This approach also solved the limited battery capacity issues as in each interval, the harvest energy is fully utilized.

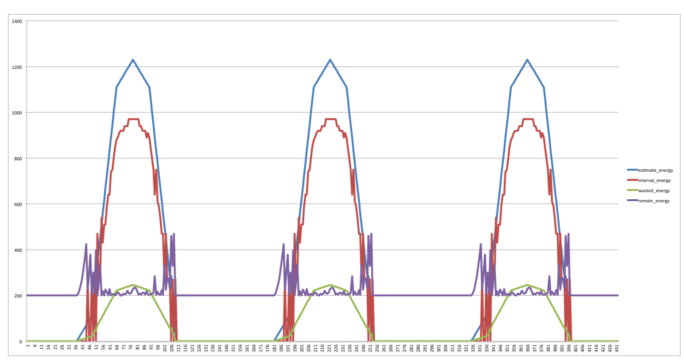

Fig. 5. Illustration of typical approach 3 with small battery. X axis is intervals. Y axis is amount of energy. Blue line is the harvested energy. Red line is the energy consumed by devices. Green line is the wasted energy. Purple line the energy remained in battery. Average wasted energy: 70.69. Number of operating intervals per day: 59

D. Proposed Approach

Our proposed approach utilizes the adapting to nature of solar energy idea from the typical approach 3 while further improving the energy saving ability and operational endurance. In the proposed approach (The Application Rate Layer introduced in Section II), the operational rates of all devices are adapted to the nature of the solar energy. That is when there is enough energy, the operational rates of all devices will be increase to fully utilize the energy and vice versa. In order to further improve the wasted energy saving ability, the proposed approach only charges the battery when residual energy is left in a specific interval (as shown in Figures 6 and 7). Such technique saves a lot of energy because the charging efficiency is typically around 0.8. Note that the wasted energy in Figures 6 and 7) is much less than that in all the previous approaches.

Also, the performance of operational endurance in the proposed approach is also better than that in approach 2 and 3. This is because for every interval of the proposed approach, only the solution that can maximize the total reward of next K intervals is selected. In some senses, the proposed approach puts the future into consideration when calculating current solution. For example, if the harvest energy is decreasing in the future intervals, in order maximize the total reward of the next K intervals, operational rate in current interval should be decreased so that energy can be stored for future usage. By comparing the Figure 6 and Figure 7), we can conclude that larger operational endurance can be achieved by increasing K at the expense of slightly increment in wasted energy.

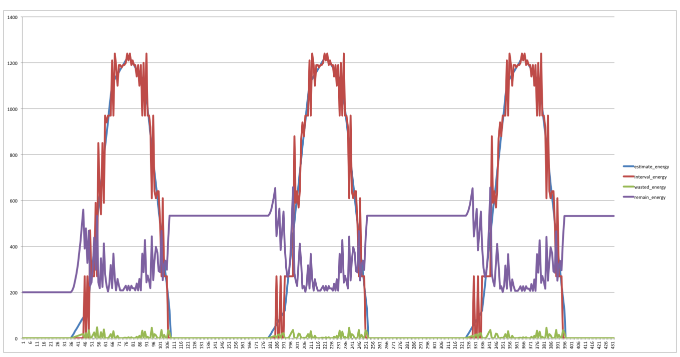

Fig. 6. Illustration of proposed approach with K = 3 using small battery. X axis is intervals. Y axis is amount of energy. Blue line is the harvested energy. Red line is the energy consumed by devices. Green line is the wasted energy. Purple line the energy remained in battery. Average wasted energy: 4.09. Number of operating intervals per day: 61

In order to have good enough services provided by the sensor node, given the operational rate of all the devices in a specific interval, we want the operating schedules of all devices to be as even as possible. Such objective is achieve by the Real-time Scheduling Layer introduced in Section II. In the following, we use one example to illustrate the idea of Real-time Scheduling Layer in one interval.

In this example, the operational rate of CO, SO2, O3, NO2, PM, TEMP, RH and RL devices are $\{6, 6, 6, 6, 5, 1, 1, 2\}$ respectively. The total energy consumption would be 640 and the total harvested energy is also 640. The 10 minutes Application Rate interval is divided into 10 one minute Real-time Scheduling intervals. We assume the executing time of

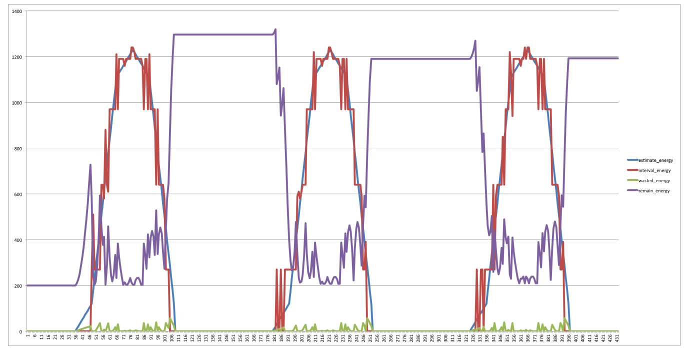

Fig. 7. Illustration of proposed approach with K = 6 using small battery. X axis is intervals. Y axis is amount of energy. Blue line is the harvested energy. Red line is the energy consumed by devices. Green line is the wasted energy. Purple line the energy remained in battery. Average wasted energy: 4.41. Number of operating intervals per day: 62

one task is exactly one minutes and several tasks can be executed at the same Real-time Scheduling interval. Also, we introduce a Gaussian noise *n(0, 10% of harvested energy)* to the harvested energy with based equals to the total harvested energy (as shown in Figure 8).

