Modular Performance Analysis of Energy-Harvesting Real-Time Networked Systems

Nan Guan[1], Mengying Zhao[2], Chun Jason Xue[2], Yongpan Liu[3] and Wang Yi[4]

[1]Northeastern University, China
[2]City University of Hong Kong, Hong Kong
[3]Tsinghua University, China
[4]Uppsala University, Sweden

***Abstract*—This paper studies the performance analysis problem of energy-harvesting real-time network systems in the Real-Time Calculus (RTC) framework. The behavior of an energy-harvesting node turns out to be a generalization of two known components in RTC: it behaves like an AND connector if the capacitor used to temporally store surplus energy has unlimited capacity and there is no energy loss, while it behaves like a greedy processing component (GPC) if the size of the capacitor is zero and thus surplus energy is lost or passed to other nodes immediately.**

In this paper, methods are developed to analyze the worst-case performance, in terms of delay and backlog, of energy-harvesting nodes as well as compute upper/lower bounds of their data and energy outputs. Moreover, with the proposed analysis methods, we disclose some interesting properties of the worst-case behaviors of energy-harvesting systems, which provide useful information to guide system design. Experiments are conducted to evaluate our theoretical contributions and also confirm that the disclosed properties are not just the result of our analysis, but indeed hold in realistic system behaviors.

I. Introduction

Cyber-physical systems (CPS) integrate digital computation and physical processes, where sensors, actuators and embedded computers are networked to sense, monitor, and control the physical world [18]. Due to close interaction with physical world, the computation in CPS are typically subject to timing constraints. Violating timing constraints in such system may lead to seriously degrade the quality of service and even lead to catastrophic consequences such as loss of human life.

Many CPS consist of a large number of sensing, computing and actuating devices that are deeply embedded into the surrounding environment. For such systems, it is usually not practical to recharge the devices or replace their batteries. Using environmental regenerative energy, e.g., kinetic, electromagnetic radiation (including photovoltaic), and thermal energy, to power the devices in such systems is a promising approach. There are increasing interests in research of energy-harvesting systems, but little work has been done to analyze their real-time behaviors.

In this paper we study the worst-case performance analysis problem of energy-harvesting real-time networked systems. We use Real-Time Calculus (RTC) as the modeling and analysis framework. RTC has proved to be powerful in performance analysis of real-time networked systems. RTC is rooted in and significantly extends Network Calculus (NC), which has a rich theoretical background [2]. The modular analysis approach of RTC is especially suitable for the design and analysis of large-scale and complex networked systems. More importantly, the variability characterization curves (VCC) used in RTC are suitable to model energy availability, since the energy input is typically unstable and difficult to be characterized by closed-form expressions in energy-harvesting systems.

VCC can model a wide range of infinitely regenerative resources, such as computation and communication time. At the first glance, existing RTC theory seems to be directly applicable to energy-harvesting systems, since energy is also a type of infinite and regenerative resource. However, there is actually a fundamental difference between energy and time: ***energy is storable, but time is not***. Computation time is simply wasted if there is currently no work to do. In contrast, surplus energy may be saved in the capacitor and used in the future. Note that in energy-harvesting systems it is necessary to use capacitors to temporally store energy as the energy input is highly unstable, and the capacitors are usually small as they are used as energy buffers but not the primary energy source. To the best of our knowledge, this is the first work to study the modeling and analysis of systems using resources with limited storage, with general setting in the RTC/NC framework. We introduce a new component ERC (Energy-as-Resource Component), to model an energy-harvesting node in the network, and analyze worst-case performance of networked ERC systems.

It turns out that ERC is a generalization of two important models in RTC: the Greedy Processing Component (GPC) and the AND connector. GPC models processing with traditional resource that are not storable, such as computation time. AND connector is used to model the partnering logic of two input event streams. Data arriving on one input port must be buffered until partner data arrive on the other input port. Partnering data join together and immediately pass the AND connector. When the energy store is of unlimited size, each node in our system behaves like an AND connector: energy can be viewed as type of data and consuming energy to process data is essentially the same as joining data from two input streams. No energy loss occurs since the capacitor is of unlimited size. On the

other hand, when the capacitor is of zero size, each node in our system behaves like a GPC since surplus energy will be dropped immediately. We develop analysis methods for ERC, and particularly:

- We proved that the delay and backlog bounds can be derived in the same way as AND connectors.
- We proved that the remaining energy bounds can be computed in the same way as GPC.
- The data output behavior of ERC is more complex, and we developed several methods to compute the output arrival curves from different perspectives. The combination of these methods let us get good analysis precision with different parameter characteristics.

Moreover, with the developed analysis methods, we observe some interesting properties of ERC:

- With given workload and energy input, the worst-case performance of an ERC only depends on the initial fill level of the capacitor, but not its total size. If the capacitor is initially poorly charged, the worst-case performance will not be improved no matter how much the capacitor size is increased. This suggests a basic rule in the design of such systems: the capacitor should be fully charged before the system starts running.
- If the capacitor is initially fully charged, the maximal and minimal energy loss at an ERC is independent from the capacitor size. This phenomenon is somehow counter-intuitive, since using a large capacitor seems to be able to store more energy and thus reduce energy loss. However, as disclosed by our analysis, although a larger capacitor may reduce energy loss in some time intervals, the global maximum and minimum remains unchanged.
- Although increasing the capacitor of an ERC is always good for improving the worst-case performance of this ERC itself, it may be harmful to the worst-case performance of other ERCs in the network.

Note that the above properties are not only the results of our analysis, but indeed exist in the realistic behaviors of ERC as confirmed by our simulation experiments using RTS Toolbox [4] (which is extended to support the new ERC component). The theoretical results of this paper are implemented in RTC Toolbox [20].

A. Related Work

Existing work on energy harvesting systems can mainly be summarized into three categories pertaining to energy modeling, energy-harvesting aware system optimization and scheduling analysis and design. A comprehensive survey of energy-harvesting sensor nodes can be found in [17].

Lu et al. [11] proposed an accurate method for modeling energy dissipation and constructed a runtime scheme to predict future harvested energy for energy-harvesting systems. In order to fully utilize the harvested energy, Liu et al. [7] proposed to manage system power by considering both the battery charging/discharging overheads and the maximum power point of the energy-harvesting devices.

Several work are done on optimization of energy-harvesting powered networked systems. Zhang et al. [22] suggested to maximize the energy preserving within corresponding time intervals in order to improve the sensor nodes' resilience. Instead of focusing on individual sensor nodes, later, they developed algorithms to maximize the minimal energy reservation for all nodes in the network [21]. Zhang et al. [15] proposed a framework to model and adapt system parameters, such as task activation rates, for various optimization objectives in the presence of fluctuating energy-harvesting. The work in [23] and [14] aim to maximize network utility and system reward in energy-harvesting networked systems respectively.

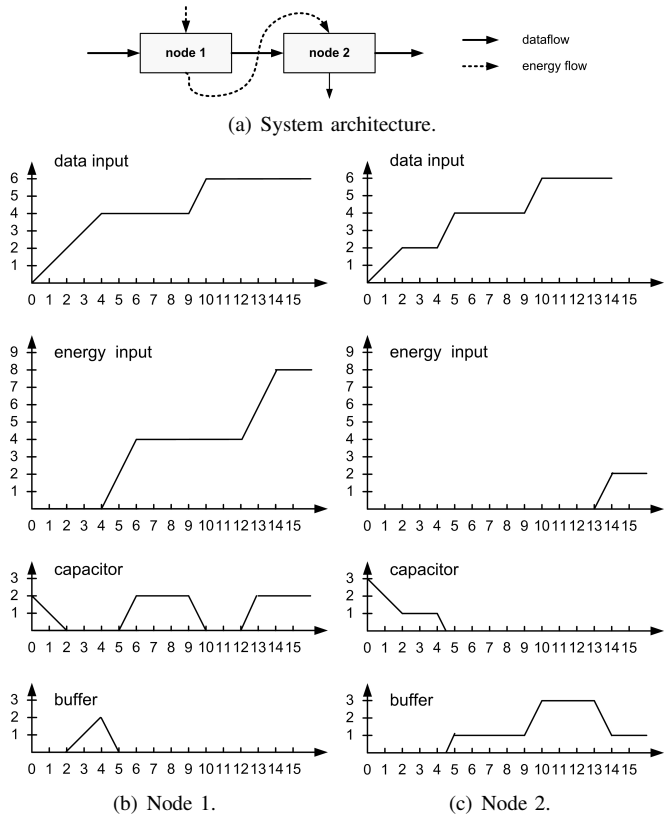
Moser et al. [13] proposed a lazy scheduling algorithm for energy-harvesting systems. To minimize the deadline violation, they proposed optimal online algorithms to dynamically assign energy to arrived tasks. Methods were proposed to test the schedulability and decide the minimum capacity requirement. Liu et al. [9] extended the work in [13] by introducing dynamic voltage scaling and frequency selection. They proposed to dynamically adjust the task execution speed based on the available energy, to efficiently use task slacks and further reduce the deadline miss rate. In the context of voltage selection, the work in [10] and [8] investigated task speedup and slowdown under the energy and timing constraints. Lu and Qiu [12] studied the scheduling and mapping problem on multi-core processors for energy-harvesting systems.

