Journal of Southeast University (English Edition) Vol. 22, No. 4, pp. 484 – 489 Dec. 2006 ISSN 1003—7985

Long-release-interval-first real-time scheduling algorithm and its schedulability test

Shen Zhuowei Wang Yun

(Key Laboratory of Computer Network and Information Integration of Ministry of Education, Southeast University, Nanjing 210096, China)

Abstract: To fulfill the requirements for hybrid real-time system scheduling, a long-release-interval-first (LRIF) real-time scheduling algorithm is proposed. The algorithm adopts both the fixed priority and the dynamic priority to assign priorities for tasks. By assigning higher priorities to the aperiodic soft real-time jobs with longer release intervals, it guarantees the executions for periodic hard real-time tasks and further probabilistically guarantees the executions for aperiodic soft real-time tasks. The schedulability test approach for the LRIF algorithm is presented. The implementation issues of the LRIF algorithm are also discussed. Simulation result shows that LRIF obtains better schedulable performance than the maximum urgency first (MUF) algorithm, the earliest deadline first (EDF) algorithm and EDF for hybrid tasks. LRIF has great capability to schedule both periodic hard real-time and aperiodic soft real-time tasks.

Key words: real-time scheduling; schedulability test; earliest deadline first; maximum urgency first; long release interval first

Classical real-time scheduling theories only focus on scheduling for periodic hard real-time tasks[1−2]. A large class of scheduling algorithms belongs to priority-driven approaches, which are efficient to schedule real-time systems[3]. According to different policies to assign priorities to jobs, priority-driven scheduling algorithms fall into two categories: fixed priority scheduling and dynamic priority scheduling. The rate monotonic (RM) algorithm proposed by Liu and Layland[4] is the well-known fixed priority scheduling algorithm. And the earliest deadline first (EDF) algorithm also proposed by Liu and Layland[4] is a typical dynamic priority scheduling algorithm. Generally speaking, the dynamic priority scheduling algorithm can achieve a higher processor utility than the fixed priority scheduling algorithm can[5]. It is proved that the EDF algorithm can schedule any set of periodic tasks with processor utilization no larger than 100%. So it is the optimal scheduling algorithm.

With the development of real-time technique and its applications, there are more and more requirements for hybrid real-time systems, in which the real-time tasks may be either hard or soft, either periodic or aperiodic[6]. There exist some algorithms for scheduling both periodic and aperiodic real-time tasks, such as the DS algorithms[7] and the EDF-based aperiodic scheduling algorithms[8]. However, these algorithms are only suitable for hard real-time systems. The maximum urgency first (MUF) algorithm[9] is an algorithm for hybrid real-time systems to schedule both hard and soft periodic real-time tasks.

Essentially, MUF is a combination of fixed and dynamic priority scheduling, also called mixed priority scheduling. It can support a schedulable bound of up to 100% for the critical set.

However, with the MUF algorithm, there is no guarantee how many jobs of soft real-time tasks will meet their deadlines. In fact, the MUF scheduler makes hard real-time tasks to be executed as soon as possible, no matter it is necessary or not. Meanwhile it postpones the execution of soft real-time tasks more or less. So MUF is not suitable for applications including soft real-time tasks. Besides, MUF cannot be used to schedule aperiodic real-time tasks.

In this paper, in order to find a better solution for hybrid real-time tasks, a long-release-interval-first task scheduling algorithm named LRIF is proposed. It can not only support the scheduling of periodic hard real-time tasks, but also probabilistically guarantee the schedulability ratios of aperiodic soft real-time tasks. The principle of LRIF is to reasonably improve the

Received 2006-09-08.

Foundation item: The Natural Science Foundation of Jiangsu Province (No. BK2005408).

