WUJNS
Wuhan University Journal of Natural Sciences

Vol.12 No.1 2007 113-116

Article ID: 1007-1202(2007)01-0113-04
DOI 10.1007/s11859-006-0231-x

# An Efficient Real-Time Fault-Tolerant Scheduling Algorithm Based on Multiprocessor Systems

☐ **YANG Fumin, LUO Wei, PANG Liping**

College of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, Hubei, China

**Abstract:** In the context of real-time fault-tolerant scheduling in multiprocessor systems, Primary-backup scheme plays an important role. A backup copy is always preferred to be executed as passive backup copy whenever possible because it can take the advantages of backup copy de-allocation technique and overloading technique to improve schedulability. In this paper, we propose a novel efficient fault-tolerant rate-monotonic best-fit algorithm efficient fault-tolerant rate-monotonic best-fit (ERMBF) based on multiprocessors systems to enhance the schedulability. Unlike existing scheduling algorithms that start scheduling tasks with only one processor. ERMBF pre-allocates a certain amount of processors before starting scheduling tasks, which enlarge the searching spaces for tasks. Besides, when a new processor is allocated, we reassign the task copies that have already been assigned to the existing processors in order to find a superior tasks assignment configuration. These two strategies are all aiming at making as many backup copies as possible to be executed as passive status. As a result, ERMBF can use fewer processors to schedule a set of tasks without losing real-time and fault-tolerant capabilities of the system. Simulation results reveal that ERMBF significantly improves the schedulability over existing, comparable algorithms in literature.

**Key words:** real-time periodic tasks; fault-tolerance; primary/backup copy; multiprocessor systems
**CLC number:** TP 316

**Received date:** 2006-05-10
**Foundation item:** Supported by the National Basic Reseach Program of China (973 Program 2004 CB318200)
**Biography:** YANG Fumin (1966-), male, Professor, research direction: real-time systems, embedded systems. E-mail: yangfm@routon.com

## 0 Introduction

Multiprocessor systems are increasingly being used for real-time applications because of their capability for high performance, reliability and extensibility. Due to the critical nature of tasks in real-time systems, it is essential that every task admitted in the system complete its execution even in the presence of faults[1]. As such, fault-tolerance is an inherent requirement for such systems[2,3].

In the context of real-time fault-tolerant scheduling in multiprocessor systems, Primary- backup scheme plays an important role. In this approach, two versions of one task are scheduled on two different processors and an acceptance test is used to check the correctness of the schedule[4-7]. The two variants of this scheme include active backup copy[4,5], and passive backup copy[6,7] Generally speaking, backup copy is always preferred to be executed as passive backup copy, because it can take the advantages of backup copy overloading and backup copy de-allocation technique to improve schedulability.

Both active backup copy and passive backup copy are incorporated into the well-known Rate-Monotonic First-Fit assignment algorithm (FTRMFF) to achieve fault-tolerance[8]. FTRMFF takes a first-fit strategy to assign tasks to processors, that is assigning task copy (primary or backup) to the first processor that could be fitted in. The intuition behind this

strategy is that we should always trying to use the existing processors to scheduling tasks. Yang C H proposed a best-fit algorithm based on a real-time fault-tolerant rate-monotonic algorithm, FTRMBF, which search all the existing processors trying to make as many backup copy to be executed as passive copy as possible to enhance system schedulability[9]. Although FTRMBF improve the system schedulability over FTRMFF to some extends, both FTRMBF and FTRMFF start searching processor with only one initial processor. This limits the searching space, particular for the tasks firstly to be assigned.

In this study, we propose a novel Efficient Fault-tolerant Rate-Monotonic Best-fit algorithm (ERMBF) based on multiprocessors systems to enhance the schedulability. First, unlike existing scheduling algorithms(FTRMBF or FTRMFF) that start scheduling tasks with only one processor. ERMBF pre-allocates a certain amount of processors before starting scheduling tasks, which enlarge the searching spaces for tasks. Moreover, when a new processor is allocated, we reassign the task copies that have already been assigned to existing processors. These two strategies are all aiming at making more backup copies to be executed as passive backup copy as possible. As a result, ERMBF can use fewer processors to schedule a set of tasks without losing real-time and fault-tolerant capabilities of the system[10].

