Modeling and Analyzing Method for CPS Software Architecture Energy Consumption

Guangquan Zhang[1,2*],Kan Zhang[1],Xueyang Zhu[3],Mingcai Chen[1],Chengkai Xu[1],Yuzhen Shao[1]

[1] School of Computer Science and Technology, Soochow University, Suzhou, China
[2] State Key Lab. for Novel Software Technology, Nanjing University, Nanjing, China
[3] State Key Lab. of Computer Science,Institute of Software, Chinese Academy of Science, Beijing, China

[*]Corresponding author Email: gqzhang@suda.edu.cn

Abstract—**CPS is a kind of networked embedded system. Its trustworthiness and cost are impacted by energy consumption. So design a low-power, high trustworthiness CPS has been a major challenge. Modeling and analyzing the energy consumption of CPS software architecture at design stage can help to find the energy consumption design defects. These methods can effectively improve the trustworthiness of the CPS software and reduce development costs. Against this problem, first introduce the concept of the energy consumption time Petri nets; then through analyzing the characteristics of the CPS software energy consumption, use energy consumption time Petri nets to construct the CPS software architecture energy consumption model and discuss the energy consumption problems of CPS software such as lowest energy consumption, largest energy consumption and average energy consumption; Finally, indicate the effectiveness of the proposed method by an instance.**

Index Terms—**modeling, analyzing, Software Architecture Energy Consumption, CPS**

I. Introduction

With the advocacy of low-carbon energy, how to develop a low-cost, low-power, and high-performance software system has become a hot research point of embedded systems in recent years. The energy consumption is the original concept of hardware. Tiwari and other people think that analyzing the energy consumption only from the perspective of the hardware structure cannot meet the demand for system-level energy consumption analysis. So the concept of software energy consumption is proposed. Software energy consumption means the energy consumption of the instruction of system components during the execution [1]. The purpose to research the software energy consumption is mapped hardware energy consumption to the software architecture and software characteristics. Then study the impact of software on system energy consumption, and use software design optimized technology to improve system energy consumption efficiency by this purpose. CPS is a kind of networked embedded systems [2], so its software energy consumption issues are critical. CPS's attributes, such as software reliability, security, performance, and so on, are constraint by energy consumption [3], so how to design a low-cost, high -trustworthiness CPS software has become the urgent needs and major challenges for CPS development. Existing energy consumption analyzing and assessment methods usually achieved the running nature of the energy consumption of the system through simulation or test. These methods have some defects such as the high experimental cost and the high re-design cost. These defects of existing energy consumption analyzing and assessment methods are particularly evident when analyzing and evaluating the CPS software.

The software architecture provides the design model and guidance for the structure, behavior and key attributes of the software, which are important factors in software development to meet the user's quality demand. As a software quality attribute, low energy consumption has become the requirement which must be considered by architecture designers. Therefore, in order to improve software development efficiency, reduce costs, and avoid the software re-design caused by energy consumption problem in later development stage, expect accurately understand the runtime software energy consumption nature such as maximum, minimum and average energy consumption at the design stage through the software architecture energy consumption model, which can help to find energy consumption design defects in the design stage and re-adjust software architecture. Petri nets is an effective modeling tool for describing and analyzing the complex systems from the view of the process. It can be widely used in describing and studying the concurrent, asynchronous, and distributed characteristics of the system. And it can provide an operational semantics and the analyzing method combining with qualitative and quantitative. Against these problems, the paper first introduced the concept of the ETPN (energy consumption time Petri nets) [4], which extended time Petri net by energy consumption. It can effectively describe the behavior and the energy consumption of the software; then by analyzing the energy consumption characteristics of CPS software, used energy consumption time Petri nets to construct the CPS software architecture energy consumption model, and discussed the minimum energy consumption, the maximum energy consumption and the average energy consumption of the CPS software based on the energy consumption model; finally, an example illustrates the correctness and validity of the proposed

doi:10.4304/jsw.8.11.2974-2981

method.

II. Related Work

The energy consumption of embedded systems has caused widespread concern of related field researchers. Reference [5] confirmed the main factor of the energy consumption of embedded system is determined by the accesses number of memory in specific embedded systems through modeling the energy consumption of microprocessor-based embedded system. In Non-preemptive real-time systems, the device will take up a lot of transition time when waking up from sleep, which hindered the dynamic energy management of systems. Against this problem, reference [6] proposed a real-time task scheduling algorithm, which can maximize the effectiveness of dynamic energy management of I/O devices. Reference [7] found that increasing the embedded system memory will lead to the increase in system cost, energy consumption. To solve this problem, a software-based embedded system RAM compression algorithm was proposed. This algorithm can significantly improve the validity of memory capacity and maintain the system performance and energy consumption without changing hardware or system design.

