Research on Real-Time System and Related Graph Task Model

Juan Xiao, Song Wang, Sheng Duan and Shanglin Li*
XiangNan University, Chenzhou, Hunan, China

*Corresponding author e-mail: lsldd@xueshumail.cn

Abstract. Generally speaking, real-time system is considered to be able to influence the environment by receiving and processing data, and returning calculation results rapid enough, so as to control the environment. In computer science, real-time system describes the software and hardware system affected by time constraints, and its correctness relies on the logical correctness of the function and the time when the result is generated. According to the main characteristics of real-time operating system, such as time constraint, predictability and reliability, it puts forward higher requirements for the time accuracy and reliability of real-time operating system. This paper first introduces the real-time system from its main characteristics, related concepts and scheduling algorithm. Then five classical graph based task models of real-time system are introduced. Finally, this paper introduces the directed graph real-time task model from two aspects of definition and semantics. As an extension of real-time system task model, directed graph real-time task model is considered to be able to provide real-time systems with stronger expressive power and support the formal study of time constraint problems.

Keywords: Real-Time Systems, Graph Task Model, DRT (Digraph Real Time) Tasks Model, Schedulability

1. Introduction
Recently, real-time system has been widely used in automation, automobile, ship, aircraft, industrial control, home appliance control and other important fields. On the one hand, the degree of automation, electrification, digitization and intelligence of modern control system is constantly improving; on the other hand, the requirements of system safety, stability, comfort and other functions and performance are becoming more and more strict. These changes require the real-time system to provide more powerful support and control capabilities. Therefore, the demand for predictability and application performance of real-time system is higher and higher.

Generally speaking, a real-time system contains a certain number of parallel tasks and shares limited hardware resources. How to predict that each task can be called before its deadline is the most important schedulability problem of real-time system. al-time system are more and more demanding. To analyze and predict the schedulability of real-time system, the corresponding real-time task model must be constructed abstractly according to the actual demand of real-time system. Based on these
abstract task models, we can do formal research and time constraint analysis on real-time system, and do schedulability analysis on real-time system.

2. Real-time Systems
Generally speaking, real-time system is considered to be able to receive data, process data, and fast enough. Return the calculation results to affect the environment, so as to control the environment [1]. In computer science, the most critical factor of real-time system is time constraint. Therefore, it is generally considered to describe the software and hardware system affected by time constraint. Its correctness depends on two aspects: the logical correctness of the function and the time of the result generation. If the time constraint is not fully satisfied, serious errors will occur in the real-time system [2]. Therefore, the main goal of real-time system research is to predict the behavior of the system timely and accurately, so as to ensure the safety and reliability of the system.

However, with the continuous improvement of the electronic and automation of the control system, the function and performance requirements of the real-time system are also higher and higher. At the same time, the type, number and load of tasks in the system also increase sharply, which brings great challenges to the time verification of real-time system.

Since 1980, real-time system has been studied systematically. In 2014, butazzo made a topic report on computer system at RTSS conference, distinguishing three kinds of computing systems: real-time system, embedded system and information physical system [3]. At the same time, he also thinks that the three are not simply inclusive, but not subordinate to each other, and there is a huge cross field between any two. Because real-time system is widely used in embedded system, it is sometimes called embedded real-time system [4].

2.1 Main Features of Real-Time System
As a real-time system of computer system, it has all the essential common properties of software and hardware of computer system. Due to different uses, real-time system has its own characteristics compared with general computer system.

(1) Time constraint
The most significant feature of real-time system is the high requirement of system time. Therefore, time constraint is an important index of the hardware and software design of real-time system.

(2) Predictability
If we want to meet the time constraints of real-time system, we need to be capable to analyze and predict the execution behavior of all tasks, and we need to ensure that the time constraints of all tasks are met. Otherwise, the real-time system needs to be redesigned to meet the time constraints. The realization of real-time system predictability is related to the hardware architecture, kernel scheduling mechanism and programming language.

(3) Reliability
Because the current application of real-time system is mostly concentrated in the important areas related to the safety of people's lives and property, any small error may bring huge losses, so the design of the system should have the ability to complete the specified function without failure in the specified time and environment. Therefore, reliability is one of the features of real-time system.

