STATIC REAL-TIME SCHEDULING ALGORITHM WITH FAULT-TOLERANCE IN MULTI-PROCESSOR SYSTEMS

Qin Xiao, Han Zongfen, Pang Liping and Li Shengli
Department of Computer Science, Huazhong University of Science and Technology
Wuhan, China, Post Code 430074

ABSTRACT

This paper proposes a static fault-tolerant schedu-ling algorithm for real-time tasks. This scheduling algorithm can schedule the tasks with the fault-tolerant requirements together with those without fault-tolerant requirements. This algorithm can generate schedules that tolerate one processor failure for all tasks with fault-tolerant requirements.

INTRODUCTION

New generation real-time systems must support real-time applications by maintaining an environment that satisfies real-time, predictability and reliability requirements[1]. Many real-time systems have the requirements of high reliability, the real-time tasks must finish with the specified deadlines in spite of hardware and software failure[2]. In the research of new generation real-time systems, scheduling algorithms with fault-tolerance is one of the key points.

BASIC IDEA OF THE ALGORITHM

Our algorithm can schedules a set of real-time tasks, which contains two kinds of real-time tasks, one has fault-tolerant requirement(FTR) and another has no such requirement. Task redundancy scheme studied here is the primary-backup copy approach. Each task with FTR has a primary copy and a backup copy, and every task without FTR has only one copy. When the primary copy fails, the backup copy will perform in series. The goal of the scheduling algorithm is to minimize the schedule length on each processor such that all tasks can be their deadlines. This algorithm starts by sorting the tasks with FTR, and use Largest Processing Time first(LPT) algorithm to schedule the primary copies of these tasks, then it applies Adapted Largest Processing Time(ALPT) algorithm to schedule the backup copies. At last, the tasks without FTR are sorted and scheduled by involving LPT algorithm.

RESULTS AND DISCUSSION

The simulation experiments are carried out by the steps as follows: first, a deadline DL is selected, then a range of values CR for the computation time of tasks is chosen. After generating a set of real-time tasks, simulation program will find the minimum number of processors on which all tasks can be scheduled to run and finished before the timing constrain.

First, we try to find out the relationship between the number of tasks and the minimum number of processors (MNP). Experiment shows that MNP increases with the rise of the number of tasks. Because the more tasks in the system, the more processors need to complete the tasks before the timing constraint.

Second, the relationship between CR and MNP is explored. Simulation result The smaller the value of CR is, the fewer processors are needed to schedule the tasks. It is because, when a common deadline is chosen, the computation time of the tasks generated by a smaller CR are possibly shorter than those which generated by a higher CR, and the tasks with shorter computation time need fewer processors to complete before the given deadline.

CONCLUSION

The assumption of this algorithm is strict, for all the tasks have a common deadline, and all the tasks are periodic. We are currently studying the algorithms that schedule tasks with different deadlines, and can also schedule the periodic tasks with aperiodic ones.

REFERENCES

[1] K.G. Shin, G.Koob, and F.Jahanian. Fault-tolerant in real-time systems. IEEE Real-time Systems Newsletter, 1991, 7(3) : 28-34.
[2] S. Han and K.G. Shin. A primary-backup channel approach to dependable real-time communication in multiple networks. IEEE Transactions on Computers, 1998, 47(1) : 46-61.