TELKOMNIKA Indonesian Journal of Electrical Engineering
Vol.12, No.6, June 2014, pp. 4444 ~ 4450
DOI: 10.11591/telkomnika.v12i6.5480

A Kind of New Real-time Scheduling Algorithm for Embedded Linux

Li Yang*, Qingyan Zhu
Hunan College of Information, Changsha, China, 410200
*Corresponding author, e-mail: 4104762@qq.com

Abstract

To solve the insufficiency of real time performance in the present real time scheduling algorithm, a new real time task classification scheduling algorithm is proposed. According to the classification of arrival of real time tasks, this algorithm is divided into periodic tasks and non-periodic tasks, and uses different improved real time scheduling algorithm to schedule different type of real time tasks. Comparing with the present other real time scheduling algorithms, experiments show this algorithm has been largely improved for integrated real time performance.

Keywords: *embedded, real time scheduling algorithm, classified scheduling algorithm, periodic task, non-periodic tasks*

1. Introduction

Linux is widely applied in embedded system, real time control and other area with the advantages of open source code and the system stability. But Linux belongs to general time-sharing operating system, real-time performance is poor, so it is needed to improve for the application of real-time environment [1]. Real-time scheduling algorithm is one of the important factors to influence system real-time performance [2], according to the mechanism of scheduling drive; real-time scheduling algorithm can be divided into three categories [3]: Time-Driven scheduling algorithm (TD), Priority-Driven scheduling algorithm (PD), and Sharing-driven scheduling algorithm (SD).

Priority-driven scheduling algorithm is the most popular scheduling algorithm; its principle is to assign each task a priority, while in every task scheduling to select the highest priority task to perform. It has two types of static and dynamic priority scheduling algorithms, the typical representative algorithms are RM and EDF scheduling algorithms [4], and there are some improved algorithms on this basis, such as NSRL [5], LLF [6] etc.

Currently in embedded Linux real-time scheduling algorithm is added to improve the real-time performance of the embedded system, but in scheduling the hybrid real-time task, this method exists a lot of defects, when many scheduling algorithms schedule the hybrid task set including periodic tasks and aperiodic tasks, only simply regard the aperiodic tasks as special periodic tasks, make the real-time performance dropped substantially. In fact according to the nature of different task, real-time tasks can be divided into different types [7], for example, according to the arrival of the task, the task can be divided into periodic tasks and non-periodic tasks, periodic task refers to reach and request the operation of task according a certain period, aperiodic tasks refers to the task of random arrival system. This paper adopts a new real-time task classification scheduling algorithm, static and dynamic scheduling algorithms are respectively used in the periodic tasks and aperiodic tasks, to maximally improve real-time performance of system.

2. Classification Scheduling Algorithm

The general framework of the classification scheduling algorithm is shown in Figure 1, the dynamic scheduling algorithm is used to schedule a periodic real-time task, and the static scheduling algorithm is used to schedule periodic real-time task.

Received December 22, 2013; Revised February 15, 2014; Accepted March 1, 2014

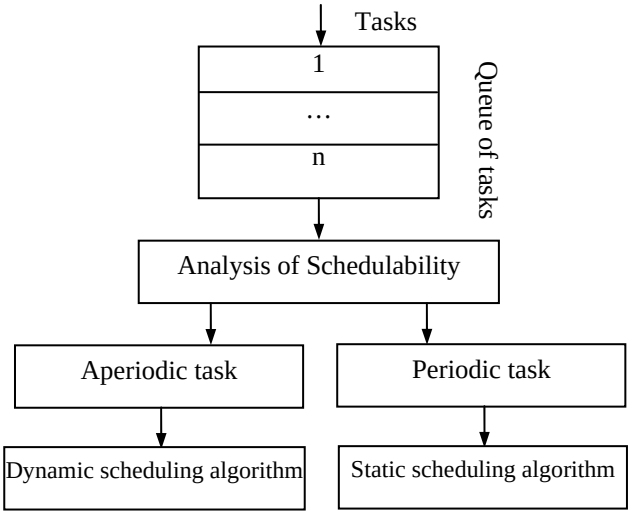

Figure 1. The Idea of Classification Scheduling Algorithm

In the proposed dynamic overmodulation method, the dynamic condition is defined when the torque error exceeds 5% of rated torque. As the dynamic condition is encountered, the original stator flux error status, ψ^+ is modified based on information of flux position, θ_ψ to produce the appropriate flux error status ψ^-. In this way, the active voltage vector that produces the largest tangential flux component is switched and held on, to create the largest increase in load angle and hence rapid dynamic torque.

The distribution steps of the classified scheduling algorithm are as follows:

(1) To do schedulability analysis to the new arrival real-time task, if it is a periodic task to operate step (2), otherwise turn to step (3).

