Quest Journals
Journal of Software Engineering and Simulation
Volume 8 ~ Issue 2 (2022) pp: 01-04
ISSN(Online) :2321-3795 ISSN (Print):2321-3809
www.questjournals.org

Research Paper

Review on scheduling mechanism of hybrid critical level system

Yong-xian Jin
College of Mathematics and Computer Science, Zhejiang Normal University, Jinhua Zhejiang 321004, China

Abstract: *Hybrid critical level system integrates high safety standards and non-safety critical tasks on the same platform to meet the development needs of current real-time system hardware platform and software functions. It has become one of the important topics in the field of embedded real-time system research. The traditional real-time system does not distinguish tasks with different importance levels, and the execution mode remains unchanged during operation. Its scheduling strategy only needs to ensure that all tasks are executed within the deadline, and can not adapt to the changes of system execution mode and task time attribute in time. The implementation of hybrid critical level system is more complex. It is necessary to consider the interaction of tasks with different importance levels in the implementation process, which brings new challenges to the research of task scheduling. This paper summarizes the classical model and related evolution model of hybrid critical level system, several static and dynamic priority scheduling algorithms, and focuses on the development process of scheduling algorithm based on response time.*

Keywords: *hybrid criticality system; Model; Scheduling mechanism*

Received** 01 Jan, 2022; **Revised** 20 Feb, 2022; **Accepted** 23 Feb, 2022 © The author(s) 2022. **Published** with open access at **www.questjournals.org

I. Hybrid Critical Level System Model

1.1 Traditional model

Most of the existing task models follow the hybrid key level system model [1] initially established by vestal, and use a limited set to represent the task set. The system has two key levels: LO and HI. Each task τ_i includes cycle T_i, deadline D_i, worst case execution time (WCET) and critical level (LO, HI). Set one of the parameters as a key parameter.

The key parameters have different values under different system key levels, which can be used for system static verification, that is, the schedulability of the system is estimated before the implementation of the system, and a reasonable scheduling scheme is proposed to meet the verification. There are two verification methods. One is designer verification to ensure the correctness of the whole system. Second, CA authentication with more conservative conditions [2], only ensures the correctness of high-level critical tasks. The requirement of meeting the designer's certification is low, and CA certification is carried out under the condition of pessimistic key parameters. At present, most literatures set the worst-case execution time as the key parameter, which is represented by a vector set (C_i (LO), C_i (HI)), as the index of static verification of the system at different levels.

1.2 The evolution of model

Considering that the system is affected by the external environment in the actual industrial application, the task execution mode changes in some way, not limited to the change of execution time, but also in other forms. Therefore, some researchers have eliminated the limitation of taking the worst-case execution time as a single key parameter and evolved a variety of flexible task models.

Reference [3-5] takes the task cycle as a key parameter, and high-level critical tasks shorten the release cycle after mode transformation. Burns et al. [6] proposed a model to optimize the execution of low critical tasks. After the system mode changes, the overall verification is still carried out, and its service level is reduced by reducing the execution time of low critical tasks or prolonging the release cycle. The key parameter in document [7] is still the worst-case execution time, but different from the traditional model, the WCET of LO task is set to a smaller value under CA authentication. Reference [8] takes the deadline as the key parameter. Reference [9] sets three key parameters: execution time, cycle and deadline.

II. Static Priority Scheduling Algorithm

The static priority allocation algorithm can assign priority to tasks before the system runs, and conduct schedulability analysis. It stipulates that the job released by all tasks each time has a fixed priority. This strategy has high stability.

Vestal first proposed two static priority allocation algorithms: cycle conversion strategy and scheduling strategy based on response time. The basic idea of cycle conversion is to reduce the execution cycle by using cycle segmentation method for high-level critical tasks, and then use DM strategy to schedule tasks, so as to ensure the correct execution of high-level critical tasks. The scheduling strategy based on response time adapts the traditional real-time response time formula proposed in literature [10] to the characteristics of hybrid critical level system, determines the response time of tasks in an iterative way, and the WCET value of all tasks is the verification condition corresponding to the current system critical level of the system, and assigns task priority in combination with the assignment method of OPA.

