

On Scheduling Tasks in Reliable Real-Time Control Systems

Ramesh Chandra and Lui Sha
rameshch@uiuc.edu, lrs@cs.uiuc.edu

1 Introduction

Feedback control is one of the most common applications of real time systems. However, the design of controller frequencies, task scheduling and reliability engineering are often done separately, resulting in sub-optimal results. This short paper provides an overview of an integrated approach to reliable real-time controller design by optimizing the system control performance subject to schedulability and software reliability constraints.

In control applications, the software reliability challenges goes beyond the specification, design, development and verification considerations. Advanced control techniques such as neural-net learns by example and can perform sophisticated non-linear controls. Indeed, the properties of certain advanced controllers can be difficult to analyze and verify.

This problem can be addressed by using analytically redundant controllers, where the sophisticated but less reliable controller is "supervised" by a simple and reliable controller. An example of using analytically redundant controllers to enhance system reliability is the Boeing 777 flight control, where the normal controller is the new 777 controller whereas the secondary controller is based on the well understood 747 control technology. The aircraft's state under the normal controller should be within the stability envelope of the 747 controller [6].

2 Fault Tolerant Task Scheduling

Seto et. al. [1] showed that in digitized continuous controller, the relationship between control performance (loss) and sampling frequencies can be modeled as an exponential function of sampling frequencies in the form of

$$\Delta J(f) = \alpha e^{-\beta t} \quad (1)$$

where $\Delta J(f)$ is the performance (loss) index due to discrete sampling, α is the magnitude coefficient and β is the decay rate. We now study the problem of finding the task periods such that the expected system performance is optimized.

Given a set of n periodic real-time task pairs T_1, T_2, \dots, T_n , each task pair T_i has a high performance specification and implementation T_{iP} , and a high reliability (assurance) specification and implementation T_{iR} . C_{iP} and C_{iR} represent the task execution times of T_{iP} and T_{iR} respectively. The performance indices of T_{iP} and T_{iR} are denoted by $\Delta J_{iP}(f)$ and $\Delta J_{iR}(f)$ respectively. f_{iP_m} and f_{iR_m} denote the minimum frequencies, while λ_{iP} and λ_{iR} denote the failure rates of T_{iP} and T_{iR} respectively. μ_{iP} denotes the repair rate of T_{iP} .

We assume that if T_{iR} fails, then the task T_i fails and that there is no repair possible in that case. D denotes the mission duration for all the tasks. The performance loss on failure of task T_i is denoted by L_i . Also, the relative weight of task pair T_i in the system is represented by w_i .

We assume that the high assurance controllers are properly verified. That is, their failure rate is negligible with respect to the application reliability requirement. We also assume that the controlled task would fail if the high assurance controller fails. There will be no restart of the high assurance controller, should it fail. We also assume that the high performance controller has been reasonably tested. That is, although the high performance controller does not meet the reliability requirement, it is not totally useless. That is, it will fail only occasionally. When the high performance controller fails, the system is under the control of the high assurance controller and the high performance controller will be restarted. The system will pass the control back to the high performance controller once it is restarted. We assume that the high performance controller's restart time is much shorter than its mean time to failure.

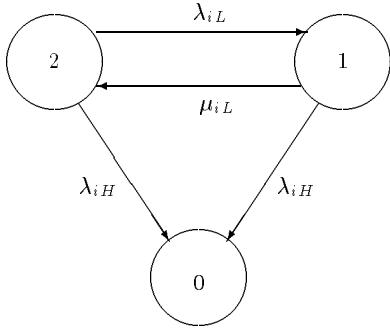


Figure 1: The Reliability Markov Model for Task T_i

To maximize the expected control performance of the system, we first compute the availability each of the control tasks for a mission duration D .

We assume that the failure rates and repair rates of tasks in the system are exponential. With this assumption, for each task T_i , we construct the Markov model representing the states of the task, as shown in Figure 1. The state 2 of the Markov model represents the state when both the implementations T_{iP} and T_{iR} of the task T_i are working. The state 1 represents the state when the high performance implementation T_{iP} has failed but the high reliability implementation T_{iR} is still working. The system would still be working fine in this state, albeit, with a loss in performance. Repair of T_{iP} in state 1 would move the state of the task back to state 2. The state 0 represents the state when the high reliability implementation has failed. In this case there is no provision for repair. The system initially starts in the state 2.

The above Markov model is solved for the transition matrix $P(t) = [p_{ij}(t)]$ at time t . In particular, since state 2 is the initial state, we are interested in the probabilities $p_{22}(t)$, $p_{21}(t)$ and $p_{20}(t)$, which represent the probability of the system being in state 2, state 1 or state 0 respectively after time t from the system start, with initial state being state 2. Since the mission duration of the system, D is known, the fraction of the mission duration spent in each of the states, A_{i2} , A_{i1} , and A_{i0} can be found using the matrix exponential function in *Mathematica (MatrixExp)* or in *Matlab (expm)*.

After the fraction of mission time spent in the different state is known, the resulting $2n$ task set can be transformed into a representation where the optimal task frequencies can be found using the methods in [1].

It is well know that it is difficult to acculately estimate the failure rate of software. Fortunately, we are

able to show that the optimal frequencies that maximizes the control performance are insensitive to errors in failure rates provided that $MTTF \gg MTTR$.

3 Conclusions

In this short paper, we outlined an approach that extends the work done in [1] to scheduling tasks with different reliabilities so that the expected control performance is maximized. Future work will include extending this work for fixed priority scheduling (RMA).

Acknowledgements

The authors would like to thank the sponsorship from the Office of Naval Research and the discussions with Danbing Seto.

References

- [1] Seto, D., Lehoczky, J.P., Sha, L., and Shin, K.G., "On Task Schedulability in Real-Time Control Systems", *Proceedings of 17th Real-Time Systems Symposium*, pp. 13-21, December, 1996.
- [2] Liberato, F., Lauzac, S., Melham, R., and Mossé, D., "Fault Tolerant Real-Time Global Scheduling on Multiprocessors", *Proceedings of Euromicro Workshop on Real-Time Systems*, 1999.
- [3] Bertossi, A., Fusiello, A., and Mancini, L., "Fault-Tolerant Deadline-Monotonic Algorithm for Scheduling Hard-Real-Time Tasks", *Proceedings of International Parallel Processing Symposium*, pp. 133-138, 1997.
- [4] Burns, A., Davis, R., and Punnekkat, S., "Feasibility Analysis of Fault-Tolerant Real-Time Task Sets", *Proceedings of Euromicro workshop on Real-Time Systems*, pp. 29-33, 1996.
- [5] Seto, D., Krogh, B., Sha, L., and Chutinan, A., "Dynamic Control Systems Upgrade Using Simplex Architecture", *IEEE Control*, August, 1998.
- [6] Yeh, Y. C. (Bob), "Dependability of the 777 Primary Flight Control System", the Proceedings of DCCA Conference, 1995.