Biographies: Shen Zhuowei (1974—), male, doctor, zwshen@seu.edu.cn; Wang Yun(1966—), female, doctor, professor, yunwang@seu.edu.cn.

schedulable task ratio by privileged scheduling jobs with longer release interval. For those jobs with shorter release interval, called "burst jobs", lower priorities are assigned. It is allowable for these burst jobs to miss their deadlines. The schedulability test approach is also deduced with priori release interval probability distribution for soft real-time jobs.

1 Real-Time Scheduling Model and LRIF Algorithm

1.1 Model

In this paper, we consider the scheduling of a real-time system on single processor. The task set running on the processor consists of periodic hard real-time tasks and aperiodic soft real-time tasks.

We use $T_i = (p_i, e_i, d_i)$ to denote a periodic hard real-time task. p_i is a period of task T_i. e_i is the worst-case execution time. And $d_i \leqslant p_i$ is the relative deadline of T_i. The k-th job of T_i can be denoted as $J_{i,k} = (p_i, e_i, D_{i,k})$, $D_{i,k}$ is the absolute deadline of $J_{i,k}$.

Suppose the release intervals of an aperiodic soft real-time task subject to a certain probability distribution, we can use $T_j = (\mathrm{Dstrb}_j, e_j, d_j)$ to denote the aperiodic soft real-time task. Dstrb_j is the probability distribution of T_j's release interval. e_j and d_j are the worst-case execution time and relative deadline of T_j, respectively. The k-th job of T_j can be denoted as $J_{j,k} = (e_j, D_{j,k})$, $D_{j,k}$ is the absolute deadline of $J_{j,k}$.

For an aperiodic soft real-time task $T_j = (\mathrm{Dstrb}_j, e_j, d_j)$, given a probability p, which we call permitted deadline miss rate, if the ratio of the missed jobs to the total jobs is lower than p, we call T_i is feasible.

1.2 LRIF algorithm

The basic requirement for the real-time system is predictability. Unless the behavior of a real-time system is predictable, a scheduler cannot guarantee that the computation deadlines of tasks will be met. The idea of the LRIF algorithm is to encourage jobs of aperiodic soft real-time tasks to arrive smoothly. Jobs with steady arrival will be set to higher priorities and scheduled as hard real-time jobs. Meanwhile, those burst jobs will be set to lower priorities, and will not be scheduled until the jobs with higher priorities are arranged.

With LRIF, a task priority consists of two parts: basic priority and dynamic priority. The dynamic priority is inversely proportional to the deadline of a job. For a hard real-time task, the policy is to assign high basic priority to all its jobs. A soft real-time task is associated with a base release interval L. To assign basic priorities to jobs of soft real-time tasks, the following policy is applied: If the interval between the release time of a job and the release time of its immediate precedent is shorter than L, the policy is to assign low basic priority to the job. Otherwise, high basic priority is assigned.

Suppose that an aperiodic soft real-time job's absolute deadline equals the sum of its release time and a fixed relative deadline. The probability distribution of a task's absolute deadline interval is the same as that of the task's release intervals. So, the policy to assign the basic priority for soft real-time jobs can also be presented as follows: If the interval between the absolute deadline of a job and the absolute deadline of its immediate precedent job is shorter than L, the policy is to assign low basic priority to the job. Otherwise, high basic priority is assigned.

Whenever a job is released or the current scheduled job finished, a reschedule operation is performed. The LRIF scheduler is used to determine the next task for execution. It first selects a job with the most urgent deadline among jobs with high basic priority. Without any jobs with high basic priority, a job with low basic priority can be selected according to the EDF scheduling policy.

LRIF consists of two parts. The first part assigns basic priorities to jobs and pushes them into ready queues. The second part involves the actions of the LRIF scheduler during run-time. The following pseudo codes illustrate these two parts.

Part 1 Assignment of the basic priorities

Input: Job $J_{i,k}$ belongs to task T_i. If T_i is an aperiodic soft real-time task, $J_{i,k} = (e_i, D_{i,k})$. If T_i is a periodic hard real-time task, $J_{i,k} = (p_i, e_i, D_{i,k})$.

