ISSN 2348 – 8034
Impact Factor- 3.155

GLOBAL JOURNAL OF ENGINEERING SCIENCE AND RESEARCHES

DISTRIBUTED REAL TIME SYSTEM BY USING DEUCON ALGORITHM AN OVERVIEW

Prof.P.B.Lohiya*[1], Prof.D.K.Dakhole[2] and Prof. S.G.Pundkar[3]
*1,2,3Assistant Professor, PRMITR, Badnera

ABSTRACT

Many real-time systems must control their CPU utilizations in order to meet end-to-end deadlines and prevent overload. Utilization control is particularly challenging in distributed real-time systems with highly unpredictable workloads and a large number of end-to-end tasks and processors. This paper presents the End-to-end Utilization CONtrol (EUCON) algorithm and the Decentralized End-to-end Utilization CONtrol (DEUCON) algorithm, which can dynamically enforce the desired utilizations on multiple processors in such systems. In contrast to centralized control schemes adopted in EUCON, DEUCON features a novel decentralized control structure that requires only localized coordination among neighbor processors. DEUCON is systematically designed based on recent advances in distributed model predictive control theory.

Keywords- *Real-time systems, embedded systems, distributed systems, feedback control real-time scheduling, end-to-end task, quality of service, model predictive control.*

I. INTRODUCTION

In recent years, a category of performance-critical distributed systems executing in open and unpredictable environment has been rapidly growing [2]. Examples of such systems include distributed real-time embedded (DRE) systems such as avionics mission computing, autonomous aerial surveillance, disaster recovery systems, and multi-tier E-business servers such as on-line trading servers. A key challenge in developing such systems is providing critical Quality of Service (QoS) guarantees while the workload and the underlying platform cannot be accurately characterized a priori. For example, the execution times of visual tracking applications can vary significantly as a function of the number of potential targets in a set of received camera images[1][3].Similarly, the resource requirements and the arrival rate of service requests in an on-line trading server can fluctuate dramatically. However, QoS guarantees are required in these systems despite their unpredictability. In particular, such systems often need to guarantee the CPU utilization on multiple processors in order to achieve overload protection and meet end-to-end deadlines. Failure to meet critical QoS guarantees such as CPU utilization constraints may result in loss of customers, financial damage, liability violations, or mission failures. as DRE systems become connected to the Internet, they are exposed to load disturbances due to variable user requests and even cyber attacks [9]. As such systems become increasingly important to our society, a new paradigm of real-time computing based on Adaptive Quality-of-service (QoS) Control (AQC) has received significant attention. In contrast to traditional approaches to real-time systems that rely on accurate knowledge about the system workload, AQC can provide robust QoS guarantees in unpredictable environments by adapting to workload variations based on dynamic feedback. A key advantage of AQC is that it adopts a control theoretic framework for systematically developing adaptation strategies. This paper, focus on utilization control, which is an important instance of AQC for distributed soft real-time systems. The goal of utilization control is to enforce the desired CPU utilizations on all the processors in a distributed system despite significant uncertainties in system workloads. Utilization control can be used to enforce appropriate schedulable utilization bounds on all processors to guarantee end-to-end task deadlines. It can also enhance system survivability by providing overload protection against workload fluctuation. DRE systems introduce many new research challenges that have not been addressed in earlier work on single processor systems. First, they require multiple-input multiple output (MIMO) control solutions to manage the system QoS on multiple processors. Second, the QoS of different processors are often coupled with each other due to complex interactions among distributed application components. In particular, many DRE systems employ the common end to end task model [4], where a task may comprise a chain of subtasks on different processors. In such systems, the CPU utilizations of different processors cannot be controlled independently. For example, changing the rate of a task will affect the CPU utilizations of all the processors where its subtasks are located. Therefore, the coupling among processors must be modeled and addressed in the design of QoS control algorithms. Finally, a utilization control algorithm must be highly scalable in order to handle large DRE systems. A centralized control algorithm is often inadequate for such systems since its communication and computation overhead usually depends on the size of the entire DRE system. This paper presents the Decentralized End-to-end Utilization CONtrol (DEUCON) algorithm for large DRE systems with end-to-end tasks. In sharp contrast to earlier solutions based on centralized control schemes [3]. DEUCON employs a completely decentralized control approach that can scale well in large distributed systems and tolerate individual processor failures.