Our work shares similarities with [13], as both of them use RTC as the basic modeling framework. However, a key difference between them is that [13] assumed predefined task deadlines, which significantly simplifies the analysis problem. In contrast, we do not assume any deadline constraint and aim to establish a general foundation for modeling and analysis of energy-harvesting systems in RTC.

A key feature making our problem difficult is energy loss due to the limited-size capacitor (otherwise it can be analyzed in the same way as AND connectors [19]). In Network Calculus theory, there are methods to analyze the so-called *lossy* systems [2], [3], [1], where the workload buffer is of limited size and input data are dropped when the buffer overflows (while in our problem the energy capacitor is limited). Unfortunately, the analysis techniques for these lossy systems are not applicable to our problem. Specifically, in these lossy systems the processing operator enjoys a nice property of being *upper semi-continuous*, and thus the system analysis can be performed by the closure of the operator with certain shifts. However, this property does not hold for our energy-harvesting node with limited-size capacitors. To the best of our knowledge, this work is the first to study general systems with limited-size resource buffer in the RTC/NC theory framework.

II. System Model

We consider a networked system consisting of a number of nodes. Each node is powered by an energy generator, e.g., a solar panel or a mini wind turbine. Each node i is also equipped with a *capacitor* of size M_i, to temporally store surplus energy. Several nodes may share the same energy generator (while each of them has its own capacitor). In this case, we assume these nodes are prioritized to receive energy from the generator: at any time point, the generator



(a) System architecture.

(b) Node 1. (c) Node 2.

Figure 1. An example illustrating the system model.

only charges the highest-priority nodes's capacitor. We assume the energy are dispatched to different nodes in a prioritized manner rather than a controlled shared manner since (i) this is good for system predictability (analog to the fixed-priority scheduling with CPU time as resource), and (ii) in practice this can be implemented by simple circuits (details are shown in the appendix).

A dataflow traverses through multiple nodes. For simplicity, we assume time to be continuous and use a fluid workload model. In real systems, there is always a minimum granularity in both time and data (bit, word, cell or packet). In that case the theory of this paper can also be applied by taking into account the extra costs in both space and time due to the minimum granularity. It takes a constant amount of energy for the node to process one unit of data. When the capacitor is empty and there is no energy input, the node will not process any data and all the input data are stored in the memory buffer, and wait until there is available energy. Data stored in the memory buffer are processed in the FIFO order. There is no predefined limitation of the data buffer size.

As in traditional RTC, we do not make any particular assumption to network topology, except that the network is acyclic. Nevertheless, the techniques of this paper can be generalized to handle cyclic networked systems by the iterative approach as introduced in [6].

Figure 1 shows an example to illustrate the system model described above, where the output data stream is the input data stream of nodes 2, and the remaining energy of node is the energy input of node 2. Node 1 has a capacitor of size $M_1 = 2$ and node 2 has a capacitor of size $M_2 = 3$.

The problem to solve in this paper is to compute upper bounds of (i) the delay at each node (the time between the arrival time of a datum and the time when it is processed) and (ii) the buffer requirement at each node.

III. RTC Framework

A. Arrival and Service Curves

RTC uses *variability characterization curves* (*curves* for short) to describe timing properties of event streams and available resource:

Definition 1 (Arrival Curve)**.** *Let $R(s,t]$ denote the total amount of requested capacity to process in time interval $(s,t]$. Then, the corresponding upper and lower arrival curves are denoted as α^u and α^l, respectively, and satisfy:*

$$\forall s < t, \quad \alpha^l(t-s) \le R(s,t] \le \alpha^u(t-s) \tag{1}$$

where $\alpha^u(0) = \alpha^l(0) = 0$.

Definition 2 (Service Curve)**.** *Let $C(s,t]$ denote the amount of data that the resource can process in time interval $(s,t]$. Then, the corresponding upper and lower service curves are denoted as β^u and β^l, respectively, and satisfy:*

$$\forall s < t, \quad \beta^l(t-s) \le C(s,t] \le \beta^u(t-s) \tag{2}$$

where $\beta^u(0) = \beta^l(0) = 0$.

B. Greedy Processing Component (GPC)

A widely used abstract component used in RTC is the Greedy Processing Component (GPC) . A GPC processes data from input (described by arrival curves α^u and α^l) in a greedy fashion, as long as it complies with the availability of resources (described by service curves β^u and β^l). A GPC produces an output data stream, described by output arrival curves α'^u and α'^l, and output remaining service, described by remaining service curves β'^u and β'^l:

$$\alpha'^u_{gpc}(\Delta) = \min\{\sup_{0\le\lambda}\{\inf_{0\le\mu\le\lambda+\Delta}\{\alpha^u(\mu) + \beta^u(\lambda+\Delta-\mu)\} - \beta^l(\lambda)\}, \beta^u(\Delta)\}$$

$$\alpha'^l_{gpc}(\Delta) = \min\{\inf_{0\le\mu\le\Delta}\{\sup_{0\le\lambda}\{\alpha^l(\mu+\lambda) - \beta^u(\lambda)\} + \beta^l(\Delta-\mu)\}, \beta^l(\Delta)\}$$

$$\beta'^u_{gpc}(\Delta) = \inf_{\Delta\le\lambda}\{\beta^u(\lambda) - \alpha^l(\lambda)\}^+$$

$$\beta'^l_{gpc}(\Delta) = \sup_{0\le\lambda\le\Delta}\{\beta^l(\lambda) - \alpha^u(\lambda)\}$$

where $\{a\}^+$ means $\sup\{a, 0\}$.

The amount of data in the input buffer, i.e., the backlog, can be bounded by $\max(0, V(\alpha^u, \beta^l))$, where $V(f,g)$ gives the maximal vertical distance from curve f to curve g:

$$V(f,g) \triangleq \sup_{\lambda\ge0}\{f(\lambda) - g(\lambda)\}$$

The delay of data can be bounded from above by $H(\alpha^u, \beta^l)$, where $H(f,g)$ gives the maximal horizontal distance from curve f to curve g

$$H(f,g) \triangleq \sup_{\lambda\ge0}\{\inf\{\varepsilon \ge 0 : f(\lambda) \le g(\lambda+\varepsilon)\}\}$$

Multiple GPCs can be connected into a network to model systems with resource sharing and networked structures. The output arrival and resource curves of one GPC are used as the input for the analysis of the downstream nodes along the data and resource flow.

C. AND Connector

Another useful abstract component is the AND connector [19], which combines two input data streams into a single combined stream. Data arriving on one input stream must be buffered until partner data arrive on the other input stream. Partnering data join together and immediately pass the AND connector, and consequently either of the internal buffers is empty at any point of time. The two input data streams are characterized by arrival curves α_1^u, α_1^l and α_2^u, α_2^l, then the output data stream can be bounded by arrival curves

$$\alpha_{and}^u = \max(\min(\sup_{\lambda \geq 0}\{\alpha^u(\Delta+\lambda) - \gamma^l(\lambda)\} + B_1 - B_2, \gamma^u(\Delta)), \min(\sup_{\lambda \geq 0}\{\gamma^u(\Delta+\lambda) - \alpha^l(\lambda)\} + B_2 - B_1, \alpha^u(\Delta)))$$

$$\alpha_{and}^l = \max(\min(\inf_{\lambda \geq 0}\{\alpha^l(\Delta+\lambda) - \gamma^u(\lambda)\} + B_1 - B_2, \gamma^l(\Delta)), \min(\sup_{\lambda \geq 0}\{\gamma^l(\Delta+\lambda) - \alpha^u(\lambda)\} + B_2 - B_1, \alpha^l(\Delta)))$$

where B_1 and B_2 denote the initial buffer fill level of the two input streams ($\min(B_1, B_2) = 0$).