The above studies are still stuck in the instruction-level and code-level energy consumption analysis and estimation, while research in the higher abstraction level about energy consumption is relatively few. So energy consumption analysis based on the software architecture has become a hot research topic. Reference [8] can effectively reduce the energy consumption of the system through optimizing process concurrent management process by extending the control / data flow chart, such as the merger process, the transmission of data and mass, calculation migration and choose the method of efficient communication. Reference [9] proposed a method supporting modeling and analyzing the energy consumption and the execution time of embedded system software in the design stage. Reference [10] proposed the assumptions about the nonlinear relationship exists between the software energy consumption and software architecture level features, extracted five software architecture level features , analyzed and measured the impact of these features on software energy consumption. Reference [11] proposed an energy consumption model based on the CSP (communicating sequential process) process algebra language. The model regarded the interface as basic research object, and defined the maximum, minimum and average energy consumption of the system. But the article is lack of verification about the effectiveness of the model.

III. Energy Consumption Time Petri Nets

In traditional Petri nets theory, the concept of time does not explicitly reflected in the prototype of Petri nets, which limits the application of Petri nets in real-time systems. Therefore, introducing the concept of time into Petri nets can effectively improve the ability of Petri nets for analyzing the time behavior and evaluating the performance of real-time systems. And expending the time Petri net by energy consumption, the generated ETPN model can be further analyzed the energy consumption of the system. The simply introduction about the basic concepts, definitions and terminology of ETPN model is as following.

A. Basic Definitions

Definition 1. Time Petri net is a six-tuple $TPN = (P, T, F, M_0; \alpha, \beta)$:

(1) $P = \{p_1, p_2, ..., p_m\}$ is a finite, non-empty set of places;

(2) $T = \{T_1, T_2, ..., T_n\}$ is a finite, non-empty set of transitions;

(3) $F \subseteq ((P \times T) \cup (T \times P)$ are represented as edges;

(4) $M_0 : P \to N$ is initial mark, (P, T, F, M_0) is a Petri net;

(5) $\alpha : T \to Q^+$ is the earliest fire time function;

(6) $\beta : T \to Q^+$ is the latest fire time function.

In order to describe the characteristic of the energy consumption of the system, on the basis of time Petri nets, form the energy consumption time Petri nets by given transition the energy consumption function. The definition is as follows.

Definition 2. Energy consumption time Petri nets is a seven-tuple $ETPN = (P, T, F, M_0; \alpha, \beta, e)$:

(1) $(P, T, F, M_0; \alpha, \beta)$ is a TPN;

(2) $e : T \to Q^+$ is the transition energy consumption function, it represents the energy consumption changes in per unit of time.

Next to discuss ETPN semantic model, and introduce the concept of energy consumption time transition system [3].

Definition 3. The semantic of ETPN is an energy consumption time system, which is a five-tuple, representing as $ETTS_{ETPN} = (S, \Sigma, s_0, \to, \omega)$:

(1) S is a set of all ETPN mark states;

(2) $\Sigma \subseteq (T \times N)$ is a set of (t, σ), and σ represents the execution time of transition t;

(3) $s_0 = m_0$ represents the initial mark state;

(4) $\to \subseteq (S \times \Sigma \times S)$ represents the relationship between transition;

(5) ω represents the energy consumption of fired transition. $\forall t \in T, m \xrightarrow{t,\sigma} m'$, then $\omega = \sigma \times e(t)$.

The behavior of energy time Petri net is defined as an energy transition sequence as the behavior of time Petri net. The energy transition sequence represents as $\delta = (t_1, \sigma_1, \omega_1)(t_2, \sigma_2, \omega_2)...(t_n, \sigma_n, \omega_n)$. $(t_i, \sigma_i, \omega_i)$ represents the execution time of transition t_i is σ_i units time. The energy consumption is $\theta_i = \sigma_i \times e(t_i)$. $e(t_i)$

is the energy consumption in unit time of transition t_i. So $\sum_{j=1}^{n}\sigma_j$ is the total time after the execution of δ, $AE=\sum_{j=1}^{n}\omega_j$ is the total energy consumption after the execution of δ. If there is an energy transition sequence δ ,which can deduce m_i from m_0, then m_i is reachable.