II. END TO END UTILIZATION CONTROL

EUCON is an adaptive algorithm that features a MIMO feedback control loop (see Figure 1) that dynamically adjusts task rates to enforce the utilization set points. The DRE sys-tem is controlled by a centralized MIMO controller. The controller may be located on a separate processor, or share a processor with some applications. EUCON must be scheduled as the highest-priority task in order to effectively control utilization under overload conditions. Each processor has a *utilization monitor* and a *rate modulator*. A separate TCP connection connects the controller with the pair of utilization monitor and rate modulator on each processor. The user inputs to the controller include the utilization set points, $B = [B_1 \dots B_n]^T$ and the rate constraints on each task. The *controlled variables* are the utilization of all processors, $u(k) = [u_1(k) \dots u_n(k)]^T$. The *control inputs* from the controller are the change to task rates

$$\Delta r(k) = [\Delta r_1(k) \dots \Delta r_m(k)]^T,$$

where

$$\Delta r_i(k) = r_i(k) - r_i(k\text{-}1)\ (1 \le i \le m).$$

The following feedback control loops are invoked in the end of every sampling period:

1. The utilization monitor on each processor sends the utilization $u_i(k)$ in the last sampling period to the controller through its feedback lane.
2. The controller collects the utilization vector $u(k)$, computes $\Delta r(k)$, and sends the new task rates to the rate modulator on each processor through its feedback lane.
3. The rate modulator on each processor changes the task rate according to the input from the controller.

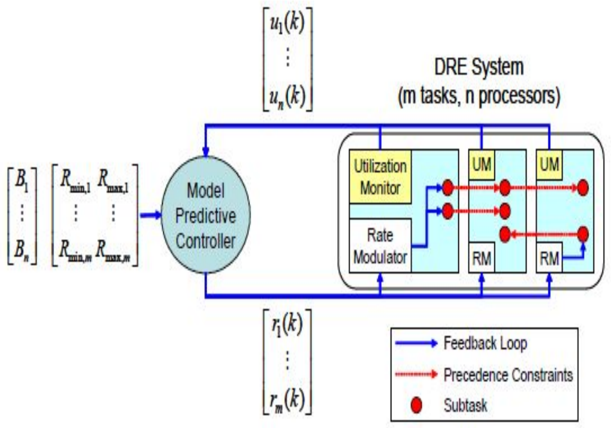


Figure 1: The MIMO feedback control loop in EUCON

2.1 Design of a Model Predictive Controller

Based on the system model, a MIMO controller can be de-signed to guarantee the utilization set points on multiple processors. The PID control approach adopted in earlier works on feedback control real-time scheduling [13][14] is not suitable for DRE systems due to the coupling among multiple processors and the constraints. To solve this control problem, a *Model Predictive Control (MPC)* [15] approach is defined. MPC is an advanced control technique used extensively in industrial process control applications. Its major advantage is that it can deal with coupled MIMO control problems with constraints on the plant and the actuators. This characteristic makes MPC very suitable for end-to-end utilization control in DRE systems that can be represented by MIMO system mod-els under a set of constraints. The basic idea of MPC is to optimize an appropriate cost function defined over a time interval in the future.

