INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH & DEVELOPMENT

ISSN 2278 – 0211 (Online)

A Combined Study of Control Scheduling and Online Adaptive System for Fault Tolerance in Multiprocessor Real Time Systems

Leena Das
Assistant Professor, KIIT University, India
Rohan Bibhudutta Routray
M. Tech Scholar, KIIT University, India

Abstract:
The recent emergence of multiple processors and related technologies in many commercial systems has increased the prevalence and the development of multiprocessor architecture. Contemporarily, real-time applications have become more complex and sophisticated in their behaviour and interaction. Inevitably, these complex real-time applications will be deployed upon these multiprocessor platforms and require different analysis techniques to verify their correctness and also their efficiency in terms of different metrics. During the past, the development of real time scheduling algorithm is enhancing and the use of Feedback Control Scheduling Algorithm (FCSA) in the control scheduling co-design of multiprocessor real - time system has also increased. FCSA provides Quality of Service (QoS) in terms of overall system performance and resource allocation in open and unpredictable environment. FCSA uses quality control feedback loop to keep CPU utilization under desired unitization bound by avoiding overloading and deadline miss ratio.

Key words*: Real time system, Fault, Feedback Control scheduling (FCS), online adaptive fault tolerant system*

1. Introduction

The Commercial usage of the computers dates back to a little more than 60 years back. This brief period can be divided into main frame computers, personal computers and post pc eras. The main frame computers were quite very expensive during that period and it served a large number of users which was its benefit. The pc era saw the emergence of the personal computers or the desktops which were quite affordable and also was meant for the individual users. The post PC era that is, in the present scenario is seeing the emergence of the portable and small systems like mobile phones, Tablets etc and these type of computers are embedded in every day applications which is making an individual interact with the several computers every day in a hustle free manner. Thus in the recent years the use of computers has changed drastically in form of usability ,durability and reliability , as compared to its former design ,modelling , scheduling etc . This Post PC era has seen the evolution of the real time system which can become the most reliable computer till date. The use of the real time systems is properly evident in the real life also it is used in the domestic purpose as well as in the industrial systems too. This inclement in the use of the real time systems has made the area of research open , and to find out all the merits and the demerits of the system to make the real time system more developed and easy to use. Since some years ago, use of Feedback Control Scheduling Algorithm (FCSA) in the control scheduling co-design of multiprocessor real time system has increased. Feedback scheduling has become an important methodology in dynamic co-design of control and scheduling for real time systems. In this paper i have discussed the two basic usually used fault tolerant techniques which are used in the multiprocessor real time systems in the recent past. Fault tolerance of programs running sequentially on uniprocessors is well understood and many efficient solutions exist for that purpose. On the other hand, programs running in parallel on shared memory multicore processors present a greater challenge due to shared memory accesses, which are a frequent source of non-determinism. This paper presents the work done on fault tolerance with primary focus on work for multiprocessor real time systems. Till date a lot of work has been carried out in the fault tolerance domain for real time systems to make the systems more reliable, but this paper focuses mainly on the recent phase of the work which is carried out in the real time multiprocessor systems which includes the feedback control scheduling and online adaptive fault tolerant system.

2. Basic Concepts

This section presents the basic concepts related to the field of fault tolerance for multiprocessor real time systems. The discussion is based merely on the way a system interacts with other systems in particular environment, when an external agent comes to make a change in the system.

A *system* is an entity that interacts with other systems which as a result provides an output which was expected by the user to provide [3] . Ex- A system can be hardware based i.e. processor or software based i.e.an application. A system is said to be in a *failed state* if the external state deviates from the desired state of the user. The cause of this failed state is a fault [3] . When a fault becomes active it can affect the total state of the system and hence giving an undesired state to the user.

A *deadline* is a timing milestone for a task in the real time systems which are required to be completed in the allotted time. If the deadline is infinity then the job has no deadline. The deadline is assigned to the system tasks such that the system produces the desired output. If a *hard deadline* is not met then it can result to a catastrophic failure in the real time system [2]. Ex-Anti missile system.

The *performance deadline* are more confined than the hard deadline, if the deadline is missed it does not sends the system to an unsafe state rather if the hard deadline is disrespected then it sends system to an unsafe state.

- Ex- Video conferencing.