(4) High efficiency
Because most of the applications of real-time system are running on some small embedded devices, there are limitations in memory, computing, energy supply and other resources. These limitations will cause the real-time system to work under high load, and must be able to respond within the specified time limit. This requires the real-time system to make efficient use of available resources, so as to achieve the desired performance.

(5) Complexity
The complexity of real-time system is mainly reflected in environment complexity and constraint complexity. The complexity of environment is that embedded real-time system has a wide range of applications and needs to interact with the external environment frequently to respond to various
environmental changes in time. Constraint complexity is mainly reflected in scheduling real-time tasks, such as time constraints, semantic constraints, resource constraints and performance constraints.

2.2 Related Concepts of Real Time System

Three aspects of the real-time system will be introduced: the basic concept of real-time system, related constraints and scheduling algorithm.

(1) The basic concept of real-time system

For real-time systems, there are usually multiple tasks coexisting, and each task has a variety of different states:
- **Activate**: indicates that the task can be executed;
- **Ready**: indicates that the task is ready and enters the preparation queue to be executed;
- **Execute**: indicates that the task is being executed;
- **suspend**: indicates that the task is suspended due to the requirements of other higher-level tasks during execution;
- **Blocking**: indicates that the task cannot be executed smoothly due to resource constraints during execution;
- **Complete**: indicates that the task has been completed.

(2) Related constraints of real-time system

Because of the characteristics of real-time system, we know that there are many constraints in the system:
- **Time constraint**: the most important constraint of real-time system is time constraint, which is generally determined by release time (the time when the task is released or ready to be executed), execution time, worst-case response time, absolute time limit (the task must be completed before the absolute time), and relative time. It is constrained by the concepts of time limit (the difference between absolute time limit and release time), completion time (the time of task completion) and response time (the time from release to completion). Therefore, for a specific task, we can judge whether the time constraint can meet the system requirements by verifying whether the response time is not greater than the relative time limit.
- **Semantic constraints**: In some real-time systems, due to the specific relationship between tasks, there is a certain execution order between tasks, which is the semantic constraint of real-time systems. For this kind of semantic constraint, we use directed graph to describe it. In the simple directed graph shown in Figure 1, Suppose there are four tasks. From the figure, we can see that the execution of tasks follows a certain order, which is represented by directional arrows. For example, task 2 and task 3 can be executed after task 1, task 3 can also be executed after task 2, and task 4 can also be executed after task 2 and task 3.

![Fig.1 Examples of Simple Digraphs](image)

- **Resource constraints**: Multiple real-time tasks may need to use system resources together, such as computing resources, storage resources and communication resources. However, most of the system resources can only be occupied by a single task at one time, which leads to resource constraints. When there are resource constraints, we need to avoid deadlock according to the agreed resource access protocol, reduce the waiting time of blocking, so as to improve the performance of the system.
Performance constraints: performance constraints refer to meeting such requirements as time constraints, predictability, reliability, and efficiency. The performance indexes of the system are as follows.

3.2 Scheduling algorithm of real-time system
In real-time system, the main task of scheduling algorithm is to allocate system resources, and in the process of completing each task scheduling, and to ensure that the implementation of the task is completed within the time limit of the system. Generally speaking, a good scheduling algorithm should be able to keep all computing resources working at full load, and allow multiple tasks to effectively share other resources of the system to meet as many constraints as possible. However, in the actual real-time system application process, these requirements for scheduling algorithm can not be met at the same time, so we can only use appropriate compromise processing.

Generally speaking, scheduling algorithms can be divided into different types according to different methods, including non preemptive and preemptive; static priority and dynamic priority; offline and online; optimal and heuristic, etc.

3. Typical Graph Based Real-Time System Task Model
Schedulability of real-time system means that each task in real-time system can be completed before its own time limit, that is, each task meets the time constraint of the system. When we design and analyze the real-time system, we must determine whether the system is schedulable. If it is not schedulable, it is necessary to increase resources or reduce the load of tasks to make the system schedulable. In order to analyze and test the real-time system schedulably, it is necessary to abstract a real-time system and construct its corresponding real-time task model. At present, many graph based real-time system task models have been proposed [5]. Next, we introduce five typical graph based implementation system models according to the time sequence of the proposed models.