The delay of data at two inputs is bounded by

$$d_{max,1} \leq H(\alpha_1^u + B_1, \alpha_2^l + B_2)$$
$$d_{max,2} \leq H(\alpha_2^u + B_2, \alpha_1^l + B_1)$$

and the backlog at the two input buffers are bounded by

$$b_{max,1} \leq \max(0, V(\alpha_1^u + B_1, \alpha_2^l + B_2))$$
$$b_{max,2} \leq \max(0, V(\alpha_2^u + B_2, \alpha_1^l + B_1))$$

D. Our Model Generalizes both GPC and AND

We model each node in the energy-harvesting system by an abstract component ERC (Energy-as-Resource Component) in the RTC framework, and connect multiple ERCs to model the overall network. In the following, we first discuss the similarity and difference between GPC and AND, and then show that ERC is actually a generalization of both GPC and AND.

We can view the resource input of GPC as a kind of data, and interpret the semantics of GPC in a similar way to AND: GPC has two data input ports, and partnering data from two input ports join together and immediately pass the GPC. However, different from AND, one of the input port of GPC (corresponding to the resource input in the original interpretation) has no buffer, while the other input still has unbounded buffer as in AND. When a datum arrives at the buffer-less input port, if the buffer at the other input is empty, this newly arrived datum is dropped. Otherwise, the newly arrived datum will join with a partner in the other input's buffer and pass the GPC immediately.

On the other hand, we can also view one of the two input streams of AND as a kind of resource, and the data from the other input port is processed only if there is available resource.

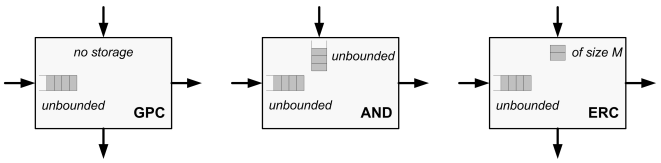


Figure 2. Illustration of GPC, AND and ERC.

However, different from GPC, the resource can be stored in a buffer if there is currently no data to process. There is no limitation of the amount of resource stored in the buffer, so the resource may be infinitely accumulated if there are fewer data than resource in the long term. In summary, the essential difference between GPC and AND lies in whether it has the storage capability at the "resource" input port.

Now we look at ERC, in which the resource input port has a capacitor of a certain size M. So both GPC and AND can be viewed as special cases of ERC: the behavior of ERC is the same as GPC if $M = 0$, while is the same as AND if $M \to +\infty$ (or M is sufficiently large and it is guaranteed that no energy loss occurs), as shown in Figure 2.

In the following sections, we develop techniques to analyze the delay bound, backlog bound and compute the output arrival curves and remaining service curves of ERC. Our analysis techniques is the same as established analysis methods of GPC when $M = 0$, and the same as that of AND when M is sufficiently large.

IV. Delay and Backlog Analysis

In this section, we present how to bound the the maximal delay and backlog of an ERC.

We use $R'(s,t]$ to denote the amount of output data created in $(s,t]$, and use $C'(s,t]$ to denote the amount of remaining energy in $(s,t]$. We use $cap(t)$ to denote the amount of energy stored in the capacitor at time t, and $buf(t)$ the number of unprocessed data backlogged in the buffer at time t.

We assume the system starts running at an arbitrarily early time point p (i.e., $p \to -\infty$). We further assume the buffer is initially empty $buf(p) = 0$, and the initial fill level of the capacitor is $cap(p) = M^0 \leq M$. We first introduce some basic properties.

Lemma 1. *At any time point, at most one of the buffer and the capacitor is not empty.*

Lemma 2. *For any time interval $(s,t]$ it holds:*

$$R'(s,t] = R(s,t] + buf(s) - buf(t)$$

Lemma 3. *For any time interval $(s,t]$, it holds:*

$$R'(s,t] = C(s,t] + cap(s) - cap(t) - C'(s,t]$$

The correctness of the above lemmas is easy to verify and complete proofs are omitted.

Theorem 1. *The maximal backlog of input data is bounded from above by* BCK*:*

$$\mathsf{BCK} = \max(0, V(\alpha^u, \beta^l + M^0)) \tag{3}$$

Proof. First, it is trivial $\mathsf{BCK} \geq 0$. In the following we prove the backlog is bounded by $V(\alpha^u, \beta^l + M^0)$.

We prove by contradiction, assuming at some time points the backlog is strictly larger than $V(\alpha^u, \beta^l + M^0)$. Let t be the earliest one among such time points and thus $cap(t) = 0$. Let s be the earliest time point before t such that the capacitor fill level is at most M^0 at any time point in $(s, t]$, and thus we have $C'(s, t] = 0$, $cap(s) = M^0$ and $buf(s) = 0$. Note that since $cap(p) = M^0$ and $cap(t) = 0$, there must exist such an s. Then by Lemma 2 and 3 we have

$$\begin{aligned} buf(t) &= R(s,t] + buf(s) - C(s,t] - cap(s) + cap(t) + C'(s,t] \\ &= R(s,t] - C(s,t] - M^0 \leq V(\alpha^u, \beta^l + M^0) \end{aligned}$$

which contradicts $buf(t) > V(\alpha^u, \beta^l + M^0)$. □

Theorem 2. *The maximal delay of data is bounded from above by* DLY*:*

$$\mathsf{DLY} = H(\alpha^u, \beta^l + M^0)$$

Proof. Let r be the arrival time of an arbitrary piece of datum. Let s be the latest time point no later than r such that $cap(s) = M^0$. Note there must exist such an s since $cap(p) = M^0$. Let $\lambda' = r - s$.

Let $\varepsilon^* = H(\alpha^u, \beta^l + M^0)$, so we have

$$\varepsilon^* \geq \inf\{\varepsilon \geq 0 : \alpha^u(\lambda') \leq \beta^l(\lambda' + \varepsilon) + M\}$$

so there exists $\epsilon \leq \varepsilon^*$ such that

$$\alpha^u(\lambda') \leq \beta^l(\lambda' + \epsilon) + M^0$$

and since $R(s, r] \leq \alpha^u(\lambda')$ and $C(s, r + \epsilon] \geq \beta^l(\lambda' + \epsilon)$, we have

$$R(s, r] \leq C(s, r + \epsilon] + M^0$$

Since $buf(s) = 0$, we know all the data released in $(s, r]$ have been processed by time $r + \epsilon$, so the delay of the considered datum is bounded by $\epsilon \leq H(\alpha^u, \beta^l + M^0)$. □

We can see that the backlog and delay bounds of ERC are computed in exactly the same way as that of the AND connector, where input port 1 corresponds to the data stream, input port 2 corresponds to energy resource, $B_1 = 0$ and $B_2 = M^0$.

Worst-Case Performance Depends on M^0, but not M

Theorem 1 and 2 imply an interesting phenomenon: given fixed α^u and β^l, the backlog and delay bounds only depend on the initial level M^0 of the capacitor, but is independent of the total capacitor size M. Note that this phenomenon is not only a result of our analysis, but indeed exists in the real system behavior. In Section VII we will further discuss this phenomenon with simulation experiments.

This phenomenon gives us important guidance in designing such systems. To optimize the worst-case performance of a component, the system designer should increase the initial fill level of the capacitor. If the initial fill level of the capacitor is low, the worst-case performance won't be improved no matter how much we increase the size of the capacitor. Therefore, a basic rule for any reasonable design of such systems is to fully charge the capacitor before the system starts running. In the remaining of this paper, we assume the systems obey the above rule, i.e., $M^0 = M$.

V. Remaining Service Curves

In this section, we present how to compute the remaining service curves of an ERC.