Definition 4. Let two ETPN models represent as *A* and *B*. Suppose P_A and P_B represent the place set of two ETPN models, P_A^I and P_B^I represent the input place sets of two ETPN models, T_A^I and T_B^I represent the input transition sets of two ETPN models, P_A^O and P_B^O represent the output place sets of two ETPN models, T_A^O and T_B^O represent the output transition sets of two ETPN models, P_A^H and P_B^H represent the inner place sets of two ETPN models. When two ETPN models meet following conditions：

(1) $P_A^H \cap P_B = \varnothing$, $P_A^I \cap P_B^I = \varnothing$;

(2) $P_A^O \cap P_B^O = \varnothing$, $P_A \cap P_B^H = \varnothing$;

Then *A* and *B* is composable, the communication set is $Communication(A,B)=(T_A^I \cap T_B^O)\cup(T_A^O \cap T_B^I)$。

The above definition is based on the component thought. Two different components in the system perform different functions. They should have different input and output. The same input should get the same output. Meanwhile, the output of the one component is often the input of another component, or vice versa.

Definition 5. If ETPN model A and B is composable, then the composition model $A\otimes B$ is also an ETPN model：

(1) $P_{A\otimes B}=P_A \cup P_B$;

(2) $T_{A\otimes B}=T_A^H \cup T_B^H \cup Communication(A,B)$;

(3) $F_{A\otimes B}=((P_{A\otimes B}\times T_{A\otimes B})\cup(T_{A\otimes B}\times P_{A\otimes B}))$;

(4) $M_0^{A\otimes B}:P_{A\otimes B}\to N$;

(5) $\alpha_{A\otimes B}:T_{A\otimes B}\to Q^+$;

(6) $\beta_{A\otimes B}:T_{A\otimes B}\to Q^+$;

(7) $e_{A\otimes B}:T_{A\otimes B}\to Q^+$.

B. Reachable State Graph

Reachable state graph is one of important methods to analyze the feature of Petri nets. Firstly introduce the definition of the interval of energy consumption, and then discuss the definition of ETPN state class and the definition of reachable state graph with the extension of the energy consumption.

Definition 6. Let $E=[a,b]$. If $a,b\in Q^+$, then E is an interval of energy consumption. Let EI represent a set of all energy consumption interval. Suppose $E_1,E_2\in EI$, $E_1=[a,b]$, $x\in Q^+$, let $\downarrow E_1=a$, $\uparrow E_1=b$, $E_1+E_2=[\downarrow E_1+\downarrow E_2,\uparrow E_1+\uparrow E_2]$, $E_1+x=[\downarrow E_1+x,\uparrow E_1+x]$, $E_1\times x=[\downarrow E_1\times x,\uparrow E_1\times x]$ be the primary and secondary computing of an energy consumption interval. Then E_1+E_2, E_1+x, $E_1\times x$ is also the energy consumption interval.

Then define the state classes of ETPN. The detail of definition is as follows.

Definition 7. For an ETPN, a state class $C=(M,\theta,E)$ is a three-tuple:

(1) M is marks set;
(2) θ is the total time interval of fired transition;
(3) E is the total energy consumption interval of fired transition;

Reachable state graph is the digraph combined with all state classes in transition fired order. The specific definition is as follows.

Definition 8. Reachable state graph $G=(CS,TS)$ is a digraph. It is obtained from $ETTS_{ETPN}$.

(1) CS is the set of all nodes, each node represents a state class;
(2) $TS:CS\xrightarrow{t}CS$ is the set of all edges, each edge represents the relationship between two states.

After extending the energy consumption for the state class, the status of the reasonableness and completeness should be discussed. Reasonable means any state class calculated by definition 3 always with an energy interval; Completeness means the total energy consumption from the initial state class to any state class certainly falls within the interval of the energy consumption.

Theorem 1. For a state class (M_n,θ_n,E_n) reached from initial state class (M_0,θ_0,E_0), E_n must be an energy consumption interval; and the total energy consumption of any transition fire sequence from (M_0,θ_0,E_0) to (M_n,θ_n,E_n) certainly falls within E_n.

Proof. Firstly, use mathematical induction to prove E must be an energy interval. By the known of condition, there is a transition fire sequence $t_0t_1t_2...t_{n-1}$ from (M_0,θ_0,E_0) to (M_n,θ_n,E_n). If $n=0$, (M_n,θ_n,E_n) is (M_0,θ_0,E_0). Theorem holds. Suppose $n\le k-1$ theorem holds, E_k is an energy consumption interval. If $n=k$, $E_{k+1}=E_k+[\alpha(t_k)\times e(t_k),\beta(t_k)\times e(t_k)]$. By the definition 4, $[\alpha(t_k)\times e(t_k),\beta(t_k)\times e(t_k)]$ is an energy consumption interval. And by the known of definition 4 and E_k, E_{k+1} is also an energy consumption interval.

