Research on Hybrid Real-Time System Scheduling Algorithm for Multiprocessor Environment

Pu Cuo Cai Ren*, JianChao Hong
School of mathematics and computer science, Northwest University for Nationalities 730030 Lanzhou Gansu, China

*Corresponding author e-mail: 1224717240@qq.com

Abstract: Firstly, the mixed key system task model of compatible with periodic tasks and sporadic tasks is studied. Based on the model compatible with these two tasks, a semi-divided real-time task scheduling mechanism is proposed. This process also considers the aggressive handling of Low critical task to improve resource utilization. Finally, the relevant program code is designed and developed, and the performance of the proposed algorithm is evaluated by means of simulation and simulation. The correctness of the model and theory is verified, and the theory and protocol design are further improved.

1. Introduction

With the development of modern embedded systems, the sharing of an embedded platform between high safety standards and other non-safety-critical functions has become a necessity. This can solve the conflict between the demand caused by the diversification and complication of software functions and the limited requirements of the hardware platform. This kind of hybrid real-time system is called mixed criticality system (MCS), and its scheduling needs to meet the following objectives at the same time.

① Under the pessimistic time constraints, the verification requirements of the certification standards for safety-critical functions are met.
② Meet the design requirements for efficient use of computing resources under more optimistic time constraints.

The scheduling requirements and reliability authentication requirements of the MCS system are much more complicated than the traditional real-time systems, which makes the classic real-time scheduling algorithm not used efficiently by the MCS system. According to the characteristics of the MCS system, the researchers proposed a series of scheduling strategies, including fixed priority based scheduling methods [1-8] and dynamic priority scheduling methods. These methods can improve the schedulability of the MCS system and ensure the security certification requirements and scheduling requirements of the system. But at the same time, the following problems exist:

① Most scheduling algorithm strategies are based on single-processor platforms and cannot be directly used for task scheduling in multiprocessor environments.
② The main goal of the existing MCS scheduling method is to improve the success rate of key tasks. It directly discards low-critical tasks when the system is in a high-critical mode. This will
reduce the system's task acceptance rate and affect the consistency of system data. ③ For MCS, real-time scheduling results under the single-processor model cannot effectively guide real-time systems based on multi-core platform design, and the simple global scheduling or partition scheduling methods used in existing multi-processors also have drawbacks. That is, the former will generate high overhead due to task migration, and the latter will lead to inefficient use of processor resources. Therefore, based on the existing real-time scheduling research, this thesis will carry out research work around the multi-processor hybrid real-time scheduling algorithm, making it suitable for the hardware platform of multi-core processors.

2. Research content, research objectives and key issues to solve

2.1. Schedulability analysis

For hard real-time systems, the schedulability of the system needs to be analyzed according to the parameters and rules of the task, thus ensuring the time correctness of the system. The determination of system schedulability can be based on response time analysis. By calculating the response time (the time distance between the release time and the execution end time) of a task, it can be compared with its relative deadline to determine whether the task can be scheduled: If satisfied, the task can be scheduled, otherwise the task cannot be scheduled. Apply this process to each task in the task set so that the schedulability of the entire system can be judged.

2.2 Hybrid mission-critical semi-division scheduling in a multiprocessor environment

Global scheduling adds a lot of overhead and is more complicated to implement, while partition scheduling has low resource utilization due to uneven load. This paper adopts a semi-divided scheduling method that combines the advantages of two schedulings: Most of the tasks are assigned to a certain processor core for partition scheduling, and some tasks run on the cores of multiple processors.

2.3 Active treatment of low-key tasks

This paper will focus on some of the current research on hybrid critical systems, and research on real-time task scheduling strategy based on semi-division and active treatment of low-key tasks in a multi-processor platform environment. The study ultimately achieved better resource utilization and balanced load while ensuring system security.

3. Research methods, technical routes, experimental protocols

This paper first studies the mixed key system task model of compatible with periodic tasks and sporadic tasks. Based on the model compatible with these two tasks, a semi-divided real-time task scheduling mechanism is proposed. This process also considers the aggressive handling of low critical task to improve resource utilization. Finally, the relevant program code is designed and developed, and the performance of the proposed algorithm is evaluated by means of simulation and simulation. The correctness of the model and theory is verified, and the theory and protocol design are further improved.
Figure 1 Operation flow chart of hybrid critical system scheduling

3.1 Research plan and feasibility analysis of "multi-critical task priority assignment strategy"

First, the task $\tau_i$ in the hybrid critical system discussed in this topic consists of 4 tuples:

$$\tau_i = (T_i, D_i, C_i, L_i).$$

Here $T_i$, $D_i$, $C_i$, and $L_i$ are the period, deadline, estimated execution time, and critical level of task $\tau_i$, respectively. The real-time scheduling method we consider is based on priority-driven. Before scheduling analysis, we first need to construct a priority assignment function.