Lemma 4. *The system starts running at an arbitrarily early time point p with $cap(p) = M$ and $buf(p) = 0$. For any time point t it holds*

$$C'(p,t] = \sup_{p \leq u \leq t} \{C(p,u] - R(p,u]\}^+ \tag{4}$$

Proof. We let u be the latest time point earlier than t such that $cap(u) = M$. Applying Lemma 2 and 3 to $(p, u]$ gives

$$C(p,u] + cap(p) - cap(u) - C'(p,u] = R(p,u] + buf(p) - buf(u)$$

By $cap(p) = cap(u) = M$ and $buf(p) = buf(u) = 0$, this can be written as $C'(p,u] = C(p,u] - R(p,u]$. On the other hand, since the capacitor is not full in $(u, t]$, we have $C'(p,t] = C'(p,u]$. In summary, we have

$$C'(p,t] = C(p,u] - R(p,u] \tag{5}$$

In the following we prove that

$$\forall v \in (p,t] : C(p,v] - R(p,v] \leq C(p,u] - R(p,u] \tag{6}$$

We prove in two cases:

- $v < u$. Since $buf(u) = 0$, so applying Lemma 2 and 3 to $(v, u]$ gives

$$\begin{aligned} &C(v,u] + cap(v) - cap(u) - C'(v,u] = R(v,u] + buf(v) \\ &\Rightarrow C(v,u] + cap(v) - M - C'(v,u] = R(v,u] + buf(v) \end{aligned}$$

  Since $cap(v) - M \leq 0$, this is written as

$$\begin{aligned} C(v,u] &\geq R(v,u] \\ \Rightarrow C(p,u] - R(p,u] &\geq C(p,v] - R(p,v] \end{aligned}$$

- $v > u$. In this case, By the definition of u and $v > u$, we know the capacitor is not full in $(u, v]$, so $C'(u,v] = 0$. We also know $buf(u) = 0$ since $cap(u) = M$. Applying Lemma 2 and 3 to $(u, v]$ gives

$$\begin{aligned} C(u,v] + M - cap(v) &= R(u,v] - buf(v) \\ \Rightarrow C(u,v] &\leq R(u,v] \\ \Rightarrow C(p,u] - R(p,u] &\geq C(p,v] - R(p,v] \end{aligned}$$

In summary, we have proved (6) for both cases. The lemma is proved by combining (5) and (6). □

Although proved in a different way, the result of Lemma 4 is exactly the same as its counterpart in the analysis of GPC [19], and thus the remaining service curves can be computed in the same way as in GPC.

Theorem 3. *The remaining energy is bounded from above and from below by the curves β'^u and β'^l,*

$$\beta'^u(\Delta) = \beta'^u_{gpc}(\Delta)$$
$$\beta'^l(\Delta) = \beta'^l_{gpc}(\Delta)$$

where β'^u_{gpc} and β'^l_{gpc} are defined in Section III-B.

Proof. With Lemma 4, the proof is the same as that for β'^u_{gpc} and β'^l_{gpc} in [19]. □

Remaining Energy Bounds are Independent of M

Theorem 3 suggests an interesting (and somehow counter-intuitive) phenomenon: both upper and lower bounds of remaining energy in a time interval is independent of the size of the capacitor.

An ERC passes remaining energy only when the capacitor is full. The capacitor is full when the system starts running at p. We let the resource come as fast as possible since the system starts, and data come as slow possible, which leads to the same maximal remaining energy as in GPC.

However, it is a bit counterintuitive that the lower bound of the remaining energy also does not change with a larger capacitor. Intuitively, using a large capacitor to store surplus energy seems to be able to decrease energy loss, and therefore the remaining energy of an ERC should decrease. Indeed, a larger capacitor may reduce energy loss in some time intervals, however, the global minimum still keeps unchanged. Note that this phenomenon is not only the result of our analysis, but indeed exists in the real system behavior. In Section VII we will further discuss this phenomenon with simulation experiments.

VI. Output Arrival Curves

The analysis of output arrival curves of ERC is more complicated. Unlike the remaining service curves that can be computed in exactly the same way as GPC, the characteristics of the output data stream may be very different under different parameter configurations. To see this, we consider the following special cases:

1) The capacitor never underflows. In this case, there is available energy whenever data arrive, so the output arrival curves are the same as the input arrival curves.
2) The capacitor size is 0. In this case, surplus energy is passed immediately and an ERC behaves in the same way as a GPC.
3) The capacitor never overflows. In this case, there is no energy loss and an ERC behaves in the same way as an AND connector.

The above observations suggest us to analyze ERC from different perspectives. In the following, we present three methods to compute the output arrival curves. Each method is designed for systems that behaves closer to one of the above special cases. As will be shown in Section VII, these three methods are in general incomparable, i.e., each method may performs better than others with different system parameters. Therefore, unifying the three methods gives us more precise results than any of them.

A. Method I: Comparing with Input Curves

The first method is designed for systems that behaves closer to the first special case in above. Recall the backlog and delay bounds in Theorem 1 and 2:

$$\mathsf{DLY} = H(\alpha^u, \beta^l + M^0) \tag{7}$$

When M^0 is the same as or larger than $V(\alpha^u, \beta^l)$ (the maximal vertical distance from α^u and β^l), $\mathsf{BCK} = \mathsf{DLY} = 0$ and thus the output arrival curve perfect matches the input arrival curve. When M^0 is a bit smaller than $V(\alpha^u, \beta^l)$, a datum may experience a small delay and there will be a small variance between the input and output arrival curves. By adding such variance into the input arrival curves we can compute the output arrival curves as stated in the following theorem.

Theorem 4. *The output data stream is bounded from above and from below by the curves α'^u and α'^l, where $\forall \Delta \geq 0$:*

$$\alpha'^u_I(\Delta) = \alpha^u(\Delta + \mathsf{DLY}) \tag{8}$$
$$\alpha'^l_I(\Delta) = \alpha^l(\Delta - \mathsf{DLY}) \tag{9}$$

The idea for proving the theorem is the same as in [16]. A complete proof is omitted here.

For systems where DLY and BCK is small, the output curves α'^u_I and α'^u_I computed in Theorem 4 are rather precise. However, as DLY and BCK increase, Theorem 4 becomes less and less precise. When the long-term slope of β^l is smaller than α^u, DLY and BCK are unbounded, with which α'^u_I is $+\infty$ and α'^l_I is 0. Experimental evaluations showing this trend will be presented in Section VII.

B. Method II: GPC-based Analysis

The second method is designed for systems that behave closer to the second special case listed in the beginning of Section VI.

Method I in last subsection performs good when the capacitor is sufficiently large (to compensate $V(\alpha^u, \beta^l)$, so that the output follows the input as much as possible). In the following, we present Method II that performs good when the capacitor is very small. In this case, the behavior of ERC is closer to GPC, and the output arrival curves in Method II is derived in a way similar to that in GPC [19].

Theorem 5. *The output data stream is bounded from above and from below by curves α'^u and α'^l, where $\forall \Delta \geq 0$:*

$$\alpha'^u_{II}(\Delta) = \min\left(\sup_{0\leq\lambda}\left\{\inf_{0\leq\theta\leq\Delta}\{x(\lambda,\theta)+z(\theta)\}\right\}, \beta^u(\Delta)+M\right)$$
$$\alpha'^l_{II}(\Delta) = \min\left(\inf_{0\leq\theta\leq\Delta}\left\{\sup_{0\leq\lambda}\{y(\lambda,\theta)\}, \beta^l(\Delta)\right\}\right) - M$$

where

$$x(\lambda,\theta) = \alpha^u(\lambda+\theta) + \beta^u(\Delta-\theta) - \beta^l(\lambda)$$
$$y(\lambda,\theta) = \alpha^l(\lambda+\theta) + \beta^l(\Delta-\theta) - \beta^u(\lambda)$$
$$z(\theta) = \min(M, \max(0, \sup_{\xi\leq\theta}\{\beta^u(\xi) - \alpha^l(\xi)\}))$$

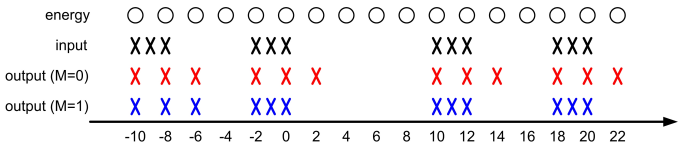


Figure 3. When the capacitor size increases, the upper bound of output may become higher and the lower bound may become lower.

The proof of this theorem is rather long and we present it in the appendix.

Essentially, the output curves obtained by Theorem 5 adds/substracts the amount of energy stored in the capacitor on the basis of the upper and lower output arrival curves of GPC. Figure 3 shows an example to illustrate this. For simplicity, in the example we view both energy and data as discrete events. Nevertheless it is the principle for continuous energy and data inputs. When $M = 0$, output data are created at the same rate as the energy input. When $M = 1$, the energy stored in the capacitor can lead to a more bursty output: the maximal amount of output data in a time interval of length $2+\varepsilon$ is 3 when $M = 1$, while is 2 when $M = 0$. Note that the lower bound also may become lower: the minimal amount of output data in a time interval of length $10-\varepsilon$ is 0 when $M = 1$ (time interval $(0, 10)$), while is 1 when $M = 0$.

Note that we can already get a safe upper output arrival curve of ERC by directly adding M to the upper output arrival curve of GPC. However, $\alpha'^{u}_{II}(\Delta)$ in Theorem 5 is a bit more precise than that as $z(\theta)$ is potentially smaller than M. This captures the following fact: if there are backlogged data at the start of a time interval (and thus the capacitor is empty), the amount of surplus energy stored into the capacitor in this time interval (i.e., the amount of extra energy to forward more data comparing with in GPC) is bounded by the difference between the maximal energy input and the minimal data input, which is potentially smaller than M. In this case, it is will be pessimistic to simply add M to the upper output arrival curve of GPC.