2.1 Response time algorithm of traditional real-time system

The traditional real-time system judges whether the task is schedulable by calculating the response time of the task in the worst case and comparing it with the deadline. If the task response time is less than the deadline, the task can be executed smoothly. Literature [8] provides the task τ_i response time calculation formula, as shown in formula (2-1), it can be seen that the total response time of the task is the sum of the task execution time C_i and the interference time I_j:

$$R_i = C_i + \sum_{j\in hp(i)} I_j \qquad (2\text{-}1)$$

Where, $hp_{(i)}$ is higher than the set τ_i task set, which can produce interference time. C_i is substituted by the worst-case execution time, so R_i reflects the estimation of the longest execution time of the task under pessimistic conditions.

2.2 Response time algorithm of hybrid critical level system

2.2.1 Vestal's response time scheduling algorithm

Vestal[1] first analyzed the response time by establishing a hybrid key system task model. On the basis of formula (2-1), assigned different WCETs of tasks under different execution modes to obtain τ_i response time calculation formula:

$$R_i = C_i + \sum_{j\in hp(i)} \left\lceil \frac{Ri}{Tj} \right\rceil * C_j(L_i) \qquad (2\text{-}2)$$

Where, L_i is key level parameter of τ_i ,also represents the current key level of the system. Task j belongs to the high priority task set hp(i), and R_i is obtained by cyclic iteration.

2.2.2 SMC (static mixed criticality)

Baruah [11] pointed out the shortcomings of vestal strategy in the research: the execution time of low critical level tasks may exceed the WCET (C_i(LO)) corresponding to their own critical level, but the system cannot capture this change. It is a kind of SMC without monitoring, and then proposed SMC with monitoring, which can monitor the change of task execution mode in real time to ensure that low critical level tasks do not exceed C_i (LO). The calculation formula is as follows:

$$R_i = C_i + \sum_{j\in hp(i)} C_j(\min(Li, Lj)) \qquad (2\text{-}3)$$

This strategy has certain limitations and can only be used to calculate the response time when the system is running stably, ignoring the details of the change of task execution in the system promotion stage.

2.2.3 AMC (adaptive mixed criticality) and AMC-max

In view of the shortcomings of SMC strategy, Baruah then proposed AMC and AMC-Max strategies. After the system enters the high key system, hang the low key tasks, focusing on the response time in the system promotion stage. AMC strategy makes up for the defects of SMC, reduces the interference time of low critical level tasks, reflects the impact details of system critical level changes on task execution, and calculates the response time in the promotion stage more accurately. AMC-max further considers the specific key level promotion points. High key level tasks present a low-key level execution mode before the promotion point, which reduces the interference time of high key level tasks, and the resulting response time is more accurate than AMC, which correspondingly has a large algorithm complexity. Formulas (2-4) and (2-5) are the calculation formulas of AMC static part and key level switching part respectively:

$$R_i(l) = C_i(l) + \sum_{j\in hp(i)|L_j\geq l} \left\lceil \frac{R_i(l)}{T_j} \right\rceil * C_j(l) \qquad (2\text{-}4)$$

$$R_i^*(l) = C_i + \sum\nolimits_{j\in \hbar p(i)|Lj\geq l} \left\lceil \frac{R_i^*(l)}{T_j} \right\rceil * C_j(l) + \sum\nolimits_{k\in \hbar p(i)|Lk<l} \left\lceil \frac{R_i(Lk)}{T_k} \right\rceil * C_k(Lk) \qquad (2\text{-}5)$$

2.2.4 Response time calculation scheme of hybrid multi-critical level system

Taking the multi-critical level system as the model, literature [12] extends the AMC and AMC-max strategies of the double critical level model respectively, and proposes two hybrid critical level scheduling strategies AMCarb-x and AMCmax-x, but AMCmax-x is only extended to three critical levels. Document [13] simplifies the analysis of the key level promotion time point and expands the scheme to L key levels. The calculation formula of AMCarb-x strategy is as follows:

$$R_i^*(l) = C_i + \sum\nolimits_{j\in \hbar p(i)|Lj\geq l} \left\lceil \frac{R_i^*(l)}{T_j} \right\rceil * C_j(l) + \sum\nolimits_{k\in \hbar p(i)|Lk<l} \left\lceil \frac{R_i(Lk)}{T_k} \right\rceil * C_k(Lk) \qquad (2\text{-}6)$$

2.3 Other scheduling algorithms based on response time

Literature [3] establishes a hybrid critical level model with pessimistic cycle as the key parameter, and puts forward the corresponding scheduling scheme. Zhang n [4] further analyzed the specific time point of critical level promotion, proposed SAMC scheduling strategy, improved the accuracy of response time calculation, and extended the strategy to multi critical-level systems.