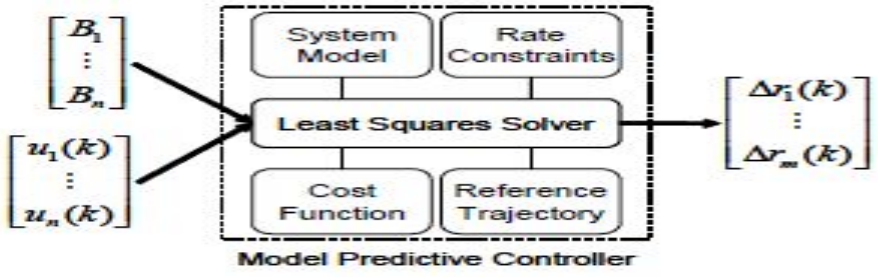

$\begin{bmatrix} B_1 \\ \vdots \\ B_n \end{bmatrix}$
$\begin{bmatrix} u_1(k) \\ \vdots \\ u_n(k) \end{bmatrix}$
System Model
Rate Constraints
Least Squares Solver
Cost Function
Reference Trajectory
Model Predictive Controller
$\begin{bmatrix} \Delta r_1(k) \\ \vdots \\ \Delta r_m(k) \end{bmatrix}$

Figure2: The Model Predictive Controller

2.2 Applications

EUCON provides a powerful technique for utilization control in a broad range of QoS critical systems.

• *End-to-end real-time scheduling*: In DRE systems, real-time tasks must meet their end-to-end deadlines[6][9].In the end-to-end scheduling approach, the deadline of an end-to-end task is divided into sub deadlines of its sub-tasks, and the problem of meeting the deadline is transformed to the problem of meeting the sub deadline of each subtask. A well known approach for guaranteeing the sub deadlines on a processor is by enforcing the *schedulable utilization bound*. The sub deadlines of all the subtasks on a processor are guaranteed if the utilization of the processor remains below the schedulable utilization bound of its subtasks. For EUCON to guarantee end-to-end deadlines, a user should specify the utilization set point of each processor to no higher than its schedulable utilization bound. Existing real-time scheduling theory has established various schedulable utilization bounds for different task models [8].

• *QoS portability*: EUCON can also be implemented in DRE middleware to support *QoS portability* [10]. When an application is deployed on a faster platform, the task rates will be automatically increased to take advantage of the extra resource. On the other hand, when an application is deployed to a slower platform, task rates will be automatically reduced to maintain the same CPU utilizations guarantees. EUCON's self-tuning capability can significantly reduce cost of porting real-time applications across platforms.

• *Overload protection*: Most QoS-critical systems (e.g., E-business servers) desire to avoid saturation of processors, which may cause system crash or severe service degradation. In COTS operating systems that support real-time priorities, high utilization by real-time threads may cause kernel starvation. EUCON allows a user to en-force desired utilization bounds for all the processors in a distributed system. Moreover, the utilization set point can be changed online. For example, a user may lower the utilization set point on a particular processor in anticipation of future workload, and EUCON will dynamically readjust task rates to enforce the new set point.

2.3 Disadvantages of EUCON:

- First, the runtime overhead depends on the size of an entire DRE system. Specifically, the worst-case computational complexity of an MPC is polynomial in the total number of tasks and the total number of processors in the system. Furthermore, since every processor in the system needs to communicate with the controller in every sampling period, the processor executing the controller can become a communication bottleneck. Therefore, a centralized control scheme cannot scale effectively in large DRE systems.
- Second, the control design of EUCON assumes that communication delays between the control processor and other processors are negligible compared to the sampling period of the controller. This assumption may not hold in networks with significant delays such as the Internet and wireless sensor networks. In addition, the processor executing the controller is a single point of failure. The entire system will lose the capability to adapt to the environment if it fails. Centralized solutions are therefore not suitable for large scale DRE. This paper, focus on developing decentralized control algorithms to improve the scalability and reliability of adaptive utilization control in DRE systems.

III. DESIGN OF DEUCON

To address the drawbacks of centralized control this paper discuss a more scalable control solution called *DEUCON* (Decentralized End-to-end Utilization CONtrol) that can dynamically enforce desired utilizations on multiple processors in large-scale DRE systems. In contrast to centralized control schemes, DEUCON features a novel decentralized control structure that requires only localized coordination among neighbor processors. DEUCON is systematically designed based on recent advances in distributed model predictive control theory. The DEUCON algorithm featuring a novel peer-to-peer control structure that enforces desired utilizations of multiple processors through localized coordination among controllers. A fundamental design challenge is to achieve system stability and desired utilizations without global information. DEUCON employs a peer-to-peer control structure with a separate local controller Ci on each master processor Pi. Each controller only coordinates with a small number of processors called its (logical) neighbors. As a foundation of our control design, first present a dynamic model of the entire system and an approach for decomposing the global system model into localized control sub problems[1].