3.1 Ordic Task Model
The first and simplest periodic task model is proposed by Liu and Layland [6]. The biggest feature of this model is that the cycle of system publishing tasks is fixed, and the WCET (worst case execution time) of each task is the same. The periodic task $o_i$ of this model can be represented by triples $(α_i, β_i, γ_i)$, and its utilization ratio $Di=α_i/β_i$. The periodic task can be represented by a node with a spin edge. As shown in Figure 2, the release of periodic task must be strictly in accordance with the interval time of $T_i$.

![Fig.2 Release of Periodic Task Model](image)

3.2 Multi-frame Task Model
The first model to extend the periodic model is the Multi-frame task model [7]. In the model, WCET is assumed to be not fixed but defined according to periodic behavior, so it has chain release structure. Multi frame task $o_i$ can perform time vector through minimum release interval $T_i$, $→ C_i = (C_{i,1}, ..., C_{i,m})$ and relative time limit $D_i = T_i$ are described.

The multi frame task can be represented by a graph with chain structure, in which the solid node represents the entry of task execution, As shown in Figure 3. In addition, the execution time of multi frame task is variable, but the release mode and relative time limit are fixed.

![Fig.3 Graph representation of multi frame tasks](image)
3.3 Generalized Multi-frame Task Model
After generalizing the multi frame task model, the generalized multi frame task model (GMF) is proposed. The generalized multi-frame task $o_3$ can be described based on three vectors: minimum release interval time vector $\rightarrow T_i = (T_{i_1}, \ldots, T_{i_{m_i}})$, execution time vector $\rightarrow C_i = (C_{i_1}, \ldots, C_{i_{m_i}})$ and relative time limit vector $\rightarrow D_i = (D_{i_1}, \ldots, D_{i_{m_i}})$, so it has a greater degree of freedom. Similar to multi frame task, generalized multi frame task can also be represented by graph with chain structure, As shown in Figure 3. but different from multi frame task, generalized multi frame task allows flexible configuration of period and time limit [8].

3.4 Non-Cyclic Generalized Multiframe Task Model
The non cyclic generalized multi frame task model (Non-Cyclic GMF) is obtained by removing the periodic restriction of task release [9]. Specifically, as long as the relative release time limit is met, new tasks can be released. As shown in Fig. 4, the acyclic generalized multi frame task can be represented by Fig. 3, which is similar to removing spin edges. Therefore, its task release does not require periodicity.

![Fig.4 Graph representation of acyclic generalized multi frame tasks](image)

3.5 Recurring Branching Task Models
Baruah [10] proposed the recurring branch task model (RB), which is less restrictive than the generalized multi framework task model. It allows selecting different nodes to release tasks, and it can simulate branch release behavior through directed tree. The two extended models of this model are: Recurring Real-Time Task Model (RRT) [11] and Non-Cyclic Recurring Real Time Task Model (Non-Cyclic RRT) [12]. The former is an arbitrary directed acyclic graph, and the latter does not need global periodicity.

4. Digraph Real-Time Task Model
The digraph real-time (DRT) model proposed by stigge et al. Is more generalized than the previous model [13]. DRT can model any real-time task which can be described by directed graph, and has powerful description ability.

4.1 DRT Model Definition
The directed graph model of real-time task describes that N independent real-time tasks $\tau = \{T_1, T_2, \ldots, T_N\}$. Each task T can be described as a directed graph G (T), which consists of two sets: one is the set V (T) of all vertices, and the other is the set E (T) of all directed edges. Vertex set V (T) = {V1, V2, \ldots, VN} indicates all possible job types released by task T. Each vertex V is marked with binary (E (V), D (V)). Where E (V) $\subseteq$ N is the worst execution time (WCET) of the job corresponding to the vertex, and D (V) $\subseteq$ N is the relative deadline of the job. Suppose that E(V) $\leq$ D (V) satisfies all job types corresponding to vertex V. The edges of a digraph G (T) represent the release order of different types of jobs released by task T. Each edge (U, V) $\subseteq$ E (T) is marked with P (U, V) $\subseteq$ N. Where P (U, V) represents the minimum release time interval between the subsequent jobs of type V and the previous jobs of type U.