# 1 System Model

Our paper considers a typical multiprocessor systems consisting of a set of tasks and a set of processors. Our distributed systems are characterized as follows.

- A set of processors

$\Omega = \{p_1, p_2, \cdots, p_M\}$

Here, $\Omega$ is the processor set, $p_i$ is the $i$-th processor and $M$ is the total number of processor. In this model, all processors are assumed to be identical to assure same execution time for each task on different processors. It is also supposed that enough processors are provided.

- A set of primary copy of real-time tasks

$\Gamma = \{\tau_1, \tau_2, \tau_3, \cdots, \tau_N\}$

$\tau_i = (C_i, T_i), \quad (i = 1,2,\cdots,N)$

Here, $\Gamma$ is the set of tasks, $\tau_i$ is the $i$-th task, and $N$ is the number of tasks which are periodic, independent and preemptive. $C_i$ denotes the execution time of $\tau_i$. $T_i$ denotes the period of task $\tau_i$.

- A set of backup copy of real-time tasks

$B_\Gamma = \{\beta_1, \beta_2, \beta_3, \cdots, \beta_N\}$

$\beta_i = (D_i, T_i), \quad i = 1,2,\cdots,N$

Here, $B_\Gamma$ is the set of backup copy of real-time tasks $\Gamma$. $\beta_i$ is the corresponding backup copy of $\tau_i$. Hence, $D_i$ is the execution time of $\beta_i$. In our model, execution time of backup copy and primary copy are assumed to be equal, that is: $D_i = C_i$. Correspondently, $T_i$ denotes the period of $\beta_i$.

In our system model, as in Ref.[8], the backup copy has two statuses: passive backup copy and active backup copy. When we assign a task, we assign its primary copy before assigning the backup copy. The status of backup copy is determined by the following:

$$\text{Status}(\beta_i) = \begin{cases} \text{passive}, & T_i - R_{ij} > D_{ij} \\ \text{active}, & T_i - R_{ij} \leqslant D_{ij} \end{cases} \quad (1)$$

where $P(\tau_i)=P_j$ and $P(\beta_i)=P_k$.

Here, $R_{ij}$ denotes the worst case response time (WCRT) of $\tau_i$ which is assigned to $P_j$[10]. For convenience, $\gamma_i$ represents a primary copy or a backup copy, namely, $\gamma_i = \tau_i$ or $\beta_i$.

To concentrate on our concerned problems, we make the following assumptions about failure characteristic of the hardware:

① Hardware provides fault isolation mechanism, that is a faulty processor can not cause incorrect behaviors in a non-faulty processors;

② Processors fail in a fail-stop manner, which means a processor is either operational or cease functioning;

③ The failure of a processor is detected by the remaining ones within the closest completion time of a task scheduled on the faulty processor.

# 2 Processor Pre-Allocation Strategy

## 2.1 Problems of Existing Scheduling Algorithm

To show the drawbacks of existing scheduling algorithms and induce our scheduling algorithm, we use a algorithm FTRMBF proposed by Yang C H to schedule a set of tasks which is also scheduled by ERMBF.

Considering a task set $\sigma$ consisting of five tasks, namely, $\tau_1=\beta_1=\{2,5\}$, $\tau_2=\beta_2=\{2,5\}$, $\tau_3=\beta_3=\{5,10\}$, $\tau_4=\beta_4=\{14,15\}$, $\tau_5=\beta_5=\{14, 16\}$. If $\sigma$ is scheduled by FTRMBF, 7 processors are needed and the final assignment result is listed in Table 1.

It is worth noting that $\beta_2$ can be executed as a passive backup copy because $C_2<T_2$. But in the FTRMBF, only one processor is allocated when starting assigning

task copies. So has $\tau_2$ has to be allocated to $P_1$ and Status($\beta_2$)=active.

Our processor pre-allocated strategy, we pre-allocate several processors when starting assigning tasks. For example, if we allocate 3 processors when starting assigning task copies. $\tau_2$ can be assigned to $P_3$ and Status($\beta_2$)= passive. The remaining task copies are also assigned as FTRMBF algorithm, the final assignment results is listed in Table 2, and only six processors are required.