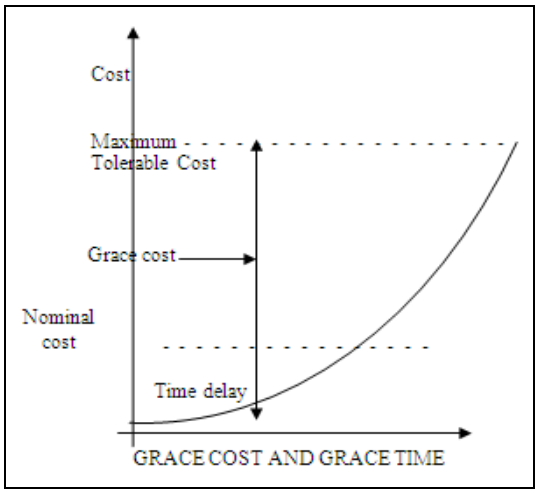

Figure 1 Deadlines Establishment Associated

In the above figure1 [2], the *Cost* of a system can be established as the timeliness with respect to the performance of the real time system. The performance and the hard deadline are separated by the grace time. The minimum cost that won't decrease the performance of the real time system is the *nominal cost* of the system. For the safety purpose the cost of the task should be made to the maximum beyond which the cost is intolerable and it is known as *maximum intolerable cost.* The difference between the maximum tolerable cost and the nominal cost is termed as *grace cost* [2].

Now coming to the notion of fault tolerant system , A fault tolerant system is a system which can maintain the ability of functioning in the presence of faults also [1] , and hence producing the results as the output as required by the system to produce.

The basic three notions of fault tolerance are

- *Fault* -As already referred previously a fault is a defect or failure which won't allow the system to produce the result as required by the system to produce. The fault is a defect which basically happens in the systems hardware or the software.
- *Failure* -Failure is a state of the system when the system is unable to produce the result as required by the system to produce and hence it's the departure of the system from the service which it needs to provide as an output.
- *Error* -In system oriented language an error can be told as the difference between the actual value and the value which is produced as a result. It is also an introduction of internal or external state of the system that may lead to failure [5].

Therefore there is a casual relationship between the notions of the fault tolerance and they are-

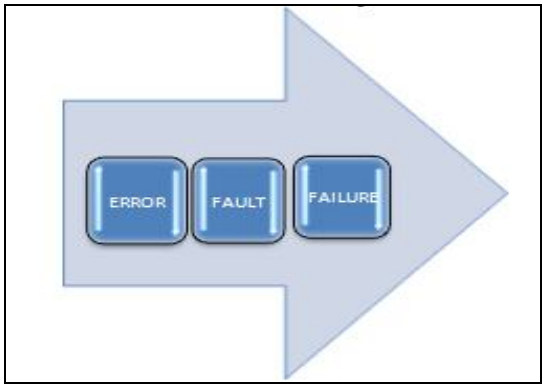

Figure 2: Casual relationship between the notions of the fault tolerance

Ex- Consider for an instance a system is running on multiprocessor architecture, a fault in one processor may cause it to crash, that could be known as failure, and can be seen as a fault in the system, and hence the ability of the system to be in a workable condition can be recorded as a fault tolerant system instead of failure tolerance.
The scheduling algorithms for multiprocessor real time systems are basically divided into two types and they are *dynamic* and *static* algorithms [1]. If the scheduling algorithm has the complete knowledge of the task sets, and its constraints are known as deadlines , completion time etc then the scheduling algorithm is known as *dynamic algorithm*. Ex- Rate monotonic algorithm, earliest deadline algorithm. As compared to the static the *dynamic algorithm* does not has the complete knowledge of the task set .Ex- spring scheduling algorithm.

3. Scheduling Algorithm for Multiprocessor Real Time System
The recent emergence of real time systems has made a boom in the present scenario of computing systems. Due to this the need of proper scheduling of tasks is required to provide the required output as needed by the user. The scheduling of real time system has seen a lot of development, in which the emergence of feedback control scheduling has been seen as a great procedure to provide the proper output. The other scheduling system named as online adaptive fault tolerant system is based on feedback control scheduling, which is implemented on multiprocessor real time systems to provide proper QOS.