4.2 DRT Model Semantics
The behavior of each real-time task t can correspond to the possible infinite path traversing the directed graph G (T). On this path, a run-time release of this type of job will be triggered every time a vertex is passed (following the parameters marked on the vertex). Through the path, the release
interval between each two consecutive release jobs obeys the constraint of the corresponding edge. A triple \((R, E, D)\) is used to formally describe a runtime job. The release time of the job is represented by \(R\), the worst execution time by \(E\), and the absolute deadline by \(D\).

The language of DRT task system is defined as the job sequence \(\sigma = [(R_{1}, E_{1}, D_{1}), (R_{2}, E_{2}, D_{2}), \ldots]\). All jobs are arranged in monotonic increasing order according to their release time, that is, for any positive integer \(I \leq J\), \(R_{I} \leq R_{J}\) is satisfied. A job sequence \(\sigma = [(R_{1}, E_{1}, D_{1}), (R_{2}, E_{2}, D_{2}), \ldots]\) is generated by real-time task \(T\) if and only if:

There is a path \(\pi = (V_{1}, V_{2}, \ldots)\) through \(G(T)\). For any \(I > 0\), the following conditions are satisfied simultaneously:

1. \(R_{i+1} - R_{i} \geq P(v_{i}, v_{i+1})\),
2. \(E_{i} \leq E(v_{i})\),
3. \(D_{i} = R_{i} + D(v_{i})\)

A job sequence \(\sigma\) is composed of job sequences generated by all tasks in the real-time task set \(\tau\), then the job sequence \(\sigma\) is generated by \(\tau\).

### 5. Conclusion

The directed graph real-time task model has good expressive ability and is a general extension of many task models. Schedulability analysis of directed graph real-time task model has been widely studied, mainly focusing on the frame separation characteristic or l-mad characteristic. The worst-case response time of directed graph real-time task with arbitrary time characteristic is worth further exploring.

### Acknowledgments

Scientific research project of Hunan Education Department (NO: 17C1487, 17A201).

### References

[1] Greenberger M. Programming Real-Time Computer Systems. 1965.
[2] Buttazzo G C. Hard real-time computing systems: predictable scheduling algorithms and applications. Springer, 2011.
[3] Buttazzo G. Real-Time systems: achievements and perspective. In Keynote speech at Real-Time Systems Symposium. 2014.
[4] Han Gang. Research on communication mechanism and optimization technology of embedded real time system. National defense science and Technology University, 2013.
[5] Stigge M, Yi W. Graph-based models for real-time workload: a survey. Real-Time Systems. 2015, 51 (5): 602–636.
[6] Liu C L, Layland J W. Scheduling algorithms for multiprogramming in a hard real-time environment. Journal of the ACM (JACM). 1973, 20 (1)
[7] Mok A K, Chen D. A multiframe model for real-time tasks. IEEE Transactions on Software Engineering. 1997, 23 (10): 635–645.
[8] Baruah S, Chen D, Gorinsky S, et al. Generalized multiframe tasks. Real Time Systems. 1999, 17 (1): 5–22
[9] Moyo N T, Nicollet E, Lafaye F, et al. On schedulability analysis of non-cyclic generalized multiframe tasks. In 22nd Euromicro Conference on Real-Time Systems (ECRTS). 2010: 271–278.
[10] Baruah S K. Feasibility analysis of recurring branching tasks. In Proceedings of the 10th Euromicro Workshop on Real-Time Systems. 1998: 138–145.
[11] Baruah S K. Dynamic-and static-priority scheduling of recurring real-time tasks. Real-Time Systems. 2003, 24 (1): 93–128.
[12] Baruah S. The non-cyclic recurring real-time task model. In 31st IEEE Real-Time Systems Symposium. 2010: 173–182.
[13] Stigge M, Ekberg P, Guan N, et al. The digraph real-time task model, Proceedings of the 17th
IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS). IEEE, 2011, 71–80.