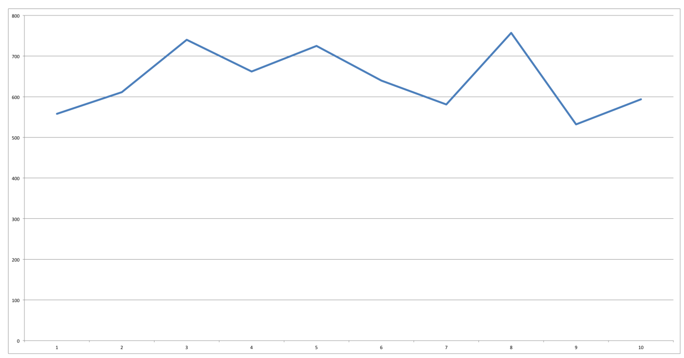

Fig. 8. Illustration of harvested energy in Real-time Scheduling intervals. X axis is intervals. Y axis is amount of energy.

The solution after solving by the constraint solver is CO: $\{1, 1, 0, 1, 0, 1, 0, 1, 0, 1\}$, SO2: $\{1, 1, 0, 1, 0, 1, 0, 1, 0, 1\}$, O3: $\{1, 1, 0, 1, 0, 1, 0, 1, 0, 1\}$, NO2: $\{1, 1, 0, 1, 0, 1, 0, 1, 0, 1\}$, PM: $\{0, 1, 0, 1, 0, 1, 0, 1, 0, 1\}$, TEMP: $\{1, 0, 0, 0, 0, 0, 0, 0, 0, 0\}$, RH: $\{0, 0, 1, 0, 0, 0, 0, 0, 0, 0\}$ and RL: $\{0, 0, 0, 0, 1, 0, 0, 0, 0, 1\}$, where 1 means executing task at that interval and 0 means no executing task at that interval. In human sense, the result is evenly distributed, i.e. good data quality.

V. Conclusion and Discussion

As illustrated in Table I, the proposed approaches (the 5th and 6th approaches) have the minimal wasted energy while the endurances are better than the worst case (the 3rd approach). We can conclude that our approach is able to fully utilized the harvested energy by reducing the amount of wasted energy, and provide sufficient good endurance.

For the Application Rate Layer, the program is run on a note book and the execution time for K = 3 is also most 0 while the execution time for K = 6 is about 10 minutes for total 144 intervals. Such computational time is acceptable as each interval is 10 minutes.

For the Real-time Scheduling Layer, when L = 10, the computational time is about 16 seconds for each interval. This is also acceptable as each interval is 1 minute long.

TABLE I. MY CAPTION

Approach	Wasted Energy	Endurance (0 to 144)
1	70.83	144
2	216.42	73
3	19.6	55
4	70.69	59
5	4.09	61
6	4.41	62

APPENDIX

The implementation of Application Rate Layer in Gencode is in /Source Code/ApplicationLevel.cpp

The implementation of Real-time Scheduling Layer in Gencode is in /Source Code/RealtimeScheduling.cpp

REFERENCES

[1] S. Ó. Murphy, Ó. Manzano, and K. Brown, "Design and evaluation of a constraint-based energy saving and scheduling recommender system," in *Principles and Practice of Constraint Programming*, ser. Lecture Notes in Computer Science, G. Pesant, Ed. Springer International Publishing, 2015, vol. 9255, pp. 687–703. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-23219-5_47

[2] A. Kansal, J. Hsu, S. Zahedi, and M. B. Srivastava, "Power management in energy harvesting sensor networks," *ACM Trans. Embed. Comput. Syst.*, vol. 6, no. 4, Sep. 2007. [Online]. Available: http://doi.acm.org/10.1145/1274858.1274870

[3] C. Moser, L. Thiele, D. Brunelli, and L. Benini, "Adaptive power management in energy harvesting systems," in *Proceedings of the Conference on Design, Automation and Test in Europe*, ser. DATE '07. San Jose, CA, USA: EDA Consortium, 2007, pp. 773–778. [Online]. Available: http://dl.acm.org/citation.cfm?id=1266366.1266532

[4] W. Y. Yi, K. M. Lo, T. Mak, K. S. Leung, Y. Leung, and M. L. Meng, "A survey of wireless sensor network based air pollution monitoring systems," *Sensors*, vol. 15, no. 12, p. 29859, 2015. [Online]. Available: http://www.mdpi.com/1424-8220/15/12/29859