3.1. Feedback Control Real Time Scheduling
The feedback control real time scheduling algorithm framework uses the control theory rather than the adhoc techniques. The previously used scheduling algorithm basically uses the heuristics approach to find out the most efficient way to solve the task sets and get the output.
Once the schedules are created that cannot be adjusted based on the continuous feedback which is required for the proper output .this is known as open loop algorithm. *Open loop algorithm* provides proper output under predictable environment [4]. But in the recent year's open loop algorithm are not viable for the unpredictable environment such as the soft real time applications such as e-business server, where neither the requests nor the parameters are known priori. Due to the above consequences the adaptive approach was used for this algorithm which provides a proper output for the real time system as required by the user. In t he past the deadline miss was avoided but in the recent trend towards real time system the effect of deadline is managed dynamically. In the past the adhoc techniques were used which were very laborious time consuming for designing the real time systems , but in the recent year feedback control scheduling establishes the system systematically and then designs adaptive real time system[4] .

3.1.1. Task Model
Each task (t_i) in a real time system environment has several quality of service levels where QoS levels is represented as n which is greater than equal to two [4] .
Each QoS level is characterized by

- $D_i[j]$: relative deadline.
- $EE_i[j]$: estimated execution time.
- $AE_i[j]$: actual execution time.
- $V_i[j]$: the value contributed by the task t_i if it is completed at QoS level j before its deadline $D_i[j]$.

Task model for periodic tasks- A periodic task is repeats after a certain period of time, it is also is referred as clock driven tasks [4].

- Pi(j) : invocation period.
- Bi[j] :estimated CPU utilization.
- Ai[j] : actual CPU utilization.

Task model for a periodic tasks - An a periodic tasks is many way similar to sporadic tasks, it recurs at random instants, but the minimum distance between the two tasks is zero.

- EIi[j]: estimated inter arrival time.

- AIi[j]: average inters arrival time between subsequent invocations.
- Bi[j]: estimated cpu utilization.
- Ai[j]: actual cpu utilization.

3.1.2. Control Related Variables

The *control related variables* are the performance metrics on which the task model depends .The control variables are controlled by the scheduler [4]. According to the *control theory* point of view the system output can be obtained according to the requirement by changing the input dynamically .The control related variables are as follows-

M(k)-*Miss ratio* - The deadline miss ratio is the ratio of number of deadline misses to that of total number of completed or the aborted tasks.

M(k) = number of deadline misses ÷ total number of completed/aborted tasks. (1)

U(k)-*CPU utilization*.- % of the CPU busy time. (2)

B(k)-*Total estimated utilization.* - This is the manipulated variables and are the attributes that can be dynamically changed by the scheduler to affect the value of the required o/p.

$B(k) = \sum_{i=0}^{n} Ui[li(k)]$ (3)

where ti is the tasks with a QOS level of li(k)in the k^{th} sampling window.

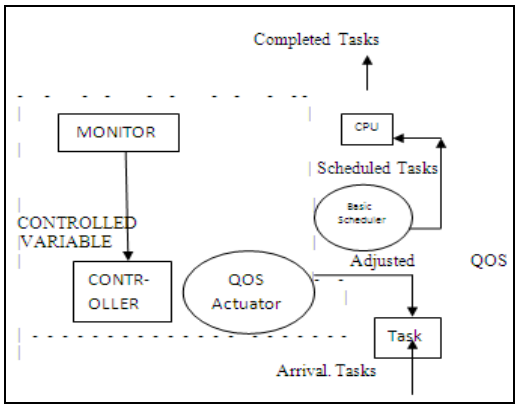

Figure 3: Feedback Control Architecture

3.1.3. Feedback Control Loop

As shown in the Fig2 [4]. The feedback control loop constitutes of monitor controller and the QOS actuator. The *monitor* measures the controlled related variables M (k) and U (k) and feeds it back to the controller. The *controller* finds the error by comparing the controlled variables and the performance reference and denoted it as D_b (k) called the controlled i/p. Controller uses the control function to find out the correct variable and make it close to the reference. The QOS actuator goal is to make

$B(k+1) = B(k) + D_b(k)$. (4)

3.1.4. Merits and Demerits of Feedback Control Scheduling

The feedback control scheduling provides robust performance guarantees for real time systems and supports fundamental resource scheduling [4].The performance of the feedback control scheduling is good under unpredictable environments [4].The RM and EDF algorithms are open loop [4] and these algorithms are based on the assumptions[5],to avoid this uncertainty feedback control scheduling algorithm are employed. From a control theory perspective, it is interesting to investigate the application of robust and adaptive control theory in computer systems with non-linearity's and variations that cannot be handled by the classical linear control scheme.