(2) To do schedulability analysis of dynamic scheduling algorithm to the new arrival aperiodic real-time task, if it meets the scheduling condition, then the real-time task can be schedulable, otherwise it is not schedulable.

(3) To do static scheduling algorithm of schedulability analysis to the new arrival periodic real-time task, if it meets the scheduling condition, then the real-time task can be schedulable, otherwise it is not schedulable.

2.1. Task Model

In the environment with single processor, a collection $T = \{t_1, t_2, ..., t_n \mid (1 < i < n)\}$ including n real time tasks is defined,the rules are:

(1) In the collection, the time attribute of aperiodic real-time task can be expressed in a triad group $t_i = (A_i, C_i, D_i)$. Among them, A_i represents the arrival time of tasks, C_i represents the worst execution time of tasks, D_i represents the relative deadline of tasks, the specified tasks must be completed before absolute deadline $d_i = A_i + D_i$.

(2) The time attribute of periodic real-time task in the collection can be expressed with a triad $t_i = (A_i, C_i, P_i)$. Among them, P_i represents the task cycle.

(3) Each task is independent with others.

(4) The task has preemption; the tasks with higher priority can seize the CPU control of lower priority task.

2.2. Related Definition

Definition 1. Processor utilization: the proportion between processor time and the cost of real-time tasks in the execution, that is, the proportion between the worst execution time and

activity time. The CPU processor utilization of real-time tasks is C_i / D_i (t_i represents the aperiodic tasks) or C_i / P_i (t_i is the periodic task).

The processor utilization of real-time task set is sum of the processor utilization of all real-time tasks. The processor utilization of implement n real-time tasks is shown in expressions (1) and (2):

$$U = \sum_{i=1}^{n} \frac{C_i}{D_i} \text{ (U is an aperiodic task)} \quad (1)$$

$$U = \sum_{i=1}^{n} \frac{C_i}{D_i} \text{ (U is a periodic task)} \quad (2)$$

Definition 2. Laxity Time: laxity time L_i is task absolute deadline minus the rest of task execution time E_i, and minus the current time t, such as expression (3):

$$L_i = d_i - E_i - t \quad (3)$$

Definition 3. Worst Case Execution Time: the worst execution time C_i is refers to the real-time tasks during the implementation of a maximum of the Time, including scheduling task of spending Time.

Define 4. Important degree (Importance): it is a positive integer, indicating that this task compared to another Importance.

3. Analysis of Commonly Used Scheduling Algorithm

(1) Rate Monotonic (RM) scheduling algorithm [8], it is a kind of typical static priority scheduling algorithm, according to the length of task cycle to determine scheduling priority, the smaller task cycles is corresponding to higher priority. Advantages: simple realization mechanism and low scheduling. Disadvantages: only task cycle is taken as a priority decision condition, it is easy to ignore those important longer cycle task; CPU utilization rate is low; resources utilization is insufficient, the processing capability for emergency real-time is not strong.

(2) The Earliest Deadline First (EDF) scheduling algorithm [9] is a kind of typical preemptive dynamic priority scheduling algorithm, according to the deadline of each task to assign priority, the task with the earliest deadline has the highest priority. The advantage is: resource allocation is flexible to guarantee the task with earliest deadline to implement with priority and CPU utilization is high. Disadvantages: system overhead is big, which can't guarantee the execution of inaccessible deadline task; in overload situation there may happen domino phenomena make real-time performance fall sharply.

(3) Least Laxity First (LLF) scheduling algorithm [10], it is also a kind of commonly used dynamic priority scheduling algorithm, it is the improvement to EDF algorithm, according to the non-descending order of task's leisure time to distribute priority, the tasks with shorter leisure time has higher priority. The provision is laxity time of running tasks remains unchanged, and laxity time of tasks in ready queue decreases with time; the priority is also dynamically changed. The advantages are: saving CPU resources and ensuring that emergency task implementation in priority. Disadvantages: the priority waiting for tasks may become higher with the decrease of free time, preemption the currently running tasks, may result in frequent switching phenomenon between tasks (i.e. bump), which makes the system performance dropped substantially.

4. Analysis of Improved Scheduling Algorithm

4.1. Dynamic Scheduling Algorithm

According to the analysis of above general algorithms, to the aperiodic real-time tasks, a kind of improved Deadline and Laxity (DAL) first scheduling algorithm, it considers the influence of the deadline and idle time on priority, and increases the preempting threshold scheduling algorithm to reduce bump.

4.1.1. Analysis of DAL Scheduling Algorithm

The algorithm combines the advantages of EDF algorithm and LLF algorithm, as well as the priority size of P_i, according to the expression (4) the dynamic changes are taken place:

$$P_i = \alpha * d_i + (1-\alpha) * L_i \tag{4}$$

In the expression, α is the balance factor of real-time tasks, the size decides the influence degree of the priority by the deadline and idle time. When α is 0 it is tending to EDF algorithm, if it is 1, and tending to LLF algorithm.