C. Method III: AND-based Analysis

The third method is designed for systems whose behaviors are closer to the third special case: ERC behaves like AND when the capacitor never overflows. In order to analyze ERC in the way analog to AND, we shall zoom in the internal structure of ERC, as shown in Figure 4. We can view ERC as an AND connector equipped with a controller. The controller guarantees the fill level of the buffer at the energy input port in the AND connector never exceeds M. In this way, a part of energy input is dropped (sent to the output port), and the remaining part is sent to the AND connector. We use $E(s,t]$ to denote the amount of energy that are actually sent into the AND connector, and γ^u and γ^l are the upper and lower curves bounding E in the interval domain. If γ^u and γ^l are known, then the output arrival curves can be computed in the same way as α_{and} in Section III-C, with α and γ as two inputs. Now the question is how to compute γ^u and γ^l.

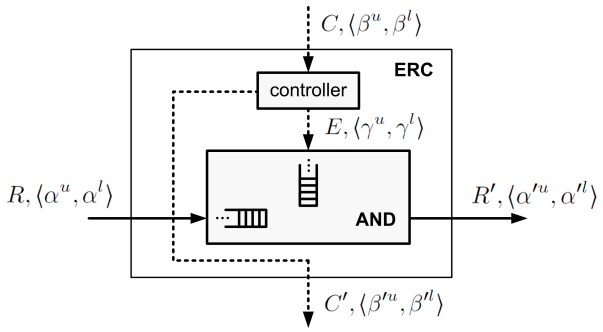


Figure 4. Modeling the internal structure of ERC based on AND.

Since the energy sent into ERC is either passed to the energy output or sent to the AND connector, we have the following relation:

Lemma 5. *For any time interval $(s,t]$ it holds:*

$$C(s,t] = E(s,t] + C'(s,t]$$

By transferring this relation to the interval domain, we can compute γ by β and β' as follows:

Lemma 6. *For any time interval $(s,t]$ with $t-s = \Delta$, $E(s,t]$ is upper and lower bounded by*

$$\gamma^u(\Delta) = \inf_{\lambda \geq \Delta} \{\beta^u(\lambda) - \beta'^l(\lambda)\}$$
$$\gamma^l(\Delta) = \sup_{0 \leq \lambda \leq \Delta} \{\beta^l(\lambda) - \beta'^u(\lambda)\}$$

Proof. For any time interval $(a,b]$ with $a \leq s$ and $b \geq t$, it holds $E(s,t] \leq E(a,b]$, so we have

$$E(s,t] = \inf_{a \leq s \wedge b \geq t} \{E(a,b]\} = \inf_{a \leq s \wedge b \geq t} \{C(a,b] - C'(a,b]\}$$
$$\leq \inf_{\lambda \geq \Delta} \{\beta^u(\lambda) - \beta'^l(\lambda)\}$$

For any time interval $(c,d]$with $s \leq c \leq d \leq t$, it holds $E(s,t] \geq E(c,d]$, so we have

$$E(s,t] = \sup_{s \leq c \leq d \leq t} \{E(c,d]\} = \sup_{s \leq c \leq d \leq t} \{C(c,d] - C'(c,d]\}$$
$$\geq \sup_{\lambda \leq \Delta} \{\beta^l(\lambda) - \beta'^u(\lambda)\}$$

□

Note that β'^u and β'^l can be computed by Theorem 3. Then we can compute the output arrival curves in the same way as AND connector:

Theorem 6. *The output data stream is bounded from above and below by the curves α'^u_{III} and α'^l_{III}, where $\forall \Delta \geq 0$:*

$$\alpha'^u_{III}(\Delta) = \alpha'^u_{and}(\Delta)$$
$$\alpha'^l_{III}(\Delta) = \alpha'^l_{and}(\Delta)$$

where α'^u_{and} and α'^l_{and} are defined in Section III-C, $\alpha_1 = \alpha$, $\alpha_2 = \gamma$, $B_1 = 0$ and $B_2 = M$.

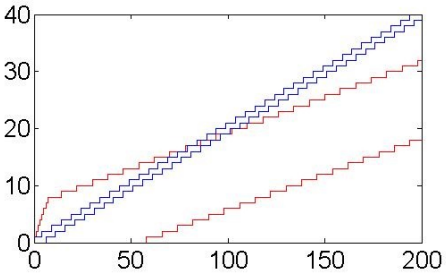


Figure 5. Input arrival (red) and service (blue) curves of the experiments in Section VII-A.

D. Summary

We have presented three methods to compute the output arrival curves. Each of these three methods may outperform the other two under certain circumstances. Finally, combining these three methods yields more precise output arrival curves:

Theorem 7. *The output data stream is bounded from above and below by the curves α'^u and α'^l, where $\forall \Delta \geq 0$:*

$$\alpha'^u(\Delta) = \min(\alpha'^u_I(\Delta), \alpha'^u_{II}(\Delta), \alpha'^u_{III}(\Delta))$$
$$\alpha'^l(\Delta) = \max(\alpha'^l_I(\Delta), \alpha'^l_{II}(\Delta), \alpha'^l_{III}(\Delta))$$

VII. Experiments

The proposed analysis methods of this paper are implemented in RTC Toolbox [20]. We also implement ERC in RTS Toolbox [4] to compare the analysis results with real system behaviors and validate the properties of ERC suggested by the theoretical results. RTS Toolbox is a discrete-event simulator, which can generate discrete event traces from VCC curves, simulate them, and transform the output traces back to VCC curves. Note that as a non-exhausted simulator, RTS Toolbox does not guarantee to find the best/worst-case behavior of the system. So the behavior represented by VCC obtained by RTS Toolbox are under-approximations of the real best/worst-case behavior.

We implement ERC by the logic in Figure 4, based on the build-in AND connector of RTS Toolbox. Since the AND connector accepts discrete events, in our experiments we also use staircase curves for energy inputs, in order to keep the consistency between our analysis and the simulation semantics.

A. Delay and Backlog

We first evaluate how does the worst-case delay and backlog of an ERC change with different capacitor size M and the initial capacitor fill level M^0. We use the input arrival and service curves in Figure 5. Table I shows the worst-case delay and backlog of this component with a fixed initial capacitor fill level M^0=3 and changing capacitor size $M = 3, 6, \cdots, 18$. The backlog and delay values in the rows of "RTC" are obtained by our analysis in Theorem 1 and 2, and the values of "RTS" are obtained by simulation. According to Theorem 1 and 2, the backlog and delay bounds by our analysis only depend on M^0 and thus must be the same with different M. From the table we can see that the maximal backlog and delay observed in the simulation also keep unchanged with different M.

Table II shows the results under the same setting but with a fixed M and changing M^0. The maximal delay and backlog becomes larger as we decrease the initial capacitor fill level, in both our analysis and simulations.

By the above results we can conclude that the worst-case performance only depends on the initial capacitor fill level M^0, but not the total capacitor size M. This is true not only in our analysis results, but also in realistic behaviors of ERC. We have conducted the same experiment with various parameter settings of the input curves, the results of which all suggest the same conclusion.

Table I
Delay and buffer values with changing capacitor capacity.

Fixed initial storage in capacitor: M^0=3							
M		3	6	9	12	15	18
RTC	backlog	4					
	delay	19					
RTS	backlog	4	4	4	4	4	4
	delay	18	18	18	18	18	18

Table II
Delay and buffer values with changing initial storage in capacitor.

Fixed initial level in capacitor: M=10							
M^0		10	8	6	4	2	0
RTC	buffer	0	0	1	3	5	7
	delay	0	0	4	14	24	34
RTS	buffer	0	0	1	3	5	7
	delay	0	0	2	12	22	32

B. Remaining Energy Curves

We experimentally validate the conclusion drawn at the end of Section V: if the capacitor is initially fully charged, the maximal and minimal amount of remaining energy in a time interval of certain length is independent from the capacitor size M.

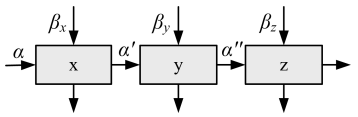
We use the same arrival (red) and service (blue) curves as above for inputs and set $M = 0, 1, \cdots, 10$. We compute the remaining service curves using Theorem 3. As expected, the resulting remaining service curves are all exactly the same.

Then we conduct simulations under the same settings, and transform the resulting energy output sequence into service curves. With all different M values, the obtained service curves also exactly overlap each other.