3.2 Construction of mission value and value density

Traditional real-time task values include system benefits, but in multi-critical systems, it is also necessary to combine mission-critical features. This feature is the result of a system with higher critical tasks than a mission with low criticality. Realize the immediate benefits of the system by constructing a dynamically changing value function $f_i(\tau_i)$. The value density of task $\tau_i$ is used to quantify the benefits that the task brings to the system per unit time:

- The design of the calculation formula for the value density $VD(\tau_i)$. We intend to use the remaining execution time $rt$ to measure, the formula is:

$$VD(\tau_i) = \alpha \frac{f_i(\tau_i)}{rt}$$

Here $f_i(\tau_i)$ is the task value, $rt$ is the remaining execution time, and $\alpha$ is the correction factor. It can be seen from this formula that the value density of the task increases with the execution of the task.
- Determine the value density correction factor $\alpha$. In the initial case, we let $\alpha = 1$, and then analyze according to the experimental feedback results, and then choose the value that is beneficial to improve performance.

- After completing the task's value density function construction, introduce it into the priority dispatch policy.

3.3 Mission criticality and priority

In general, high-critical tasks should be given higher priority. We intend to first reflect the criticality $L_i$ of task $\tau_i$ in the priority assignment function $f_p(\tau_i)$, such as:

$$f_p(\tau_i) = e^{\Delta L_i}$$  \hspace{1cm} (2)

In the formula (2), the exponential form is used to reflect the critical level, and the improvement of the critical $L_i$ will raise the priority of the task $\tau_i$. This basically guarantees the consistency of task priorities and criticality, thus avoiding priority inversion or critical inversion problems.

3.4 Mission urgency

The urgency $\Delta$ of task $\tau_i$ reflects the urgency of the available idle time. The shorter the idle time, the more urgent the task, which can be expressed as:

$$\Delta = \beta \frac{1}{lt(\tau_i)}$$  \hspace{1cm} (3)

Here $lt(\tau_i)$ refers to the spare time at time $t$, and $\beta$ is the urgency correction factor. When the system is initialized, let $\beta = 1$, after the end of the analysis of the system simulation results, select one $\beta$ that is conducive to improve system performance.

3.5 Comprehensive construction priority function problem

From the above analysis, the priority allocation function of this topic is to construct the dispatch function comprehensively considering the factors such as key $L_i$, value density $VD(\tau_i)$, and urgency $\Delta$. The basic steps are as follows:

- Construct a priority dispatch function. Combining the above formulas (1), (2) and (3) together, the priority function $f_p(\tau_i)$ of task $\tau_i$ is initially constructed:

$$f_p(\tau_i) = e^{\Delta L_i} \cdot \Delta \cdot VD(\tau_i)$$  \hspace{1cm} (4)

We conducted several experiments and then analyzed the system success rate, effective utilization, and system revenue to determine the priority dispatch function.

3.6 Research Analysis and Feasibility Analysis of "Response Time Analysis"

This paper assumes that the tasks in the system are all task sets that limit the deadline, that is, the deadline $D_i$ of task $\tau_i$ is less than its period $T_i$. The response time analysis is based on the concept of "k-busy period": k-busy period is a maximum continuous time region that satisfies the following
conditions: At any point in time in the time region, the processor is executing a task with a high priority.

1). Task interference analysis

The critical moments of the task in multiprocessor scheduling are unknown, so the interference experienced by the task during the busy period cannot be accurately calculated, and only an upper limit can be obtained approximately. The maximum load of a high priority task $\tau_i$ during the k-busy period can be divided into three parts:

(1) Front Task Instance Load: The load contributed by all task instances (front task instances) that were released before the start of the k-busy period and whose absolute deadline is within the k-busy period. There is at most one instance of the front task.

(2) Central task instance load: The load contributed by all task instances (middle task instances) with both release and absolute deadlines during the k-busy period.

(3) Rear task instance load: All loads that are released before the end of the k-busy period, and the absolute deadline is: k-the task instance (Rear task instance) after the end of the busy period.

2). Response time analysis

The critical improvement of the system affects the response time of the hybrid critical system. If the system is upgraded from low criticality to high criticality at time $t$:

(1) At time $t$, all high-critical tasks have been executed. At this time, without considering the impact of the promotion of high-critical tasks, the unfinished low-critical tasks are continuously executed in the scheduling window, and the response time of task $\tau_i$ can be calculated as:

$$R_i = \sum_{j \in hp(\tau_i)} \left[ \frac{R_j}{P_j} \right] \times C_j + C_i$$

Here $hp(\tau_i)$ denotes a task set with a higher priority than task $\tau_i$. Since the relative cut-off time limit is a key parameter, it can be known from Equation 5) that the calculation result of the task response time is not affected by the critical change.