Bai EC [14] changed the traditional response time calculation method and proposed AMC-PM algorithm, which only cares about the execution time s of the task itself when the system mode changes, and obtains the response time by traversing all possible values of s. The algorithm reduces the complexity of the algorithm compared with AMC-max, and the scheduling ratio is higher than AMCarb. It is an algorithm that combines the advantages of the two.

Burn [15] proposed a fault-tolerant system framework to analyze task schedulability through response time. The framework allows several jobs of a task to be overloaded. Define three execution modes: normal, robust and HI-criticality. In the robust mode, a job of a task is allowed to be discarded, and two fault-tolerant indexes F and M are used as the marks of mode conversion respectively (F, M represents the maximum overload job allowed in the normal and robust modes). If the overload job exceeds F, it enters the robust mode, and if it exceeds M, it enters the high critical level mode, The designer can adjust F and M according to the actual situation to balance the fault tolerance and schedulability of the system.

III. Dynamic priority allocation strategy

Different from the static priority scheduling strategy, the dynamic priority scheduling strategy is characterized in that the task priority will be adjusted according to the change of deadline urgency.

EDF [16] is an optimal dynamic priority allocation algorithm for traditional real-time systems. EDF algorithm can not reflect the impact on scheduling for key level changes, which may lead to key level reversal. For this, literature [17] proposed hybrid key system dynamic priority algorithm EDF-VD (virtual deadline). The basic idea is to set an extended parameter to construct the virtual deadline of high-level key tasks, and adopt EDF strategy for scheduling according to the new deadline of task set, It can reflect the characteristics of priority scheduling of high-level critical tasks, and use the speed factor α to measure the efficiency of the algorithm, the smaller the speed factor, the higher the performance of the algorithm, α is 1.618. Reference [18] further optimized the EDF-VD algorithm and reduced the acceleration factor to 4/3. Then, in reference [19], the EDF-VD algorithm is extended on the multiprocessor system. Combined with the FP-EDF algorithm scheduling of multiprocessors, the setting methods of extension parameters under the global and partition scheduling modes of processors are discussed respectively, and the restriction of single extension parameters is cancelled. The extension parameters that meet the conditions can be determined by the search method to improve the scheduling performance of the system.

Reference [20] extended the EDF-VD algorithm of multi critical level system, proved that the acceleration factors of implicit deadline and arbitrary deadline are 4/3 and 1.866 respectively, and proposed EDF-NUVD algorithm, which sets different extension parameters for each task, which has better scheduling performance than EDF-VD.

Reference [9] divides the system execution process into three stages. In the preparation stage, the system load is calculated by using the demand upper limit function [21], and the extension parameters are set according to different load conditions to form different virtual deadlines. Reference [22] increases the selection range of extension parameters, tests the minimum schedulable value of the system EDF through the search method, and improves the schedulability of the task set.

Reference [7] proposes a variable execution rate algorithm, which sets the execution rate parameters θ_{iL} and θ_{iHI} when the system is at LO and HI levels respectively, adjust the WCET of the task, realize the priority scheduling of high-level critical tasks after the system critical level is improved, and improve the performance compared with EDF-VD.

IV. Conclusion

Although great achievements have been made in the research of mixed critical system, it has not met the needs of software and hardware development in actual industry. For example, there is much room for improvement in how to make better use of processor resources and improve the performance of the system.

On the one hand, in order to meet the relatively conservative verification, most scheduling strategies estimate the schedulable ratio offline with the worst case to ensure the smooth execution of high critical level tasks. When the critical level is promoted, the execution time of low critical level tasks is sacrificed, but simply discarding the low critical level tasks may cause data integrity damage and generate some additional consumption. In practical applications, processor resources are usually not fully utilized, and the task execution time is far less than the worst-case execution time. Therefore, the existing scheduling schemes established by static analysis are too negative, and more active scheduling methods need to be further studied.

On the other hand, the focus of the current research on mixed critical system is to ensure the correct execution of tasks in the critical level promotion phase of the system, and the system entering the high critical level mode is a special state. If it is always maintained in this state, it is not in line with the actual situation, so the system also exists in the process of falling from high critical level to low critical level.