Similarly, use mathematical induction to prove the total energy consumption of any transition fire sequence from (M_0,θ_0,E_0) to (M_n,θ_n,E_n) certainly fall within E_n. If $n=0$, then total energy consumption $AE_0=0$. Theorem holds. Suppose $n\le k-1$ theorem holds, then the total energy consumption AE_{k-1} of transition fire sequence $t_0t_1t_2...t_{k-1}$ certainly falls within E_k. If $n=k$, then the fire energy consumption C_k of transition t_k certainly falls within $[\alpha(t_k)\times e(t_k),\beta(t_k)\times e(t_k)]$. So $AE_{k-1}+C_k$ certainly falls within $E_k+[\alpha(t_k)\times e(t_k),\beta(t_k)\times e(t_k)]$. The total energy consumption AE_k of transition fire sequence $t_0t_1t_2...t_k$ certainly falls within E_{k+1}, while $E_{k+1}=E_k+[\alpha(t_k)\times e(t_k),\beta(t_k)\times e(t_k)]$.

□

IV. MODELING AND ANALYSIS OF CPS SOFTWARE ARCHITECTURE ENERGY CONSUMPTION

CPS is the system closely integrated computing system with the physical environment. So the energy consumption of computing and physical processes must be taken into account by constructing and analyzing the energy consumption model. Firstly, analyze the features of CPS software energy consumption.

A. Features of CPS Software Energy Consumption

There are three modules in any CPS: computing module, communication module and the physical module. The computing module is responsible for the implementation of the computing work of CPS, including data processing, control decisions; the communication module is responsible for information exchange between the various modules; the physical module is responsible for the perception and control of the physical environment. Figure 1 is an abstract structure of CPS energy consumption, which can define the energy consumption of the CPS software p^{cps}.

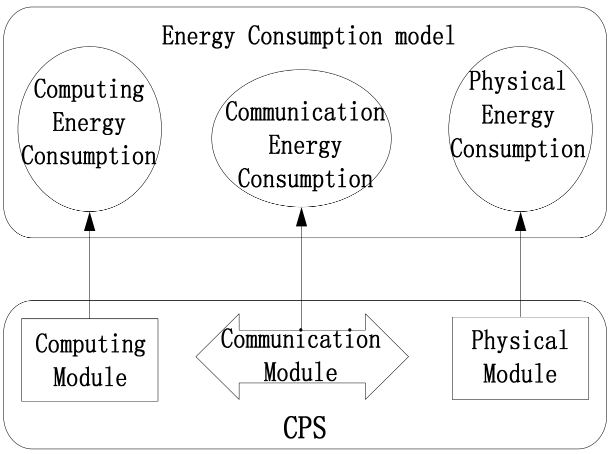

Figure 1. Abstract Structure of CPS Energy Consumption

Definition 9. The CPS software energy consumption depends on the computing and non-computing energy consumption. The energy consumed in the computing process is called computing energy consumption, including computing module and communication module energy consumption. Energy consumption in the non-computing process is called non-computing energy consumption, mainly refers to the physical module's energy consumption. The p^{cps} can be represented as follows:

$$p^{cps} = computing\_energy + non-computing\_energy.$$

B. Modeling CPS Software Architecture Energy Consumption

After analyzing the energy consumption features of the CPS software, use the energy consumption time Petri nets to construct CPS software architecture energy consumption model according to definition 9. CPS contains computing module, communication module and the physical module, so the CPS software ETPN model is defined as follows.

Definition 10. Energy consumption time Petri of CPS is $ETPN^{CPS}=(CP\_ETPN, CM\_ETPN, NC\_ETPN)$. C_ETPN, CM_ETPN and NC_ETPN are the ETPN models of computing module, communication module and the physical module.

According to definition 8, construct the CPS software ETPN model. The construction process is as follows.

(1) Firstly, analyze the parts of the system. CPS is composed by independent embedded devices. Each device has a computing, communications or control module. Each module is responsible for a certain function, and has a certain degree of autonomy, but it depends on the computing results of the other modules of the system. The structure model of CPS is shown in Figure 2. There are numerous devices in CPS. Each device is composed by the computing module, communication module, or (and) the control module. Effective and reliable communication between devices through the communication module depends on the network.