Output: Basic priority of $J_{i,k}$.

```
if (T_i is hard real-time task){
  basic_prior(J_{i,k}) = High;
  Push J_{i,k} into High_basic_prior_queue according to EDF policy;
} else{
  if (D_{i,k} - D_{i,k-1} ≥ L_i) //L_i is the base release interval of task T_i{
      basic_prior(J_{i,k}) = High;
      Push J_{i,k} into High_basic_prior_queue according to EDF policy;
  } else {
      basic_prior(J_{i,k}) = Low;
      Push J_{i,k} into Low_basic_prior_queue according to EDF policy;
  }
}
return basic_prior(J_{i,k});
```

Part 2 Reschedule operation of LRIF

```
Input: Real-time task set{T_i}
Output: None
while (true){
   if ((new job is released)//(currently scheduled job is finished)) {
         if (High_basic_prior_queue is not empty){
                Get the job from the head of High_basic_prior_queue, and schedule it for execution;
               continue; //go to next loop
         }else {//There is no job with high basic priority
         Get the job from the head of Low_basic_prior_queue, and schedule it for execution;
         continue; //go to next loop
      }
   }
}
```

In part 1, given the length of the ready queue n, the time complexity to push a job into a ready queue is $O(\log n)$. So the time complexity of part 1 is $O(\log n)$. It is equal to the time complexities of EDF and MUF. Considering that jobs with low basic priority are permitted to miss their deadlines, a job with low basic priority may be directly inserted at the tail of low_basic_prior_queue, which costs a time of $O(1)$. The time complexity of part 2 is just $O(1)$. Therefore, the time complexity of LRIF is the same as that of EDF and MUF.

With LRIF, soft real-time jobs with the release interval longer than the base release interval are assigned high basic priorities and scheduled as hard real-time jobs. These jobs will not miss their deadlines. For the hard real-time tasks, the scheduling behavior of LRIF is the same as that of EDF. Hence the schedulability test approach for EDF based hard real-time systems can also be applied to the systems scheduled with LRIF.

Since some soft real-time jobs have the same basic priorities as hard real-time jobs according to their release intervals, it must be careful to develop an efficient schedulability test algorithm for soft real-time tasks and to determine their base release intervals, in order not to incur the miss of hard real-time jobs' deadlines.

2 Schedulability Analysis for LRIF

The schedulability test problem for an aperiodic real-time task can be stated as follows: Given the permitted deadline miss rate, and the release interval probability distribution of a task set, is the task set feasible with a certain scheduling algorithm?

2.1 Schedulability test approach for LRIF

In order to deduce the schedulability test algorithm for aperiodic soft real-time tasks, we first propose and prove the following theorem.

Theorem 1 Suppose that the release interval of a task $T_i = (\mathrm{Dstrb}_i, e_i, d_i)$ is less than $L_i (L_i \geqslant d_i)$ with a probability p. If a task set is feasible with EDF, in which T_i is replaced by a periodic real-time task $T_i' = (L_i, e_i, d_i)$, then the jobs of T_i will miss their deadlines with a probability not more than p, when the LRIF algorithm is applied and the base release interval of T_i is L_i.

Proof All the jobs of T_i belong to two sets: S_1 and S_2. All the jobs in S_1 have release intervals not shorter than L_i. All the jobs in S_2 have release intervals shorter than L_i. With LRIF, all the jobs in S_2 are set to a low basic priority. These jobs have no effect on the feasibility of jobs with a high basic priority. All the jobs in S_1 are set to a high basic priority. For the jobs with a high basic priority, the scheduling result with LRIF is the same as that with EDF. Since T_i' is feasible with EDF, and the jobs in S_1 have the release intervals not shorter than L_i, which is the release interval of T_i', all the jobs in S_1 keep their deadlines. Considering that some jobs in S_2 may meet their deadlines. $P\{J_{i,k}$ misses its deadline$\} \leqslant P\{J_{i,k} \in S_2\}$, according to the definition of S_2, $P\{J_{i,k} \in S_2\} = p$. Therefore, $P\{J_{i,k}$ misses its deadline$\} \leqslant p$.