**Table 1 Tasks assignment results by FTRBF on $\sigma$**

| Task | $P(\tau_i)$ | $P(\beta_i)$ | Status($\beta_i$) |
|---|---|---|---|
| Task1 | $P_1$ | $P_2$ | passive |
| Task2 | $P_1$ | $P_3$ | active |
| Task3 | $P_3$ | $P_2$ | active |
| Task4 | $P_4$ | $P_5$ | active |
| Task5 | $P_6$ | $P_7$ | active |

**Table 2 Tasks assignment results by ERMBF on $\sigma$**

| Task | $P(\tau_i)$ | $P(\beta_i)$ | Status($\beta_i$) |
|---|---|---|---|
| Task1 | $P_1$ | $P_2$ | passive |
| Task2 | $P_3$ | $P_2$ | passive |
| Task3 | $P_1$ | $P_3$ | active |
| Task4 | $P_4$ | $P_2$ | active |
| Task5 | $P_5$ | $P_6$ | active |

### 2.2 Pre-Allocation Processor Number

In our pre-allocation strategy, the pre-allocation number PreNumber can be calculated by the following equation

$$\text{PreNumber} = \left\lfloor \sum_{\tau_i \in \Gamma} C_i / T_i \right\rfloor \quad (2)$$

The calculation of PreNumber is based on the following philosophy: Pre-allocation processor number is the number of processors needed by all primary copies, it is assumed that Rate-Monotonic scheduling algorithm can take full advantages of the processors time. So PreNumber is the least amount of processors needed by primary copies.

Although the calculation of PreNumber is some what conservative because RMS can not take full advantages of processors and it ignore active backup copies, it is enough and effective for the initial processor pre-allocation.

## 3 Task Copies Reallocation Strategy

Both FTRMBF and FTRMFF will allocate one more processor when current processors can not accommodate the existing tasks copies. The philosophy behind this method is that FTRMBF and FTRMFF always try to utilize the existing processors as long as possible. However, the drawback of this method is that each task copy assignment is based on the existing resources, namely, number of processors. When a new processor is added, this means that more resources are available. But the task copies previously assigned can not take advantages of the additional processors.

To solve this problem, another strategy that we will take is the task copies reallocation strategy. That is, we will reassign the task copies that have already assigned to existing processors when a new processor is allocated. The intuition behind the strategy is that when we assigned a task copy, it's decision is always based on the existing number of processors. However, when a new processor is added, it means that we can have more resources than previous times. At this moment, reorganizing existing allocation situation can have better performances.

## 4 Performance Evaluation

The metric in our experiments is the number of processors required to assign a set of tasks. In the out come of the experiments, we denote with $N$ the number of the processors used by ERMBF and $M$ the number of processors required by FTRMBF. The outcome of the experiments is given in Fig.1, which shows that both $M$ and $N$ increase proportionally with $L$. This is because as $L$ increases, more processors are needed to schedule the set of tasks. Most importantly, it is clear that ERMBF can significantly save processors with respect to FTRMBF.

More over, Figure 2 shows the values of $(N-M)/M$ for the same experiments, which give the ratio of saving processors introduced by ERMBF with respect to FTRMBF. It is observed that when $\alpha = 0.5$. The advantage of ERMBF over FTRMBF is mostly pronounced. This due to the fact that when $\alpha = 0.5$ most backup copy have the potentiality to be executed as passive backup copy. The ERMBF has more ability to exploit such potentiality compared with FTRMBF. In summary, the ERMBF can saves 18.2%, 9.1% and 3.7% respect to

FTRMBF for $\alpha$=0.5,0.2, and 0.8 respectively.