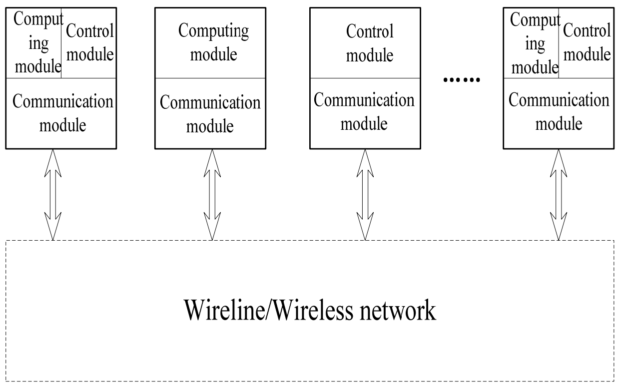

Figure 2. The structure model of CPS

(2) Abstract each module into an ETPN model. Add the input or output places or transition for each module needed communication as the communication interface with other modules. By definition 4, determine the composable ETPN model. In practical systems, sending and receiving information will consume energy. Therefore, add energy consumption function to the input/output transition.

(3) After established ETPN model of each module, connect each module by the communication interface according to the communicating order. As

the information transmission process also consuming energy (such as the energy consumption of the network device), add energy consumption function to transition connecting two ETPN model, which represents the transmission energy consumption.

C. Analyzing CPS Software Architecture Energy Consumption

After constructing the CPS software architecture energy consumption model, reachable state graph of the model can be constructed according to definition 6. Then CPS software architecture energy consumption problems can be analyzed by reachable state graph. The CPS software architecture energy consumption problems include the software maximum energy consumption, minimum energy consumption and the average energy consumption analysis. Here can be converted to searching the energy consumption path from initial state to the target state in reachable state graph. The analysis of these questions is as follows. Because of the real-time of CPS software, the desired behavior sequence within a certain time period is generally limited. So we will not consider the energy consumption problems in the case of infinite behavior sequence.

- **Minimum Energy Consumption Path Analysis**

Theorem 2. In reachable state graph, the minimum energy path reaching any node must be a simple path.

Proof. Use proof by contradiction to prove. Let the minimum energy path reaching a node in reachable state graph is a simple path, which is represented as $\varepsilon = C_0 \xrightarrow{t_0} C_1 \xrightarrow{t_1} \ldots \xrightarrow{t_{j-1}} C_j \xrightarrow{t_j} \ldots \xrightarrow{t_{k-1}} C_k \xrightarrow{t_k} \ldots \xrightarrow{t_{m-1}} C_m$. The energy consumption of the path is AE. There are at least a sub-path contains the ring in this path. Let the ring as $\varepsilon_1 = C_j \xrightarrow{t_j} \ldots \xrightarrow{t_{k-1}} C_k$. And then let the rest of the sub-path as $\varepsilon_2 = C_0 \xrightarrow{t_0} C_1 \xrightarrow{t_1} \ldots \xrightarrow{t_{j-1}} C_j$, $\varepsilon_3 = C_k \xrightarrow{t_k} \ldots \xrightarrow{t_{m-1}} C_m$. Let $AE = AE_1 + AE_2 + AE_3$, $AE' = AE_2 + AE_3$. Obviously, $AE > AE'$. Therefore, after replaced the ring ε_1 by the state C_j, the energy consumption of the path is less than the original path. This contradict the assumption.

□

Then the algorithm to search reachable state graph for the minimum energy consumption path reaching a node can be constructed by theorem 2. Since there may be different paths with the same energy consumption, the minimum energy path may be more than ones. Because this article only focus on the energy consumption of path, and are not concerned about the path itself. So let one of paths as search result. Since each state has an energy interval, let the minimum value of the energy consumption interval as the minimum energy consumption reaching the state.

Algorithm 1. Minimum Energy Consumption Path

Input: reachable state graph G, target node C_m;

Output: the minimum energy consumption path min_path, the minimum energy consumption min_AE;

```
min_AE = ∞ , AE = 0 ; // AE is the energy consumption  of current path
min_path = ϕ;
current_path = <C0>;
do
    node = the last node of current_path;
    if there is no new node, then delete the last node of
     current_path;
    else
        node = new node of current node;
        if node == Cm then
            compute the AE of current_path;
            if AE < min_AE
            then
                min_AE = AE;
                min_path = ϕ;
                min_path = current_path ∪ Cm ;
        else current_path = current_path ∪ node;
while current path = < >
return min_path;
```

- **Maximum Energy Consumption Path Analysis**

Theorem 3. If a period of time to reach any node was given, then there is the maximum energy consumption path of all path to reach the node.