(2) At time $t$, there are still high-critical tasks that have not been executed. After entering the high-critical mode, it should be analyzed whether the high-critical task satisfies its deadline $D_i$ in the current scheduling period, that is, whether the completion of the high-critical task is satisfied during the period of $(a + D_i - t)$, as discussed below:

If $a_i + D_i(HI) \geq a_i + R_i$, then the scheduling scheme being executed in the current scheduling window $[a_i, a_i + D_i(HI)]$ is still valid, and no other operations are required;

If $a_i + D_i(HI) < a_i + R_i$, first consider whether it is possible to allow the high-critical tasks to get enough time to complete the calculation by discarding the low-key tasks that have not been executed or not completed during the period $X_i$. Before time $t$, the execution time of the high-critical task is

$$C_i = \max \left( 0, (C_i - X_i + \sum_{j \in hp(\tau_i), j \neq i} \frac{X_j}{P_j} \times C_j) \right)$$
The condition $Con(a_j,t)$ indicates the release time of the higher priority task $\tau_j$, and the condition $a_j > t$ is satisfied. When the system is in the low-critical operating mode, the lower-priority, high-critical tasks that have been executed may still be preemptively executed during execution. Therefore, the execution time of the high critical task $\tau_j$ not completed can be defined as:

$$t + C_{i,t} \leq a_i + D_i(HI)$$  

In this case, the low-critical task to be executed during this time can be discarded to meet the deadline in the high-critical mode; if the formula (7) is not met, it is necessary to “deprive” the calculation time from the high-priority low-critical task before time t, because these tasks are interference tasks for high-critical tasks.

4. Summary

(1) It was originally pessimistic by obtaining a safe response time limit by very similar approximation of the behavior of the task set. This topic intends to use the problem window extension technology to establish a concept similar to the key moment for task scheduling. This means that the maximum response time for a task occurs when all high-priority tasks release the task instance at the same time. Therefore, it is only necessary to approximate the load of a small number of tasks, and complete the response time analysis under the multi-processor platform, thereby greatly improving the accuracy of the analysis.

(2) Currently, the scheduling of mixed key tasks mainly uses global scheduling and partition scheduling, but they have their own characteristics. This paper proposes semi-division scheduling based on the advantages of partition scheduling and global scheduling. That is, most tasks are allocated to a certain processor core for partition scheduling, and some tasks run on multiple processor cores. The scheduling strategy improves the task receiving rate and resource utilization of the system under the premise of ensuring high-critical task execution.

5. Acknowledgments

Key R&D Program of Gansu Science and Technology Program Project(18YF1FA122) The work is supported by Natural Science Foundation of China (No. 61662065) and Scientific and New Silk Road Economic project of Northwest University for Nationalities (No.XSCZL201605) Central colleges and universities Basic research business projects of Northwest Minzu University (No.31920160083)

References

[1] Estrin D, Govindan R, Heidemann J, Kumar S. Next century challenges: Scalable coordinate in sensor network. In: Proceedings of the 5th ACM/IEEE International Conference on Mobile Computing and Networking. Seattle: IEEE Computer Society, 1999,263~270.

[2] Heinzelman W. Application-Specific protocol architectures for wireless networks [Ph.D. Thesis]. Boston: Massachusetts Institute of Technology, 2000.

[3] Yan Jin,Ling Wang:An energy-efficient multi-level clustering algorithm for large-scale wireless sensor networks[J]. Computer networks,52(2008)542-562.
[4] LI Cheng-Fa, CHEN Gui-Hai: An Uneven Cluster-Based Routing Protocol for Wireless Sensor Networks[J]. Chinese journal of computers. Jan. 2007 vol. 30 No. 1.

[5] Younis O, Fahmy S. HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks[J]. IEEE Transactions on Mobile Computing. 2004, 3(4): 660-670.

[6] Al-Karaki JN, Ul-Mustafa R, Kamal AE. Data aggregation in wireless sensor networks—Exact and approximate algorithms. In: Proc. of the IEEE Workshop on High Performance Switching and Routing. Phoenix: IEEE Communications Society, 2004. 241-245.

[7] Paolo Santi. Topology Control in Wireless Ad.Hoc and Sensor Networks[M]. 2005

[8] Estrin D. Wireless sensor networks tutorial part IV: Sensor network protocols//Proceedings of the ACM Mobile Computing and Networking (MobiCom), Atlanta, GA