Reference

[1]. Vestal S. Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance[C]//Real-Time Systems Symposium, 2007. RTSS 2007. 28th IEEE International. IEEE, 2007: 239-243.

[2]. Burns A , Baruah S K . Timing Faults and Mixed Criticality Systems[C]// Dependable and Historic Computing - Essays Dedicated to Brian Randell on the Occasion of His 75th Birthday. Springer-Verlag, 2011.

[3]. Baruah S, Chattopadhyay B. Response-time analysis of mixed criticality systems with pessimistic frequency specification[C]// IEEE, International Conference on Embedded and Real-Time Computing Systems and Applications. IEEE, 2014:237-246.

[4]. Zhang N, Xu C, Li J, et al. A sufficient response-time analysis for mixed criticality systems with pessimistic period[J]. Journal of Computational Information Systems, 2015, 11(6):1955-1964.

[5]. Baruah S. Schedulability analysis of mixed-criticality systems with multiple frequency specifications[C]//2016 International Conference on Embedded Software (EMSOFT). IEEE, 2016: 1-10.

[6]. Burns A, Baruah S. Towards a more practical model for mixed criticality systems[C]//Workshop on Mixed-Criticality Systems (colocated with RTSS). 2013.

[7]. Baruah S , Burns A , Guo Z . Scheduling Mixed-Criticality Systems to Guarantee Some Service under All Non-erroneous Behaviors[C]// 2016 28th Euromicro Conference on Real-Time Systems (ECRTS). IEEE, 2016.

[8]. Huang Lida, Li Renfa Research on hybrid key level real-time task scheduling with deadline as key parameter [J] Computer research and development (7): 1641-1647

[9]. Baruah S . Schedulability Analysis for a General Model of Mixed-Criticality Recurrent Real-Time Tasks[C]// Real-time Systems Symposium. IEEE, 2017.

[10]. Joseph, M. Finding Response Times in a Real-Time System[J]. The Computer Journal, 1986, 29(5):390-395.

[11]. Baruah S K, Burns A, Davis R I. Response-Time Analysis for Mixed Criticality Systems[C]// Real-time Systems Symposium. IEEE Computer Society, 2011:34-43.

[12]. Fleming T. Extending mixed criticality scheduling[D]. University of York, 2013.

[13]. Li L, Li R, Huang L, et al. A New RTA Based Scheduling Algorithm for Mixed-Criticality Systems[C]// IEEE, International Conference on Computational Science and Engineering. IEEE Computer Society, 2013:722-729.

[14]. Bai EC, Zhang WZ. Method of response-time analysis for mixed criticality systems. Ruan Jian Xue Bao/Journal of Software, 2015,26(Suppl.(2)):257−262.

[15]. Burns A, Davis R I, Baruah S, et al. Robust mixed-criticality systems[J]. IEEE Transactions on Computers, 2018, 67(10): 1478-1491.

[16]. Liu C L, Layland J W. Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment[J]. Journal of the Acm, 1973, 20(1):46-61

[17]. Baruah S K , Bonifaci V , D'Angelo G , et al. Mixed-Criticality scheduling of sporadic task systems[C]// European Conference on Algorithms. Springer-Verlag, 2011.

[18]. Baruah S, Bonifaci V, DAngelo G, et al. The preemptive uniprocessor scheduling of mixed-criticality implicit-deadline sporadic task systems[C]//2012 24th Euromicro Conference on Real-Time Systems. IEEE, 2012: 145-154.

[19]. Baruah S ,Chattopadhyay B , Li H , et al. Mixed-criticality scheduling on multiprocessors[J]. Real-Time Systems, 2014, 50(1):142-177.

[20]. Socci D , Poplavko P , Bensalem S , et al. Mixed Critical Earliest Deadline First[J]. Euromicro Workshop on Real-Time Systems-Proceedings, 2013, 8114:93-102.

[21]. Zhao R, Zhu Y, Lian L I. Mixed-criticaLity Task ScheduLing ALgorithm Based on Heterogeneous MuLti-core System[J]. Computer Engineering, 2018:51-55.

[22]. Baruah S. Schedulability analysis of mixed-criticality systems with multiple frequency specifications[C]//2016 International Conference on Embedded Software (EMSOFT). IEEE, 2016: 1-10.