Proof. Without loss of generality, let the reach node be C_m. Because each transition consumes time, the length of any path to reach C_m is certainly limited after given the time deadline to reach C_m. If not, the time will be unlimited. So the energy consumption and the number of path is limited. So after determining all paths and energy consumption to reach C_m in a given time period, it can certainly find the energy consumption of a path is greater than or equal to the energy consumption of the other paths. That means the maximum energy consumption path to reach C_m exists.

□

Then the algorithm to search reachable state graph for the maximum energy consumption path reaching a node can be constructed by theorem 2. Since there may be different paths with the same energy consumption, let one of paths as search result. Since each state has an energy interval, let the maximum value of the energy consumption interval as the maximum energy consumption reaching the state.

Algorithm 2. Maximum Energy Consumption Path

Input: reachable state graph G, target node C_m, time deadline T_D;

Output: the maximum energy consumption path max_path, the maximum energy consumption max_AE;

```
max_AE = ∞ , AE = 0 ; // AE is the energy consumption of current path
T = 0; //T is the time consumption of current path
max_path = ϕ;
```

current_path = $<C_0>$;
do
    node = the last node of current_path;
    if there is no new node, then delete the last node of current_path;
    else
        node = new node of current node;
        if T= T + t(node) <= T_D then
            if node ==C_m then
                compute AE of current_path;
                if AE > max_AE then
                    max_AE = AE;
                    max_path = ϕ;
                    max_path = current_path $\cup$ C_m;
            else current_path = current_path $\cup$ node;
while current path = < >
return max_path;

Lemma 1. For a bounded ETPN, algorithm 1 and 2 will be terminated.

Proof. For bounded ETPN, the number of marks and state classes is limited. So the mark set M and the implementation time domain D of bounded ETPN are bounded. Therefore, its energy consumption interval is bounded. Obviously, the number of $C=(M,\theta,E)$ is limited. That is the number of node in reachable state graph is limited. The complexity of algorithms 1 and 2 is proportional to the number of nodes in the reachable state graph. Therefore, the algorithm 1 and 2 for bounded ETPN must be terminated.

□

- **Average Energy Consumption Analysis**

In some cases, the average energy consumption of the system need to be understood, which can help evaluate the sustainability, reliability, execution costs, and other attributes under known energy conditions. In reachable state graph, the average energy consumption analysis method is first to identify the energy consumption of all paths to the target state. Then compute the arithmetic mean of the energy consumption of these paths according to given execution probability of each path. The specific algorithm is as follows.

Algorithm 3. Compute Average Energy Consumption
Input: eachable state graph G, system path execution probability array F[n];
Output: Average Energy Consumption AveE;
$AE=0$, AveE = 0; // AE is the energy consumption of current path
path_num = 0; //path_num is the number of path
current_path = $<C_0>$;
do
    node = the last node of current_path;
    if there is no new node, then
        compute AE of current_path;
        AveE = AveE + AE * F[path_num];
        path_num++;
        delete the last node of current_path;
    else
        node = new node of current node;
while current path = < >
return AveE;

V. Case Study

TCAS (Traffic Alert and Collision Avoidance System) is information system not depending on any land-based air traffic control. It is the typical applications of CPS in the field of aviation. It can help the pilot quickly identify the relative position between the aircraft.

The TCAS generally consists of three modules: distance detection module, collision management module and flight control module. Distance detection module is mainly responsible for detecting the distance with other aircraft. And send distance information to the collision management module. Collision management module judges whether the distance safety after received the distance information. And send warning or safety information to the flight control module. The flight control module judges after the receipt of the information, and send avoidance or recovery action to the collision management module. At last, the collision management module is responsible for the implementation of avoidance or recovery action.

Model the TCAS by the above proposed modeling approach. The concrete steps are as follows:

(1) Analyze TCAS, including the distance detection module, collision management module and flight control module;
(2) Build ETPN model for each module, and identify the communication interface of each module;
(3) According to the order of communication between modules, connect the respective interface.

According to the above described modeling steps, obtain the ETPN model of TCAS shown in Figure 3. The name, execution time and the energy consumption of each transition is shown in Table 1.