3.1 Global System Model

A DRE system can be approximated by the following *global* system model [5]:

$$u(k+1) = u(k) + GF\Delta r(k) \quad (3.1)$$

The vector $\Delta r(k)$ represents the changes in task rates. The *subtask allocation matrix*, F, is an $n \times m$ matrix, where $f_{ij} = c_{jl}$ if a subtask T_{jl} of task T_j is allocated to pro- cessor P_i, and $f_{ij} = 0$ if no subtask of task T_j is allocated to processor P_i. F captures the *coupling* among processors due to end-to-end tasks. G = $diag[g_1 : : : g_n]$ where g_i represents the ratio between the change in the actual utilization and its estimation.
The exact value of g_i is *unknown* due to the unpredictability in execution times. Note that G describes the effect of uncertainty in workload on the utilization of a DRE system. Figure 3 shows a DRE system with five processors and five tasks.

3.2 Problem Decomposition

From a local controller *Ci*'s perspective, the goal of decomposition is to partition the set of system variables into three subsets, including *local variables* on host processor *Pi*, *neighbor variables* on *Pi*'s neighbors, and all other variables in the system. *Ci*'s sub problem only includes its local and neighbor variables. Consider several definitions before presenting our decomposition scheme [1].

- *Definition 1*. Processor Pj is Pi's direct neighbor if 1) Pj has a subtask belonging to an end-to-end task mastered by Pi and 2) Pj is not Pi itself.
- *Definition 2*. The concerned tasks of Pi are the tasks that have subtasks located on Pi or Pi's direct neighbors.
- *Definition 3*. Processor Pj is Pi's indirect neighbor if 1) Pj is the master processor of any of Pi's concerned tasks and 2) Pj is not Pi's direct neighbor or Pi itself. For example, consider controller C1 in the system shown in Figure 3. P1 has one direct neighbor (P2) due to task T1 mastered by P1. Its concerned tasks include T1, T5, and T2 (which has a subtask on direct neighbor P2). Hence, P3, the master processor of T2, is P1's indirect neighbor.

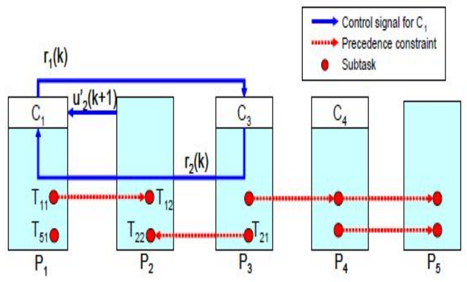


Figure 3: Data exchange between C1 and its neighbors

In DEUCON has a localized model which only includes its local and neighbor variables. This local model of *Ci* is described as:

$$nu_i(k+1)=nu_i(k)+G_i\ F_i\Delta nr_i(k)\ (3.2)$$

where nui(k) and nri(k) are vectors comprised of all elements in *NUi*(*k*) and *NRi*(*k*), respectively. Gi and Fi are defined in the same way as G and F in 2, but include only the processors in *NUi*(*k*) and the task rates in *NRi*(*k*).
For example, the controller *C*1 shown in Figure 2, is modeled with the following parameters.

$$G_1=\begin{bmatrix} \\ \\ \end{bmatrix} \qquad F_1=\begin{bmatrix} \\ \\ \end{bmatrix}$$

$$nu_1(k)=\begin{bmatrix} \\ \\ \end{bmatrix} \qquad \Delta nr_1(k)=\begin{bmatrix} \\ \\ \end{bmatrix}$$