The schedulability test approach for aperiodic soft real-time tasks with LRIF can be derived from theorem 1:

1) For every aperiodic soft real-time task T_i in the task set, given the permitted deadline miss rate p_i^{miss}, set the task's base release interval L_i according to the probability distribution of task's release interval. That means to set the base release interval L_i, so that the job's release interval is shorter than L_i with the possibility of p_i^{miss}.

2) For every aperiodic soft real-time task in the task set, replace the task by its corresponding periodic hard real-time task, in which the period equals the base release interval and the deadline and the worst-case execution time remain the same.

3) Test the feasibility of the task set with the schedulability test algorithm for EDF. If the task set is feasible with EDF, then the original task set is feasible with LRIF; i. e., the aperiodic tasks are feasible with LRIF.

The schedulability test for periodic hard real-time systems with EDF can be made based on processor utilization according to Ref. [4]. It has the time complexity of $O(n)$, in which n refers to the task number in a system. So, the schedulability test approach for aperiodic soft real-time tasks has the time complexity of $O(n)$.

2.2 Assignment of the base release interval

Assignment of the base release intervals for aperiodic soft real-time tasks is critical in the LRIF scheduling algorithm and its schedulability test. The following example illustrates how to set the base release interval according to the probability distribution of a task's release interval and the permitted deadline miss rate.

Suppose that the job arrival rate of an aperiodic soft real-time task has a Poisson distribution with λ. The release interval of the task subjects to a negative exponential distribution[10]. The probability distribution function is $F(t) = 1 - e^{-\lambda t}$.

Suppose that the base release interval of T_i is L_i, if the permitted deadline miss rate is p, according to theorem 1:

$$P\{L < L_i\} = p \tag{1}$$

where L denotes the release interval of T_i.

Because L subjects to the negative exponential distribution,

$$P\{L < L_i\} = F(L_i) = 1 - e^{-\lambda L_i} \tag{2}$$

Combining Eqs. (1) and (2),

$$1 - e^{-\lambda L_i} = p$$

Therefore, $L_i = \ln\left(\frac{1}{1-p}\right)/\lambda$.

3 Implementation issues of LRIF

To implement the LRIF scheduling algorithm, a scheduling framework illustrated in Fig. 1 is developed. The framework consists of two parts: the LRIF admission controller and the LRIF scheduler.

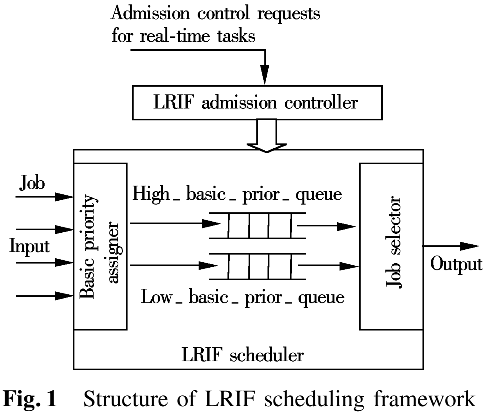

Fig. 1 Structure of LRIF scheduling framework

The LRIF admission controller is in charge of schedulability test. The schedulability test approach is presented in section 2. When a new task asks for entering the system, it makes a decision for admission. A task is rejected if it is unfeasible. Otherwise, LRIF accepts it, and stores the real-time parameters such as task type, period or base release interval, relative deadline, and worst-case execution time, into LRIF scheduler.

The jobs of a task can be scheduled by the LRIF scheduler and guaranteed to be feasible, when the task is admitted.

There are two ready queues in the LRIF scheduler: a queue for jobs with a high basic priority, and a queue for jobs wtih a low basic priority. The LRIF scheduler takes two steps.