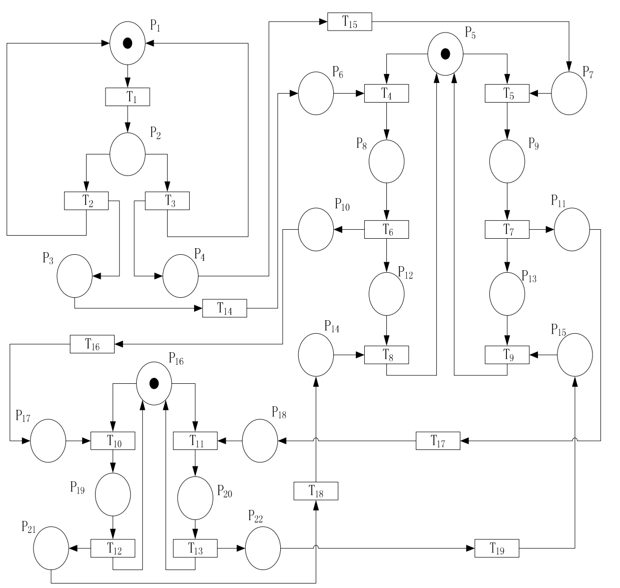

Figure 3. the ETPN model of TCAS

TABLE 1.
THE INFORMATION OF EACH TRANSITION

Transition	Name	Execution time	Energy consumption
T_1	Detecte distance	[3,8]	20
T_2	send dangerous distance	[1,3]	8
T_3	send safety distance	[1,2]	7
T_4	receive dangerous distance	[2,3]	9
T_5	receive safety distance	[1,2]	8
T_6	send warming information	[2,4]	8
T_7	send safety information	[1,3]	7
T_8	avoid	[3,7]	21
T_9	recover	[3,6]	17
T_{10}	receive warming information	[2,5]	8
T_{11}	receive safety information	[2,4]	7
T_{12}	Send avoiddance	[1,4]	9
T_{13}	Send recovery	[1,3]	7
T_{14} ,T_{15},T_{16}, T_{17},T_{18},T_{19}	communicaiton	[1,2]	10

According to ETPN model of Figure 3, construct the reachable state graph of TCAS, which is shown in Figure 4. Due to space constraints, and the mark set is not the focus of this article. Therefore, the specific content of each mark set in the reachable state graph is not given.

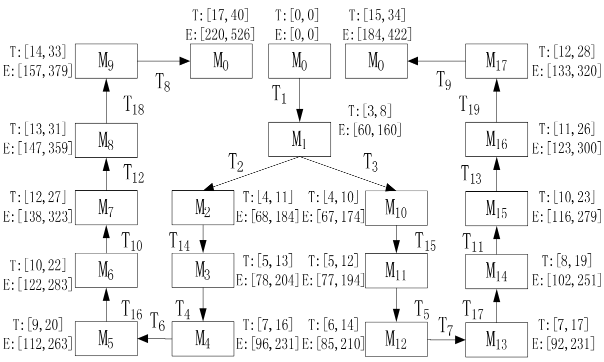

Figure 4. reachable state graph

After constructed Figure 4, the relevant energy consumption problems of TCAS can be analyzed by using the above proposed analytical method. After applying the algorithm 1 and algorithm to Figure 4, the minimum energy consumption path is <M_0, M_1, M_{10}, M_{11}, M_{12}, M_{13}, M_{14}, M_{15}, M_{16}, M_{17}, M_0,>. The minimum energy consumption is 184. The minimum energy consumption path is <M_0, M_1, M_2, M_3, M_4, M_5, M_6, M_7, M_8, M_9, M_0,>. The maximum energy consumption is 526. Assumed that the execution probability of T2 and T3 were 0.3 and 0.7, and then the average energy consumption of the system is 323.6. In addition to the minimum energy consumption, largest energy consumption and average energy consumption, sometimes need analyze the energy consumption of specific certain period process according to requirements. Then the energy consumption of the process can be analyzed as long as finding the corresponding part of the reachable state graph of the process.

VI. CONCLUSIONS

In the aspect of software trustworthiness analysis and evaluation research, some related methods have been proposed [12-15]. As a networked embedded system, the trustworthiness, cost and other attributes of CPS are constrained by energy consumption. How to find design defects of the energy consumption of CPS software in the design phase, effectively reduce development costs, and improve system trustworthiness, has become requirements and challenges for CPS development. To solve this problem, we firstly introduced the concept of energy consumption time Petri nets. Then used energy consumption time Petri nets to construct the CPS software architecture energy consumption model through analyzing the energy consumption features of the CPS software, and proposed analysis methods for CPS software minimum energy consumption, maximum energy consumption and the average energy consumption.