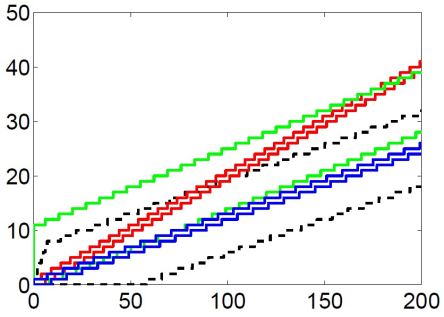
Therefore, we confirm the conclusion that as long as the capacitor is initially fully charged, the maximal and minimal amount of remaining energy in a time interval of certain length is always the same regardless the capacitor size M.

C. Output Arrival Curves

We evaluate the analysis precision of the three methods to compute the output arrival curves. We conduct experiments with two sets of input arrival (red) and service (blue) curves, as shown in Figure 6-(a1) and (b1). For each setting, we change



(a) System Architecture



(b) Input curves

Figure 7. Example for system-level evluation.

the capacitor size $M = 1, 3, 5$. The output arrival curves by Method I is in green, Method II in blue and Method III in red. We also add the curves transformed from the simulation results (black dash lines). From Figure 6 we can see that the three methods are in general incomparable, and combining them is indeed considerably more precise then using any of them independently.

D. System-Level Analysis

We performance analysis and simulation with the system shown in Figure 7-(a). The output arrival curves of node x are used as input arrival curves of node y. The initial input arrival curves α (black dashed) and the three input service curves β_x (red), β_y (green) and β_z (blue) are shown in Figure 7-(b).

Table III shows the backlog and delay bounds by both our analysis and RTS Toolbox simulation with different M_x. The default capacitor size of node x, y and z are 7, 5 and 3 respectively. Then we perform the analysis and run simulation for all the whole system, the obtained delay and backlog bound of of each node by our analysis (in columns under "RTC") and by simulation (in columns under "RTS") are recorded in the row led by $M_x = 7$. Then we decrease M_x by 1 and repeat the above procedure, until $M_x = 0$ are done.

Table III
BACKLOG AND DELAY BOUNDS BY ANALYSIS AND SIMULATION WITH DIFFERENT M_x

M_x	RTC						RTS					
	backlog			delay			backlog			delay		
	x	y	z	x	y	z	x	y	z	x	y	z
7	0	3	7	0	15	50	0	2	4	0	14	25
6	1	3	7	4	15	54	1	2	4	2	13	25
5	2	3	8	9	15	59	2	2	4	7	11	25
4	3	2	8	14	13	58	3	2	4	12	9	25
3	4	2	8	19	12	57	4	2	4	18	9	25
2	5	2	8	24	12	57	5	1	4	22	3	25
1	6	2	8	29	12	57	6	0	3	27	0	23
0	7	2	7	34	10	51	7	0	3	32	0	18

As expected, when increasing the size of M_x, the maximal backlog and delay by both our analysis and simulation decrease. However, as M_x increases, the maximal backlog and delay at ERC y may increase. The reason can be explained as follows. At ERC x the input service curves are less bursty than the input data input (comparing the red line and the black dashed line). When M_x is small, the behavior of the output data is dominated by the resource availability (in short time intervals), and the output arrival curves look more like the input service curve than the input arrival curve. When M_x is large, the behavior of the output data is dominated by the input data arrival, and the output arrival curves look more like the input arrival curve than the input service curve. Therefore, as M_x increases, the output arrival curves of ERC x becomes more bursty, and when they are sent to ERC y, the maximal backlog and delay at ERC y may become larger (in both analysis results and simulation behaviors).

As M_x increase, the maximal backlog and delay observed in the simulation of ERC z also comply with the above discussion. However, the change in corresponding analysis results is not monotonic. The backlog and delay bounds first increase then decrease. This is due to the imprecision of our analysis methods, but not the real ERC behavior.

VIII. Conclusions and Future Work

This paper studied the modeling and analysis of energy-harvesting real-time network systems in the RTC framework. Each node in the system is modeled by an ERC (Energy-as-Resource Component), which has a limited-size capacitor at the energy input. If the capacitor size is 0, ERC behaves just like GPC, where extra energy will be passed immediately. If the capacitor size is sufficiently large and no energy loss occurs, ERC behaves just like the AND connector. Therefore, the analysis problem of ERC is a generalization of, and thus is more difficult than that of GPC and AND connector.

We extend existing RTC framework to analyze ERC. We proved that the delay and backlog bounds of ERC can be analyzed in the same way as AND connectors, and the remaining energy curves can be computed in the same way as GPC. The output arrival curves are computed from different perspectives yield high-quality results with different parameter characteristics.

Based on our analysis, we also observe some interesting properties of ERC. First, the worst-case performance of an ERC only depends on the initial fill level of the capacitor, but not its total size. Second, the capacitor is initially fully charged, the maximal and minimal energy loss at an ERC is independent from the capacitor size. Third, increasing the capacitor of an ERC is always good for improving the worst-case performance of this ERC itself, but may decrease the worst-case performance of other ERCs in the network, and thus be harmful to the overall worst-case performance of the whole network system. These observations are confirmed by experiments with implementation of the analysis methods as well as simulations.

In this paper energy is the only resource constraint to decide when a datum is processed. In the next step we will study the modeling and analysis of real-time network systems where both energy and computation time poses constraints

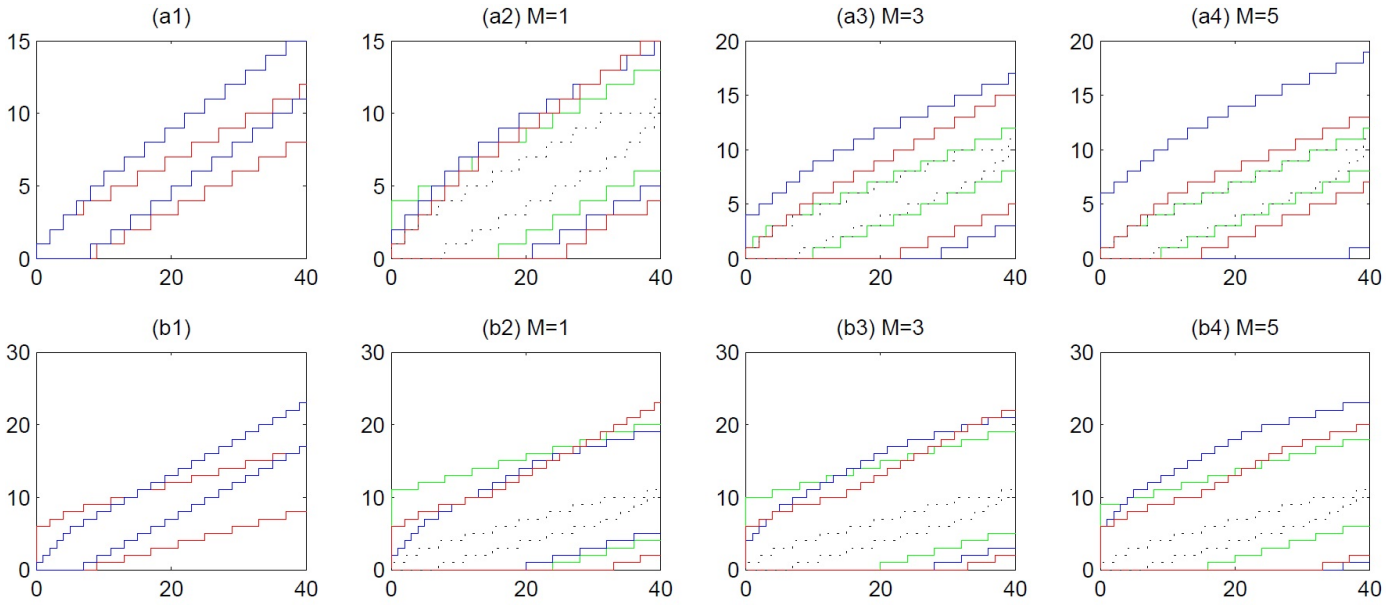


Figure 6. Comparison of the three methods to compute the output arrival curves.

to the processing of data. Another future work is to study holistic properties of ERC networks. A challenge is to apply the *pay-burst-only-once* technique [2] to derive tighter end-to-end delay bounds than summing up the delays of individual components. We also plan to apply the idea of Finitary Real-Time Calculus [5] to improve the analysis efficiency of ERC networks.

REFERENCES

[1] R. Agrawal, R. L. Cruz, C. Okino, and R. Rajan. Performance bounds for flow control protocols. *IEEE/ACM Transactions on Networking*, 1999.