Step 1 Assign the basic priorities to the arriving jobs and push them into either of the two ready queues.

Step 2 Select jobs from the two ready queues for execution.

Step 1 is done by the basic priority assigner. Step 2 is done by the job selector. The details of both of them are presented in section 1.2.

4 Simulations and Analysis

To evaluate the capabilities of LRIF to guarantee the feasibilities of both periodic hard real-time and aperiodic soft real-time tasks, the following simulation is designed.

The real-time task set consists of five periodic hard real-time tasks and two aperiodic soft real-time tasks. The total processor utility of the five periodic hard real-time tasks reaches 50%. One aperiodic soft real-time task subjects to a normal distribution, the other subjects to a uniform distribution. To simplify the discussion, the processor utilities of the two aperiodic soft real-time tasks are the same, which vary between 5% and 25% in different cases.

The task set is scheduled by LRIF and other scheduling algorithms respectively. The deadline miss rates of tasks are recorded. Because the MUF scheduling algorithm can only schedule periodic tasks, it was not chosen as the contrast algorithm. In this simulation, pure EDF (denoted as EDF) and EDF for hybrid real-time tasks (noted as H-EDF) are compared. H-EDF schedules the task set with the following policy: Hard real-time jobs are scheduled with EDF as long as these jobs exist. Otherwise, soft real-time jobs are scheduled with EDF.

The deadline miss rates of the tasks with EDF are shown in Tab. 1. The deadline miss rates of the tasks with EDF and H-EDF are shown in Tab. 2 and Tab. 3, respectively.

Tab. 1 Deadline miss rates of tasks with LRIF

Task ID	Case number				
	1	2	3	4	5
T_1	0	0	0	0	0
T_2	0	0	0	0	0
T_3	0	0	0	0	0
T_4	0	0	0	0	0
T_5	0	0	0	0	0
T_6	0. 027	0. 03	0. 044	0. 068	0. 097
T_7	0	0	0	0	0. 012
Processor utility/%	60	70	80	90	100

Tab. 2 Deadline miss rates of tasks with EDF

Task ID	Case number				
	1	2	3	4	5
T_1	0	0	0	0	0
T_2	0	0	0	0. 018	0. 04
T_3	0	0	0	0	0. 01
T_4	0	0	0	0	0. 02
T_5	0	0	0	0	0. 01
T_6	0. 011	0. 03	0. 033	0. 065	0. 093
T_7	0	0	0	0	0. 036
Processor utility/%	60	70	80	90	100

Tab. 3 Deadline miss rates of tasks with H-EDF

Task ID	Case number				
	1	2	3	4	5
T_1	0	0	0	0	0
T_2	0	0	0	0	0
T_3	0	0	0	0	0
T_4	0	0	0	0	0
T_5	0	0	0	0	0
T_6	0. 031	0. 052	0. 086	0. 105	0. 12
T_7	0	0	0. 024	0. 036	0. 059
Processor utility/%	60	70	80	90	100

As shown in Tabs. 1 to 3, from case 1 to case 3, all the tasks are feasible with any of the three algorithms. That means all the three algorithms are satisfying when the load is not heavy. When the load turns heavy, the difference appears. With EDF, no matter hard or soft, all the jobs are scheduled according to their absolute deadlines. So jobs of periodic hard real-time tasks may miss their deadlines, while the deadline miss rates of aperiodic soft real-time tasks are low. With H-EDF, jobs of the hard real-time tasks have the precedence, so all the periodic hard real-time tasks are feasible. However, there's no guarantee for soft real-time jobs, which incurs larger deadline miss rates of aperiodic soft real-time tasks. With LRIF, all the periodic hard real-time tasks are guaranteed feasible. At the same time, the deadline miss rates of the aperiodic soft real-time tasks are lower than the permitted deadline miss rates.