From 3.2, *C*1's local model is

$$u_1(k+1) = u_1(k)+g_1(c_{11}\ \Delta r_1(k)+c_{51}\Delta r_5(k))$$

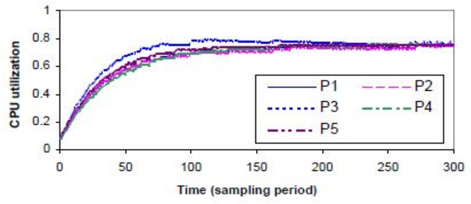
$$u_2(k+1) = u_2(k)+g_2(c_{12}\Delta r_1(k)+c_{22}\Delta r_2(k))$$

3.3Advantages of DEUCON

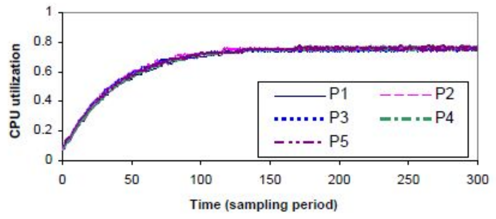
Compared to centralized control schemes, a fundamental advantage of DEUCON is that both the computation and communication overhead of a controller depend on the size of its neighborhood instead of the entire system. This feature allows DEUCON to scale effectively in many large DRE systems. Another important advantage of DEUCON is that it can tolerate considerable network delays. DEUCON can tolerate much longer communication delays than EUCON, which assumes the delays to be negligible. This approach can be easily extended to handle larger communication delays. DEUCON can also improve system fault tolerance by avoiding a centralized controller, which is a single point of failure in the whole system. In DEUCON, even if the system failure of a processor may disable a local controller, the subtasks on the failed processor can be immediately migrated to their backup processors and then be effectively controlled by other local controllers there. As a result, single processor failures will not cause the system to lose control in DEUCON.

IV. SYSTEM PERFORMANCE

Figures 4(a) and (b) show the utilizations of processors *P*1 to *P*5 when execution times of tasks are *one-eighth* of their estimations. In this case observe a noticeable difference in the transient state between DEUCON and EUCON. While the utilizations of EUCON follow the same trajectory, utilizations of DEUCON diverge in the middle of the run and then converge to their set points in the end.



(a) DEUCON



(b) EUCON

Figure 4: CPU utilization of P1 to P5 (ietf=8)

To further investigate the CPU utilizations on other processors, Figure 4 plots the average utilizations of all processors when *ietf* is 5. The deviations of all utilizations are less than 0.008. Observe that on *P*2 to *P*7, the difference between the utilizations and the set points for DEUCON are slightly larger than that of EUCON.

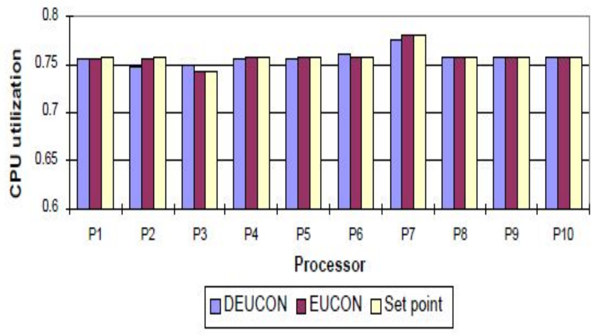


Figure 5: Average CPU utilization (ietf=5)

V. DISCUSSIONS

DEUCON is particularly efficient for systems with long running tasks with fixed routes. When the task allocation changes dynamically due to task arrival, termination, or route changes, any controller whose direct or indirect neighborhoods are affected needs to reconfigure its internal model and update its neighborhood information in order to maintain utilization control. For instance, when a task changes its route, all the controllers within its old and new neighborhoods need to reconfigure their models and neighborhood information. Note that, thanks to the localized control scheme of DEUCON, usually only a subset of the controllers needs to perform reconfiguration in response a workload change. In an earlier work, people have successfully implemented this controller reconfiguration mechanism for a EUCON controller in a DRE middleware[11]. DEUCON is designed to control the aggregate utilization of each processor. An alternative to the approach is per-task control in which the system monitors and controls the utilization of each individual task. There exists a trade-off between the granularity and the overhead of control. Although a per-task control approach may enable fine-grained control over individual tasks and achieve performance isolation among tasks, it also introduces a higher overhead because it requires monitoring the utilization of every subtask, which is commonly implemented as a separate thread in real-time middleware systems. In contrast, although the per-processor control approach adopted by DEUCON cannot support fine-grained control over individual tasks, it can be more efficient because it only needs to monitor the utilization of each processor. In general, the choice of control approaches depends on application requirements and platform characteristics. This paper adopt the per-processor control approach because aim to implement DEUCON as a middleware service for large scale DRE systems. Monitoring the utilization of every subtask at the middleware (user) level may introduce a non negligible overhead in those systems.