[2] J. L. Boudec and P. Thiran. Network calculus - a theory of deterministic queuing systems for the internet. In *LNCS 2050*. Springer Verlag, 2001.

[3] C.S. Chang. On deterministic traffic regulation and service guarantee: A systematic approach by filtering. *IEEE Transactions on Information Theory*, 1998.

[4] Computer Engineering and ETH Zurich Networks Laboratory (TIK). Real-Time Simulation (RTS) Toolbox, 2009.

[5] Nan Guan and Wang Yi. Finitary real-time calculus: Efficient performance analysis of distributed embedded systems. In *RTSS*, 2013.

[6] Bengt Jonsson, Simon Perathoner, Lothar Thiele, and Wang Yi. Cyclic dependencies in modular performance analysis. In *EMSOFT*, 2008.

[7] Shaobo Liu, Jun Lu, Qing Wu, and Qinru Qiu. Load-matching adaptive task scheduling for energy efficiency in energy harvesting real-time embedded systems. In *International Symposium on Low-Power Electronics and Design*, pages 325–330, 2010.

[8] Shaobo Liu, Jun Lu, Qing Wu, and Qinru Qiu. Harvesting-aware power management for real-time systems with renewable energy. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 20(8):1473–1486, Aug 2012.

[9] Shaobo Liu, Qinru Qiu, and Qing Wu. Energy aware dynamic voltage and frequency selection for real-time systems with energy harvesting. In *Design, Automation and Test in Europe*, pages 236–241, 2008.

[10] Shaobo Liu, Qing Wu, and Qinru Qiu. An adaptive scheduling and voltage/frequency selection algorithm for real-time energy harvesting systems. In *Design Automation Conference*, pages 782–787, 2009.

[11] Jun Lu, Shaobo Liu, Qing Wu, and Qinru Qiu. Accurate modeling and prediction of energy availability in energy harvesting real-time embedded systems. In *International Green Computing Conference*, pages 469–476, Aug 2010.

[12] Jun Lu and Qinru Qiu. Scheduling and mapping of periodic tasks on multi-core embedded systems with energy harvesting. In *International Green Computing Conference and Workshops*, pages 1–6, 2011.

[13] C. Moser, D. Brunelli, L. Thiele, and L. Benini. Real-time scheduling with regenerative energy. In *18th Euromicro Conference on Real-Time Systems*, pages 10 pp.–270, 2006.

[14] C. Moser, Jian-Jia Chen, and L. Thiele. Reward maximization for embedded systems with renewable energies. In *International Conference on Embedded and Real-Time Computing Systems and Applications*, pages 247–256, 2008.

[15] Clemens Moser, Lothar Thiele, Davide Brunelli, and Luca Benini. Adaptive power management in energy harvesting systems. In *Proceedings of the Conference on Design, Automation and Test in Europe*, pages 773–778, 2007.

[16] Simon Perathoner, Nikolay Stoimenov, and Lothar Thiele. Reliable mode changes in real-time systems with fixed priority or edf scheduling. In *In Proc. of the Conference on Design, Automation and Test in*, 2009.

[17] Sujesha Sudevalayam and Purushottam Kulkarni. Energy Harvesting Sensor Nodes: Survey and Implications. *IEEE Communications Surveys & Tutorials*, 13(3):443–461, 2011.

[18] Terry Tidwell, Robert Glaubius, Christopher D. Gill, and William D. Smart. Optimizing expected time utility in cyber-physical systems schedulers. In *RTSS*, 2010.

[19] Ernesto Wandeler. Modular performance analysis and interface-based design for embedded real-time systems. In *PhD thesis, ETHZ*, 2006.

[20] Ernesto Wandeler and Lothar Thiele. Real-Time Calculus (RTC) Toolbox, 2006.

[21] Bo Zhang, R. Simon, and H. Aydin. Harvesting-aware energy management for time-critical wireless sensor networks with joint voltage and modulation scaling. *IEEE Transactions on Industrial Informatics*, 9(1):514–526, Feb 2013.

[22] Bo Zhang, Robert Simon, and Hakan Aydin. Joint voltage and modulation scaling for energy harvesting sensor networks. In *International Workshop on Energy Aware Design and Analysis of Cyber Physical Systems*, 2010.

[23] Bo Zhang, Robert Simon, and Hakan Aydin. Maximum utility rate allocation for energy harvesting wireless sensor networks. In *International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, pages 7–16, 2011.

APPENDIX I: PROOF OF THEOREM 5

A. *Proof of Upper Bound Curve* $\alpha_{II}^{\prime u}$

Proof. We will prove that for any time interval $(s,t]$ with $t-s=\Delta$, it holds $R'(s,t] \le \alpha_{II}^{\prime u}(\Delta)$. We prove two parts:

$$R'(s,t] \le \beta^u(\Delta) + M \tag{10}$$

$$R'(s,t] \le \sup_{0\le\lambda}\left\{\inf_{0\le\theta\le\Delta}\{x(\lambda,\theta)+z(\theta)\}\right\} \tag{11}$$

We first prove (10). By Lemma 3 we have

$$\begin{aligned} R'(s,t] =& C(s,t] + cap(s) - cap(t) - C'(s,t] \\ \le& C(s,t] + M \le \beta^u(\Delta) + M \end{aligned}$$

Then we prove (11), distinguishing two cases: (i) $buf(s) = 0$, (ii) $buf(s) > 0$.

(i) $buf(s) = 0$. In this case by Lemma 2 we know

$$R'(s,t] \le R(s,t] \le \alpha^u(\Delta)$$

In the following we prove $\alpha^u(\Delta) \ge \alpha_{II}^{\prime u}(\Delta)$. Let $\lambda' = \Delta - \theta$, then we have

$$\begin{aligned} &\alpha_{II}^{\prime u}(\Delta) > \sup_{\lambda\ge 0}\{\inf_{\theta\le\Delta}\{x(\lambda,\theta)\}\} \ge \inf_{\theta\le\Delta}\{x(\lambda',\theta)\} \\ =\ & \alpha^u(\Delta) + \beta^u(\lambda') - \beta^l(\lambda') \ge \alpha^u(\Delta) \end{aligned}$$

(ii) $buf(s) > 0$. Let a be the earliest time point no later than s such that the buffer is continuously non-empty in $(a,s]$, and b the earliest time point no later than t such that the buffer is continuously non-empty in $(b,t]$ ($b = t$ if $buf(t) = 0$). Note that $a = b$ if the buffer is continuously non-empty in $(s,t]$.

$$\begin{aligned} R'(s,t] =& R'(a,b] - R'(a,s] + R'(b,t] \\ \le& \sup_{a\le s}\{R'(a,b] - R'(a,s] + R'(b,t]\} \end{aligned}$$

We know the following relations:

1) Since the buffer is empty at both a and b, we have $R'(a,b] = R(a,b]$.
2) Since the buffer is continuously non-empty in $(a,s]$, we have $R'(a,s] = C(a,s]$.
3) Since the buffer is continuously non-empty in $(b,t]$, we have $R'(b,t] = C(b,t]$.

Applying these relations to the above inequality gives

$$R'(s,t] < \sup_{a\le s}\{R(a,b] - C(a,s] + C(b,t]\} \tag{12}$$

In the following, we prove that $\forall b' \in (s,t]$ it holds:

$$R(a,b] + C(b,t] \le R(a,b'] + C(b',t] + z(b'-s) \tag{13}$$

We prove for two cases:

1) $b' \ge b$. By Lemma 2 and 3 we have

$$C(b,b'] + cap(b) - cap(b') - C'(b,b'] = R(b,b'] + buf(b) - buf(b')$$

Since the buffer is empty at b and is continuously non-empty in $(b,b']$, we known $buf(b) = 0$ and $C'(b,b'] = cap(b') = 0$. Therefore, the above equation can be rewritten as

$$R(b,b'] = C(b,b'] + cap(b) + buf(b') \ge C(b,b']$$

by which we know $C(b,t] - C(b',t] \le R(a,b'] - R(a,b]$, and thus (13) holds.