The next work will consider how to use the analysis results such as minimum energy consumption, maximum energy consumption and average energy consumption to evaluate the trustworthiness of the CPS software energy consumption. It can help understand how energy consumption impact on the trustworthiness of CPS software.

ACKNOWLEDGMENT

This research was sponsored by the Natural Science Foundation of Jiangsu Province (BK2011281), Applied Basic Research Program of Suzhou (SYG201241), Scientic Research Innovation of College Graduated in Jangsu Province under Grant (CXLX12-0809, CXLX13-820) and College Students' Practice and Innovation Training Project of Soochow University (2012yb010).

REFERENCES

[1] Tiwari V, Malik S, Wolfe A. Power analysis of embedded software: A first step towards software power minimization. *IEEE Transactions on Very Large Scale*

Integration, 2(4), pp. 437-444, 1994.

[2] Poovendran R. Cyber-Physical Systems:Close Encounters Between Two Parallel Worlds. *Proceedings of the IEEE*, 98(8), pp. 1362-1366, 2010.

[3] Sandeep K.S. G, Tridib M, Georgios V, et al. Research directions in energy-sustainable cyber–physical systems. *Sustainable Computing: Informatics and Systems*, 1(1), pp. 57-74, 2011.

[4] Tavares E., Maciel P., Silva B., et al. Hard realtime tasks' scheduling considering voltage scaling, precedence and exclusion relations. *Information Processing Letters*, 108(2), pp. 50-59, 2008.

[5] Konstantakos V, Chatzigeorgiou A, Nikolaidis S, et al. Energy consumption estimation in embedded systems. *IEEE Transactions on Instrumentation and Measurement*, 57(4), pp. 797-804, 2008.

[6] Euiseong S, Sangwon K, Seonyeong P, et al. Dynamic alteration schemes of real-time schedules for I/O device energy efficiency. *ACM Transactions on Embedded Computing Systems*, 10(2), pp. 23, 2010.

[7] Lei Y, Robert P. Dick, Haris Lekatsas, et al. Online memory compression for embedded systems. *ACM Transactions on Embedded Computing Systems*, 9(3), pp. 27, 2010.

[8] Fei Y, Ravi S, Raghunathan A, et a1. Energy-optimizing source code transformations for operating system- driven embedded software. *ACM Trans on Embedded Computing Systems*, 7(1), pp. 1-26, 2007.

[9] Gustavo C, Paulo M, Eduardo T, et al. Energy consumption and execution time estimation of embedded system applications. *Microprocessors and Microsystems*, 35(4), pp. 426-440, 2011.

[10] Liu X B, Guo B, Shen Y, et al. Embedded Software Energy Modeling Method at Architecture Level. *Journal of Software*, 23(2), pp. 230−239, 2012.

[11] Zhang T T, Wu X, Li C D, et al. On Energy-Consumption Analysis and Evaluation for Component-Based Embedded System with CSP. *Chinese Journal of Computers*, 32(9), pp. 1876-1883, 2009.

[12] Zhang N, Cui G, Liu H W. Software Reliability Analysis using Queuing-based Model with Testing Effort. *Journal of Software*, 8(6), pp. 1301-1307, 2013.

[13] Guo Y, Ma P J, Su X H. Dynamic Collection of Reliability-Related Data and Reliability Evaluation for Internet software. *Journal of Software*, 8(6), pp. 1390-1397, 2013.

[14] Dong J L, Shi N G, Wang J Z. Research and Application on Multi-Layer Matter-Element Extension Synthesis Evaluation Method for Software Quality. *Journal of Software*, 6(9), pp. 1866-1872, 2011

[15] Shi L, Yang S L, Li K, B G. Developing an Evaluation Approach for Software Trustworthiness Using Combination Weights and TOPSIS. *Journal of Software,* 7(3), pp. 532-543, 2012

Guangquan Zhang received the MS and PhD degrees in computer science from Chongqing University, in 1988 and 1999, respectively. He is currently a professor in the School of Computer Science and Technology, Soochow University, China. , and is the senior member of CCF. His research interests include networking software engineering, formal methods, cloud computing and Cyber Physical Systems, et.al.

Kan Zhang was born in Jiangsu, China. He received his Bachelor degree in Software Engineering in 2010 from School of Computer science and Technology, Soochow University, China.Now he is a graduate student at Soochow University, China. His main research interests include Embedded system, CPS and software trustworthiness evaluation.