VI. CONCLUSION

This paper extends the QoS control framework from single-processor to DRE systems. The challenging end-to-end utili zation control problem is solved by a model predictive control approach. DEUCON features a novel decentralized control structure to handle the coupling among multiple processors due to end-to-end tasks. DEUCON can significantly improve the system scalability by distributing the computation and communication cost from a central processor to local controllers distributed in the whole system and tolerating network delays.

REFERENCES

1. *Xiaorui Wang, Dong Jia, Chenyang Lu, and Xenofon Koutsoukos "DEUCON: Decentralized End-to-End Utilization Control for Distributed Real-Time Systems", IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 18, NO. 7, JULY 2007*
2. *T.F. Abdelzaher, et. al., "Feedback Performance Control in Software Services," IEEE Control Systems, 23(3), June 2003.*
3. *D. Henriksson and T. Olsson, "Maximizing the Use of Computational Resources in Multi-Camera Feedback Control," Proc. IEEE Real-Time and Embedded Technology and Applications Symp. (RTAS '04), May 2004.*
4. *J.W.S. Liu, Real-Time Systems. Prentice Hall, 2000.*
5. *C. Lu, X. Wang, and X. Koutsoukos, "Feedback Utilization Control in Distributed Real-Time Systems with End-to-End Tasks," IEEE Trans. Parallel and Distributed Systems, vol. 16, no. 6, pp. 550-561, June 2005.*
6. *B. Kao and H. Garcia-Molina,"Deadline Assignment in a Dis-tributed Soft Real-Time System," IEEE Transactions on Paral lel and Distributed Systems, Dec. 1997.*
7. *J.P. Lehoczky, "Fixed Priority Scheduling of Periodic Task Sets with Arbitrary Deadlines," IEEE Real-Time Systems Symposium, 1990.*
8. *C.L. Liu and J.W. Layland, "Scheduling Algorithms for Multi-programming in a Hard Real-Time Environment," Journal of ACM, 20(1): 46-61, 1973.*
9. *J. Sun and J.W.S. Liu, "Synchronization Protocols in Distributed Real-Time Systems," International Conference on Distributed Computing Systems, 1996.*
10. *C. Lu, X. Wang, and C.D. Gill, "Feedback Control Real-Time Scheduling in ORB Middleware," IEEE Real-Time and Embedded Technology and Applications Symposium, May 2003.*
11. *X. Wang, C. Lu, and X. Koutsoukos, "Enhancing the Robustness of Distributed Real-Time Middleware via End-to-End Utilization Control," Proc. IEEE Real-Time Systems Symp. (RTSS '05), 2005.*
12. *Xiaorui Wang, Xing Fu, Xue Liu, and Zonghua Gu. "Power-Aware CPU Utilization Control for Distributed Real-Time Systems"*
13. *C. Lu, J.A. Stankovic, G. Tao, and S.H. Son, "Feedback Control Real-Time Scheduling: Framework, Modeling, and Algorithms," Real-Time Systems Journal, 23(1/2): 85-126, 2002.*
14. *J.A. Stankovic, et. al., "Feedback Control Real-Time Scheduling in Distributed Real-Time Systems," IEEE Real-Time Systems Symposium, 2001.*
15. *J.M. Maciejowski, Predictive Control with Constraints, Prentice Hall, 2002.*