The idea of the preempting threshold scheduling algorithm is [11]: using the priority (P_i, P_{ri}) to control the implementation of tasks, hereinto P_i is the basic priority, $P_{ri}(P_i < P_{ri} < n)$ is the preemption threshold of tasks, which permits the fulfillment of preemptive task priority and basic priority at the same time, is greater than the basic task priority and preemption threshold with the task preempted. This algorithm absorbs their respective advantages of preemption and non-preemptive scheduling algorithm, reduces the number of preemption, and reduces the system overhead by the jolt. The complete algorithm process of describing task scheduling is shown in Figure 2.

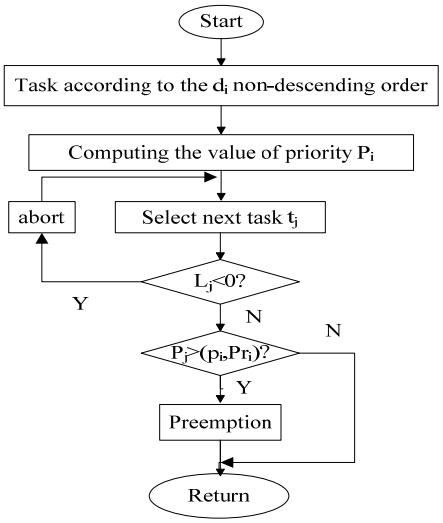

Figure 2. Task Scheduling Flow Chart of Algorithm

4.1.2. Analysis of Algorithm Schedulability

For an independent a periodic real-time task set, the minimum unbound of processor utilization ratio is 1. Therefore, as long as the processor utilization of a real-time task set $U<1$, then they can be scheduled. When a new aperiodic real-time task $t_i = (A_i, C_i, D_i)$ arrives, if it meets expression (5):

$$\frac{C_i}{D_i} + U_n \le 1 \tag{5}$$

So it can be scheduled, among them, U_n is processor utilization rate of current real-time tasks set.

4.2. Static Scheduling Algorithm

According to the analysis of above general algorithms, a kind of improved Rate And Importance (RAI) scheduling algorithm is used to solve periodic real-time task, considers the proportion of cycle and important degree of two factors, improves implementation probability rate of important tasks.

4.2.1. Analysis of RAI Scheduling Algorithm

In RAI algorithm, the priority of a task is appointed before operation, which is decided by period and important degree, where the importance degree is set by the user according to demand. Regulation is: the task with short cycle has higher priority; the task with higher important degree. This algorithm can be divided into the following several ways:

(1) Real-time tasks with same cycle and different important degree, according to the important degree execute in descending order.

(2) Real-time tasks with same important degree and different cycle, accordance with the RM scheduling algorithm, the tasks with shorter cycle implement in priority.

(3) Real-time tasks with same important degree and cycle, according to the way of FIFO to do scheduling.

(4) Real-time tasks with different important degree and cycle, regarding the important degree as preemptive conditions, to warranty priority implementation of important tasks.

4.2.2. Analysis of Algorithm Schedulability

For n independent periodic real-time tasks set with each other, the minimum up bound of the processor utilization ratio is $n*(2^{1/n}-1)$. Therefore, as long as the processor utilization U of real-time task set is not more than the minimum up bound, then they can be scheduling. When a new periodic real-time task $t_i(A_i, C_i, P_i)$ arrived, if it meets the formula (6):

$$\frac{C_i}{P_i} + U_n \le n*(2^{1/n}-1) \quad (6)$$

Then it is scheduled, and hereinto, U_n is the processor amortization of current real-time set.

5. Test And Analysis of the Performance

In order to test the real-time performance of this algorithm, in the experiment platform with AMD Sempron 2.01GHz single processor and 1.75GB memory, to hybrid real-time tasks set including 100 tasks, respectively adopts this algorithm, EDF scheduling algorithm, and RM scheduling algorithm, the test is executed with the performance evaluation indicators of schedule delays and deadline miss rate. In the test, the submission order of task is same, and the difference of task submission time is 1s.

5.1. Scheduling Delay

Scheduling Latency (SL) represents the time spent for real-time tasks from the happening of scheduling chance to be scheduling; it is an important indicator to measure the real-time performance. While the scheduling delay is lower, the better is the performance of real time.

From Table 1, the algorithm in this paper is less than EDF scheduling algorithm no matter the largest, the smallest, and the average scheduling algorithm, but they are slightly above the RM scheduling algorithm. It indicates the scheduling delay performance of algorithm in this paper is between EDF scheduling algorithm and RM scheduling algorithm, scheduling cost is slightly higher than RM scheduling algorithm, and far lower than EDF scheduling algorithm.