2) $b' < b$. by Lemma 3 and $buf(b) = 0$ we have:

$$\begin{aligned} &C(b',b] + cap(b') - cap(b) - C'(b',b] = R(b',b] + buf(b') \\ &\qquad\Rightarrow\quad R(b',b] \le C(b',b] + cap(b') \end{aligned} \tag{14}$$

Let c be the earliest time point no earlier than s such that $buf(c) = 0$. Since the buffer becomes from non-empty to empty at c, we also know $cap(c) = 0$. Note that there must exist such a point c since $buf(s) > 0$.
Applying Lemma 2 and 3 to $(c,b']$ gives

$$C(c,b'] + cap(c) - cap(b') - C'(c,b'] = R(c,b'] + buf(c)$$

and since $buf(c) = 0$ and $cap(c) = 0$, we have

$$\begin{aligned} cap(b') \le& C(c,b'] - R(c,b']) \le \beta^u(b'-c) - \alpha^l(b'-c) \\ \le& \sup_{\xi\le b'-s}\{\beta^u(\xi) - \alpha^l(\xi)\} \end{aligned}$$

Moreover, it trivially holds $0 \le cap(b') \le M$. So

$$cap(b') \le z(b'-s) \tag{15}$$

Combining (14) and (15) gives

$$\begin{aligned} &R(b',b] \le C(b',b] + z(b'-s) \\ \Rightarrow &R(a,b] - R(a,b'] \le +C(b',t] - C(b,t] + z(b'-s) \end{aligned}$$

so (13) holds for this case.

In summary, we have proved (13) for both cases. Now putting (12) and (13) together, we have

$$\begin{aligned} &R'[s,t) \\ <& \sup_{a\le s}\{\inf_{s\le b'\le t}\{R(a,b'] - C(a,s] + C(b',t] + z(b'-s)\}\} \\ =& \sup_{s-a\ge 0}\{\inf_{\substack{b'-s\ge 0 \\ \wedge\, t-b'\ge 0}}\{R(a,b'] - C(a,s] + C(b',t] + z(b'-s)\}\} \\ \le& \sup_{s-a\ge 0}\{\inf_{\substack{b'-s\ge 0 \\ \wedge\, t-b'\ge 0}}\{\alpha^u(b'-a) - \beta^l(s-a) + \beta^u(t-b') + z(b'-s)\}\} \\ =& \sup_{0\le\lambda}\{\inf_{0\le\theta\le\Delta}\{\alpha^u(\lambda+\theta) - \beta^l(\lambda) + \beta^u(\Delta-\theta) + z(\theta)\}\} \end{aligned}$$

So (11) is also proved. □

B. *Proof of Lower Bound Curve* $\alpha_{II}^{\prime l}$

Proof. We will prove that for any time interval $(s,t]$ with $t-s=\Delta$, it holds $R'(s,t] \ge \alpha_{II}^{\prime l}(\Delta)$.

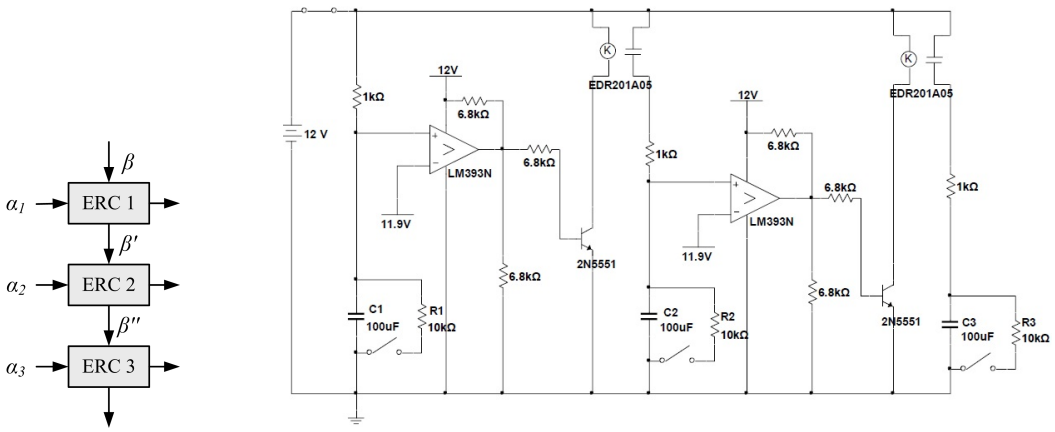
To this end, we first construct an artificial component G^*, whose capacitor is of size 0, to process the input events. The artificial component G^* works in exactly the same way as the GPC. Then we bound $R'(s,t]$ from below by comparing the output event sequences generated by our original component G and by the artificial component G^*.

We use $R'^*(s,t]$ to denote the number of output events in the G^*-sequence. We will prove

$$R'(s,t] \ge R'^*(s,t] - M \tag{16}$$

The proof is established if this is true, since for GPC we have

$$R'^*(s,t] \ge \min\left(\inf_{0\le\theta\le\Delta}\left\{\sup_{0\le\lambda}\{y(\lambda,\theta)\}, \beta^l(\Delta)\right\}\right) \tag{17}$$



(a) System model (b) Circuits

Figure 8. An ERC network with prioritized capacitors and the implementation circuits.

In the following we prove (16). Let u^* be the earliest time point no later than t such that for any time instant in $(u^*, t]$ the buffer is non-empty in the G^*-sequence. We will prove the following two parts, the combination of which gives (16):

$$R'(u^*, t] \geq R'^*(u^*, t] \tag{18}$$

$$R'(s, u^*] \geq R'^*(s, u^*] - M \tag{19}$$

We first prove (18). Let u be the earliest time point no later than t such that for any time instant in $(u, t]$ the buffer is non-empty in the G-sequence. We know $u \geq u^*$ since at any time point the accumulated amount of processed events (from the system starting time) by G is no smaller than that by G^*. The the capacitor size is 0 and the buffer is continuously non-empty in $(u, t]$ in the G^*-sequence, so we have

$$R'^*(u, t] = C(u, t] \tag{20}$$

The buffer is also continuously non-empty in $(u, t]$ in the G-sequence, so we have

$$R'(u, t] = C(u, t] \tag{21}$$

In the G^*-sequence, empty at $buf(u^*) = 0$ and the buffer is continuously busy in $(u^*, u]$, so we know

$$R'^*(u^*, u] \leq R(u^*, u] \tag{22}$$

On the other hand, in the G-sequence, the buffer is empty at u in the, so we

$$R'(u^*, u] \geq R(u^*, u] \tag{23}$$

By (22) and (23) we know $R'^*[u^*, u) \leq R'[u^*, u)$, which, together with (20) and (21), proves (18).

Then we prove (19). Let w^* be the earliest time point no later than s such that for any time instant in $(w^*, s]$ the buffer is non-empty in the G^*-sequencers. Let w be the earliest time point no later than s such that for any time instant in $(w, s]$ the buffer is non-empty in the G-sequence ($w = w^* = s$ if $buf(s) = 0$). We will prove (19) by three parts

$$R'(w^*, u^*] = R'^*(w^*, u^*] \tag{24}$$

$$R'(w, s] \leq R'^*(w, s] \tag{25}$$

$$R'(w^*, w] \geq R'^*(w^*, w] - M \tag{26}$$

We first prove (24). Since the buffer is empty at both w^* and u^* in the G^*-sequence, we know $R'^*(w, u^*] = C(w, u^*]$. At any time point the accumulated amount of processed events (from the system starting time) by G is no smaller than by G^*, so in the G-sequence, the buffer is also empty at w^* and u^*, so we also have $R'(w^*, u^*] = C(w^*, u^*]$. In summary we have (24).

Secondly, we prove (25). Since in the G^*-sequence the buffer is continuously non-empty in $(w, s]$, we have $R'^*(w, s] = C(w, s]$. On the other hand, the buffer is also continuously non-empty in $(w, s]$ in the G-sequence, by Lemma 3 we know $R'(w, s] = C(w, s] - cap(s) - C'(w, s] \leq C(w, s]$. In summary, we have $R'(w, s] \leq R'^*(w, s]$.

Finally, we prove (26). In the G-sequence, by Lemma 3 we know

$$R'(w^*, w] \leq C(w^*, w] + M \tag{27}$$

In the G^*-sequence, the buffer is continuously non-empty in $(w^*, w]$, so we have

$$R'^*(w^*, w] = C(w^*, w] \tag{28}$$

Combining (27) and (27) proves (26). □

APPENDIX II: CIRCUITS FOR PRIORITIZED CAPACITOR

Figure 8-(b) illustrates the circuit to prioritize the capacitors of three nodes connected as in 8-(a). R1, R2 and R3 represents the workload to process input data of three ERC nodes 1, 2 and 3 (a switch for each emulate the decision of whether the data buffer is empty or not). C1, C2 and C3 are the capacitor of three nodes. The dual differential comparator LM393N between node 1 and 2 determines whether C1 is full or not. If yes, the amplifier 2N5551 triggers the relay EDR201A05 and C2 is connected to the charging circuit. The prioritization between C2 and C3 is implemented similarly.