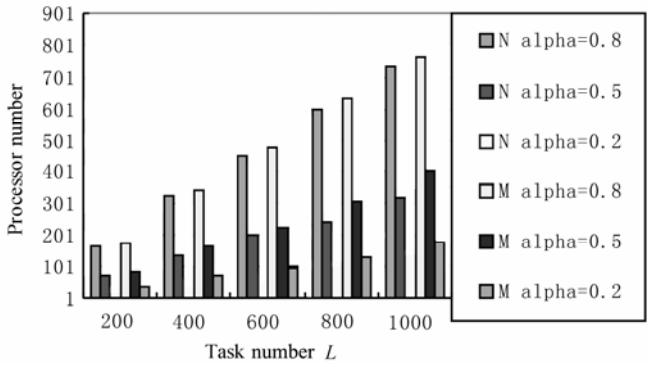


**Fig.1 Comparison of the required processor number between ERMBF and FTRMBF**

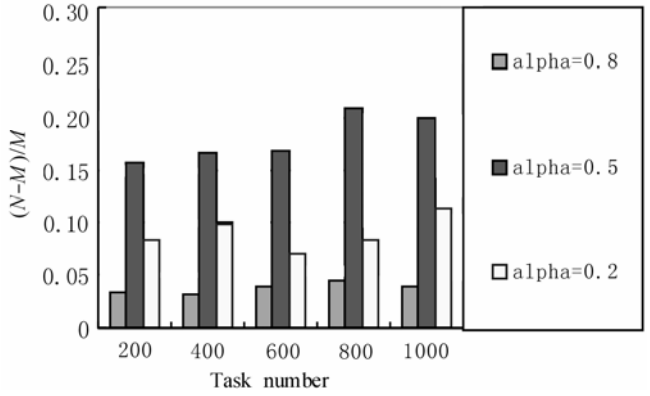


**Fig.2 Ratios $(N-M)/M$ show the effectiveness of ERMBF over FTRMBF**

## 5 Conclusion

The present paper analyzes the drawbacks of traditional First-Fit and Best-Fit algorithm, and proposed a novel real-time fault-tolerant Rate-Monotonic Best-Fit algorithm ERMBF. ERMBF pre-allocates a certain a mounts of processors before tasks assignment, which enlarge searching spaces for task copies, thus making as many backup copes to be executed as passive backup copies as possible. Moreover, we take a task copies reallocation strategy to obtain better assignment configuration when a new processor is added. Consequently, fewer processors are needed and the schedulability of the system is enhanced. Experiments results show that ERMBF significantly out perform existing comparable algorithms in terms of schedulability, measured as number or processor needed to schedule a set of primary copies and its corresponding backup copies.

Future studies in this area are two folders. First, we are going to develop more efficient scheduling algorithms aiming at boosting system's schedulability. Second, we intend to investigate a more complex version of scheduling algorithm, in which the precedence constrains among tasks and aperiodic tasks scheduling are incorporated.

## References


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

[2] Dhall S K, Liu C L. On a Real-Time Scheduling Problem[J].*Operations Research*, 1978, **26**(1):127-140.

[3] Srinivasan S, Jha N K. Safety and Reliability Driven Task Allocation in Distributed Systems[J]. *IEEE Trans Parallel and Distributed Systems*, 1999, **10**(3): 238-251.

[4] Yang C H, Deconinck G, Gui W. Fault-Tolerant Scheduling for Real-Time Embedded Control Systems[J]. *Journal of Computer Science and Technology*, 2004, **19**(2):191-202.

[5] Wang F, Ramamritham K, Stankovic J. Determining Redundancy Levels for Fault-Tolerant Real-Time Systems[J]. *IEEE Transactions on Computers*, 1995,**44**(2): 292-301.

[6] Liu H, Fei S M. A Fault-Tolerant Scheduling Algorithm Based on EDF for Distributed Control Systems[J]. *Chinese Journal of Computers*, 2003, **14**(8): 1371-1378 (Ch).

[7] Luo W, Yang F M, Pang L P, *et al*. Fault-Tolerant Scheduling Based on Periodic Tasks for Heterogeneous Systems[C]// *Proc* 3*rd Int Conf Autonomic and Trusted Computing*. Wuhan: Springer, 2006: 571-580.

[8] Bertossi A A, Mancini L V, Rossini F. Fault-Tolerant Rate Monotonic First-Fit Scheduling in Hard-Real-Time Systems[J]. *IEEE Trans Parallel and Distributed Systems*, 1999,**10**(9): 934-945.

[9] Yang C H, Ji L, Sheng D Y, *et al*. Fault-Tolerant Scheduling Based on the BEST-FIT Heuristics for Real Time Multiprocessor Systems[J]. *Computer Engineer and Science*, 2003, **25**(5):61-64(Ch).

[10] Pandya P. Finding Response Times in a Real-Time System[J]. *The Computer J*, 1986, **29**(1): 390-395.


□