Table 1. Comparisons of Test Result About Scheduling Delay

Method	Scheduling Delay / μ_s Maximum	Scheduling Delay / μ_s Maximize	Scheduling Delay / μ_s Average
EDF scheduling algorithm	6	19	13.92
Classification algorithm	4	15	11.31
RM scheduling algorithm	3	14	10.68

5.2. Deadline Miss Rate

Deadline Miss Rate (DMR) represents the ratio between quantity of real-time tasks that does not fulfill deadline in the system in a certain period of time and total number of real-time tasks; it is another important indicator to measure the real-time performance. Deadline miss rate is lower, the real-time performance is better. Figure 3 presents the condition that deadline miss rate varies with the increase quantity of real-time tasks, where three kinds of algorithms are in the same condition, the transverse axis presents real-time task quantity, the vertical axis presents deadline miss rate. As can be seen from the graph, the deadline miss rate of the algorithm in this paper varies with the increase of load, the growth rate is faster than EDF scheduling algorithm, and far slower than RM scheduling algorithm. The performance of deadline miss rate of the algorithm in this paper is between EDF scheduling algorithm and RM scheduling algorithm.

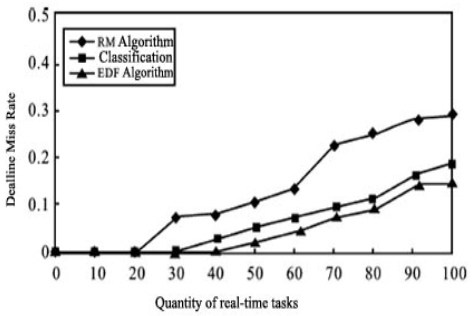

Figure 3. Comparison of Test Results of Deadline Miss Rate

6. Concluding Remarks

This paper puts forward a kind of classification scheduling algorithm, using two kinds of improved scheduling algorithm, and make a regulation that different types of real-time tasks can flexibly choose suitable scheduling algorithm for scheduling. In addition, schedulability analysis in the algorithm is simple; schedulability analysis of real-time task of using any scheduling algorithm is independent with real-time tasks of another kind of scheduling algorithm. Scheduling delay and deadline miss rate as performance indicators, to hybrid real-time tasks set including 100 tasks, the algorithm in this paper makes a comparison and EDF and RM scheduling algorithm. The experimental results show that the algorithm in this paper has their respective advantages in EDF and RM scheduling algorithm, integrated real-time performance has a great deal of improvement.

References

[1] Hong Jin, Hongan Wang, Qiang Wang. A real-time scheduling algorithm based on priority table and its implementation. *Journal of Software*. 2004;15(3): 360-370.
[2] Shenghui Liu, Song Ma. Research and application of real-time scheduling mechanism based on Linux kernel. *Computer engineering and application*. 2008; 44(6): 121-123.
[3] Chun-Hsiung Lan, Hai-Ming Chen, Mei-Hsiu Chen, Chih-Wei Chiu, Cheng-Jui Tsai. Green Production Scheduling/Planning with Parallel-Machine Layout for Multiple Orders and Diversified Due Dates. *JCIT*. 2012; 7(7): 43-50.

[4] Weixiao, Zhibao Feng. Research and implementation of advanced EDF scheduling algorithm. *Computer engineering*. 2009; 35(18): 231-233.
[5] Liu CL, Layland J. Scheduling algorithms for multi programming in a hard real-time environment. *Journal of ACM*. 2003, 20(1) : 46-61.
[6] Sun Yuan, Zhao Xiaobing, Yang Guosheng. An Optimal Reservation-based Feedback Scheduling Algorithm for Vehicular Application Specific Operating Systems. *IJACT*. 2012; 4(1): 359-377.
[7] Jiwen Dong, Yang Zhang. Improvement and application of embedded real-time operating system scheduling algorithm. *Computer applications*. 2009; 29(9): 2516-2519.
[8] Yunfu Tan, Jie Liu, Guohua Liu. A kind of improved real-time scheduling algorithm in Linux and its application. *Computer science*. 2008; 35(10): 256-258.
[9] Hong Jin, Hongan Wang. An integrated design method of task priority. *Journal of Software*. 2003; 14(3): 376- 382.
[10] Dana M, Pascale M, Laurent G. Analysis of deadline assignment methods in distributed real-time systems. *Computer Communication*. 2004; 27(15): 1412-1423.
[11] Terrasa A, Garcia Fomes A, Botti VJ. Flexible real-time Linux: a flexible hard real-time environment. *Real-Time Systems.* 2004; 22(2): 151-173.