5 Conclusion

In this paper, an EDF-based real-time scheduling algorithm named LRIF is proposed. With LRIF, both periodic hard real-time and aperiodic soft real-time tasks are feasible. The time complexity of LRIF is the same as that of EDF and MUF. The schedulability test approach for LRIF is developed. The implementation issues of this scheduling algorithm are also discussed. Simulation result verifies its capability to guarantee the feasibilities of both periodic hard real-time and aperiodic soft real-time tasks.

The LRIF scheduling algorithm is flexible and extensible. Currently, only periodic hard real-time tasks and aperiodic soft real-time tasks can be scheduled with LRIF. In order to schedule periodic soft real-time tasks with LRIF, little improvement need to be done in LRIF scheduling framework. Because the release intervals of periodic soft real-time tasks are fixed, in basic priority assigner, other policy needs to be applied so that a certain proportion of periodic soft real-time jobs will be assigned a high basic priority. The schedulability test approach needs to be adjusted correspondingly.

References

[1] Krishna C M, Shin K G. *Real-time systems* [M]. Columbus, Ohio: McGraw-Hill Companies, Inc. , 1997. 47 − 72.

[2] Stankovic J, Spuri M, Di Natale M, et al. Implications of classical scheduling results for real-time systems [J]. *IEEE Computer Magazine*, 1995, **28**(6): 16 − 25.

[3] Liu J W S. *Real-time systems* [M]. New Jersey: Prentice-Hall, Inc. , 2000.

[4] Liu C L, Layland J. Scheduling algorithms for multiprogramming in a hard real-time environment [J]. *Journal of ACM*, 1973, **20**(1): 46 − 61.

[5] Buttazzo G C. Rate monotonic vs. EDF: judgment day [J]. *Journal of Real-Time Systems*, 2005, **29**(1): 5 − 26.

[6] Shen Zhuowei. Research on hybrid real-time CORBA scheduling model and its schedulability [D]. Nanjing: School of Computer Science and Engineering of Southeast University, 2004. (in Chinese)

[7] Strosnider J K, Lehoczky J P, Sha L. The deferrable server algorithm for enhanced aperiodic responsiveness in hard real-time environments [J]. *IEEE Transactions on Computers*, 1995, **44**(1): 73 − 91.

[8] Ripoll I, Garcia-Fornes A, Crespo A. Optimal aperiodic scheduling for dynamic-priority systems [A]. In: *Proceedings of the IEEE International Workshop on Real-Time Computing Systems and Applications* (*IEEE RTCSA*) [C]. Seoul, 1996. 294 − 300.

[9] Stewart D B, Khosla P K. Real-time scheduling of sensor-

based control systems [A]. In: Halang W, Ramamaritham K, eds. *Real-Time Programming*[C]. New York: Pergamon Press Inc., 1992. 139 − 144.

[10] Lu Chuanlai. *Queuing theory* [M]. Beijing: Beijing University of Posts and Telecommunications Press, 1994. (in Chinese)

长释放时间间隔优先调度算法及其可调度性分析

沈卓炜　　汪　芸

(东南大学计算机网络和信息集成教育部重点实验室,南京 210096)

摘要:针对混合任务实时调度的需求和现有混合任务实时调度算法的局限性,提出了一种长释放时间间隔优先的混合任务实时调度算法 LRIF,该算法采用固定优先级和动态优先级相结合的调度策略,根据非周期性软实时任务作业到达流分布,将一定比例的软实时任务作业当作硬实时作业调度,除了可对周期性硬实时任务提供调度保证外,同时还可确保非周期性软实时任务的可调度率.还提出了 LRIF 调度算法的可调度性分析方法,并讨论了 LRIF 调度算法的实现方法.测试结果表明:LRIF 调度算法具有更好的调度性能,可有效调度由周期性硬实时任务和非周期性软实时任务构成的混合实时系统.

关键词:实时调度;可调度性分析;截止期优先调度;最大紧急度优先调度;长释放时间间隔优先调度

中图分类号:TP393