3.2. Online Adaptive Fault Tolerant System

The basis of online adaptive fault tolerant system comes from feedback control scheduling. Due to the large number of real time constrains and requirements, the design complexity of feedback based control co-design of multiprocessor embedded systems has increased and over 90% of the embedded controllers are used to control real time processes [8].

The online adaptive fault tolerant system is the upper hand as compared to that of feedback control scheduling algorithm. This is implemented in multiprocessor real time systems. Feedback control scheduling algorithm provides the QOS in terms of resource allocation and system performance, and also avoids the overloading of the system and deadline misses. In the recent past for multiprocessor real time systems the Model predictive controller was used but in the recent years model predictive controller won't perform well in unpredictable environment.

3.2.1. Schedulibilty Utilization Bound

If the system predetermined schedule utilization bound is greater than the present environment then no tasks would be entered into the schedule [8]. EX- For RM algorithm the utilization bound does not exceeds m ($2^{1/m}$ - 1) where m is the number of tasks for the processors.

3.2.2. Processor Aim

Each and every processor present in the multiprocessor real time system aim is to maintain each processor utilization between the utilization bound , minimize the error E(k) = y(k) - y(ref) and dynamically adjust the task execution rates.
Utilization model for each processor y(k+1) = y(k) + bΔr(k).
Where y€ R^n represents the processor utilization vector with size n .
Δr signifies the change to task execution rates for m tasks running on the system.
B € R^{n*m} => B=GF.
F is the available subtasks allocation matrix that signifies which tasks are running on which processor.
G is the scalar value that denotes ratio of the change to actual utilization of the processor i and its estimation. The size of G_i measures the estimation error.

3.2.3. Online Adaptive Controller Design.

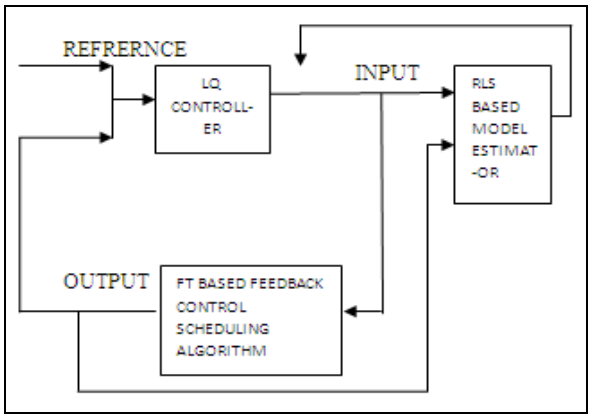

Figure 4: Online Adaptive Controller Design

From the above figure 4 [6] the online adaptive controller design basically comprises of RLS model estimator, LQ controller and the computing system. The RLS and the LQ model do not require the prior knowledge of the system model, which is a huge advantage for the proper functioning of the system.

3.2.3.1. RLS Based Model Estimator

Recursive least square based model estimator basically learns and periodically updates the model and the computing system [7] , the computing system is the fault tolerant based feedback control scheduling algorithm.
The recursive least square is used to find the solutions by recursively finding and taking a cost function and minimizing the cost function by dynamically taking some values which here are referred as filters.
The above figure 4 distributed computing system can be described with the general multiple input multiple output model as follows [6] -

$A(q^{-1})y(k) = B(q^{-1})u(k)+d(k).$ (5)

The above equation is used to find out the relationship between the input and the output of multiple input and multiple output model . $A(q^{-1})$ and $B(q^{-1})$ are the backward shift operators and d(k) represents the disturbances. *Backward shift operators* are useful to find out the results quickly. u(k) is the control i/p and y(k) is the control o/p.
Now as we already know from the fault tolerant based feedback scheduling algorithm

$L(q^{-1})\ y(k) = M(q^{-1})\ u(k) +d(k)$ (6)

where $L(q^{-1})$ and $M(q^{-1})$ are the matrix polynomials in backward shift operators. The *backward shift operators* represent the closest point in the time scale which produces the previous element.

$L(q^{-1}) = L - L_1\ q^{-1} -.....- L_n\ q^{-1}$
$M(q^{-1}) = M_0\ q^{-1} - M\ q^{-1}.....- M_{n-1}\ q^{-1}$ (7)

Here L is the order of the equation in this case the order is 1.

3.2.3.2. LQ Based Optimal Controller

Linear quadratic optimal controller makes the system output track the reference command with small tracking error . It is used to keep the output at the desired output point. The large changes are also avoided to find out the desired output.

The above objective as sited above can be achieved by minimizing the quadratic cost function J [7] .

$J = w(y(k+1) - y_{reference}(k+1))^2 + (q(u(k) - u(k-1))^2$ (8)

Where w = positive semi definite weighting matrix

q= positive definite weighting matrix to penalize large changes in the control input.

The goal of the controller is to steer the system into the optimum reference tracking and penalize the large changes in the control variables.

3.2.4. Merits and Demerits of Online Adaptive Fault Tolerant System

Online adaptive fault tolerant based feedback control scheduling algorithm is based on feedback control scheduling algorithm. The overall system is the addition of RLS and LQ which provides a proper QOS .It provides the proper output according the user requirement and keeps the output at the desired set point. The system also steers the system into the reference tracking output and penalizes the large changes in the output. The system also finds out the output quickly by learning and updating the output as required. The CPU utilization can be more effective with respect to efficiency and provide more QOS as required by the user.

4. Conclusion and Future Work

In the recent years the use of control theory in the real time systems design has increased in a steady state. Basically scheduling is the key notion in the real time systems, because this makes the system performance and resource utilization more perfect. In the past scheduling algorithms like rate monotonic and earliest deadline first used in the real time systems. The above classical algorithms are open loop scheduling algorithms. Open loop here refers that once the tasks are created they are not adjusted based on the continuous feedback. In other words the open loop algorithms will give a correct output if the parameters of the systems are known prior to the execution of the tasks. In the recent years due to the declining rate of the semiconductor devices and the low cost designing of the processors, multiprocessors are more selected to be used. Due to this change in the recent years the multiprocessor systems has emerged as a powerful computing means for the real time a system. With some merits there is a de merit also , the demerit comes as the scheduling of the tasks . The scheduling of the tasks in the open and unpredictable environment is rapidly growing. The unpredictable environment here refers to the resource requirement which are needed are not known priori.

In this unpredictable environment of resource requirement the use of adaptive scheduling algorithm is more preferable . From the concluded work as done till date , we find that the area of scheduling tasks in the multiprocessor environment is an absolute necessary in the field of research for betterment of the systems. This paper has made some contribution towards achieving this goal by giving different aspects of scheduling in multiprocessor real time systems and comparing them to give the brief knowledge about scheduling.

5. References

1. Rajib Mall, Real time systems - Theory and practice, Pearson education, 2007.
2. Antonio P. Mangalhaeis , Mario Z. Rela , Jao G . Silva, Deadlines In Real Time systems, Technical Report, 1993.
3. Hamid Mushtaq, Zaid Al-Ars, Koen Bertels ,Survey of Fault Tolerance Techniques for Shared Memory Multicore/Multiprocessor Systems, 978-1-4673-0469-6/11, 2011, IEEE.
4. Chenyang Lu, Feedback Control Real-Time Scheduling**,** Dissertation Presented to the Faculty of the School of Engineering and Applied Science University of Virginia, May 2001.
5. websitehttp://popart.inrialpes.fr/~girault/Projets/FT/.
6. Oumair Naseer and Rana Atif Ali Khan. Online Adaptive fault tolerant based feedback control Scheduling Algorithm, International Journal of Embedded Systems and Applications (IJESA) Vol.2, No.3, September 2012.
7. Xue Liu, Xiaoyun Zhu Pradeep Padala, Zhikui Wang, Sharad Singhal , Optimal Multivariate Control for Differentiated Services on a Shared Hosting Platform, Decision and Control, 2007 46th IEEE .
8. P. Agrawal. Fault tolerance in multiprocessor systems without dedicated redundancy, IEEE transactions on computers, 37:358-362, March 1988,
9. Jianguo Yao and Xue Liu ,"Online Adaptive utilization control for embedded systems" CODES+ISSS'08, October 19–24, 2008