I.J. Information Engineering and Electronic Business, **2015, 3, 1-10**
Published Online May 2015 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijieeb.2015.03.01

Evaluation of Load Balancing Performance of Parallel Processing Linear Time-Delay Systems

Sohag Kabir
Department of Computer Science, University of Hull, Hull, UK
Email: s.kabir@hull.ac.uk

A S M Ashraful Alam
Department of Computer Science, University of Otago, Dunedin, New Zealand
Email: aalam@cs.otago.ac.nz

Tanzima Azad
Department of Computer Science, Bangladesh University of Engineering & Technology, Dhaka, Bangladesh
Email: azad.tanzima@dbbl.com.bd

Abstract—Time delays in system states or control may result into unacceptable system operation or uncertainty in specialised technical systems like aircraft control, plant control, robotics etc. The issue of robustness, controllability, traceability, flexible management, reliability, and safety of such systems with time-delays, has been one of the primary research focuses of the last few decades. In parallel computing, different computing subunits share their tasks to balance loads to increase performance and throughput. In order to do so, subsystems have to communicate among themselves, adding further delay on top of existing system delay. It is possible to maintain performance and stability of the whole system, by designing observer for every subsystem in the system, overseeing the system state and compensating for existing time-delay. This paper reviews present literature to identify a linear time-delay system for load balancing and evaluates the stability and load balancing performance of the system with and without an observer. Stability is analysed in terms of oscillation in the system responses and performance is evaluated as the speed of load-balancing operation.

Index Terms—Parallel Processing, Time-delay Systems, Load Balancing, Cluster Computing, Observer Design, Networked System.

I. Introduction

Parallel computing is the dominant paradigm in system architecture of large complex systems. Different organisations like the Federal Bureau of Investigation (FBI) is using parallel computing techniques to develop and manage the Combined DNA Index System (CODIS) and National DNA Index System (NDIS) [1]. Simultaneous calculations are carried out in parallel-computing systems, following the principle that big problems can be split into smaller ones and, then can be solved simultaneously. A busy computing unit may share some of its loads with a relatively idle computing unit, resulting in a significant increase in system performance and throughput. However, if these computing units are in a network, then information about the workload on the processing subunits is not instantly available to other subunits due to time-delay experienced during communication, which affects system performance adversely. Communication and synchronization among subsystems are typically some of the greatest challenges to performance in parallel computing [2].

Time-delay is one of the most significant issues for reliability in real-time control system. Time-delay systems are usually known as systems with deceased time, or systems of differential-difference equations, or systems of equations with deviation in argument. They are a constituent of infinite dimensional functional differential equations (FDEs) which are opposite of ordinary differential equations (ODEs) [3]. In clustered machine environment, additional communication delay is inevitable. This may lead to the system being unstable, uncontrollable or untraceable. The needs for observing time-delay in complex systems comes from the real requirement of system tracing, control and/or detection of system failure. In order to do so, it is often needed to redefine the parameters representing the states in a system with time delay. The difficulties concerning sensitivity along with robustness of the feedback system in regard to time delays has attracted much interest in academia and industries because of the effects of the time-delays on the performance of such systems. For a few systems, small delays result in destabilisation while many other systems are tolerant to small time delays [4]. In parallel computing, a significant amount of communication takes place among subsystems, therefore the subsystems experience time-delays during mutual communication that affects the system performance. The adverse effect of time-delay can be minimised by observing the states of time-delay systems and then utilising the information to model a linear time-delay system aimed at designing observers for all subsystems to minimise delays and increase stability.

This paper extends the work presented in [5], to evaluate load balancing performance of systems in the presence of liner time-delay. To do so, a linear time-delay system is modelled and simulated in SIMULINK to analyse the stability and performance of the system in performing load balancing operation. Stability is analysed in terms of oscillation in system responses. At the same time, performance is measured as speed of operation, i.e. how quickly the system can balance loads between subsystems. It is observed that the system responses oscillate more with higher gain in comparison to lower gain due to delays experienced by subsystems during communication. System gain is referred to as the reduction rate of waiting time (speed of operation) and value of system gain is constrained by delays experienced by the subsystems. During analysing the stability of the system, a linear control law is used instead of a saturated control law of real system. Because system responses oscillate more with linear control law therefore it is considerably easier to analyse the oscillatory behaviour of the system. After analysing system performance and stability, relationships among system gain, stability of the system performance and the delays experienced by the subsystems due to mutual communications are established. We have designed the observers for all subsystems of the parallel processing linear time delay system from the findings of these analyses. Observer of each subsystem estimates some necessary values for the subsystem to allow it to work independently without communicating with other subsystems thus minimising the effect of time-delay in the system performance. To evaluate the system performance the system with observers are modelled and simulated using SIMULINK.

The rest of the paper is organized as follows: Section II presents the background study and literature review on time-delay systems, observer design and load-balancing techniques. The parallel processing linear time-delay system is described, modelled, simulated, and results of the simulation are shown in Section III. In Section IV, observers are designed, system with observers is modelled, simulated and the result of the simulation is presented. Finally, a discussion on the results and concluding remarks are presented in Section V and VI respectively.

II. Background Study and Literature Review

A. *Time-Delay Systems and Their Mathematical Models*

Time-delay is an observable fact that occurs regularly in various systems, in different ways around us. A very familiar application where the delay arises is a live telecast on television. The video clip seen on TV differs in time from the actions take place in real time during live broadcast. Time-delay is encountered regularly in different control systems; either in the systems states or the control input [6]. Time-delay can be contributed as a source of instability in many systems. From a research perspective, the issue of delay is very important for performance and consistency of control systems. A considerable amount of research on performance analysis of time-delay systems has been performed in the last decade and researcher like [7]–[10] have contributed to stability investigation of time-delay systems. Robustness of time delay system is studied in [11]. Improved time-delay systems and stabilised controllers are suggested in [6], [12], [13]. A network-based controller for time-delay systems is studied in [14] and steadiness of network-based control system is analysed in [15]. In [1], a linear time delay model was studied to see load balancing instabilities in parallel computing.

Time-delay system can be modelled in different ways. It can be described by the differential equation model, transfer function model, and others. A simple linear time-delay system can be represented as:

$$\dot{x}(t) = a\,x(t) + bu(t) \tag{1}$$

where a and b are constants, $x(t)$ and $u(t)$ are functions that change with time. Replacing $x(t)$ by $\sin x(t)$ in (1), we find the following non-linear equation for the time delay system:

$$\dot{x}(t) = a \sin x(t) + bu(t) \tag{2}$$

The equation can be rewritten for a time delay τ in following way with another constant c:

$$\dot{x}(t) = a\,x(t) + bu(t) + cx(t-\tau) \tag{3}$$

It is also possible to model a time-delay system using delay differential equation (DDE). In DDE, the derivative of an unknown function at a specific time is expressed with regard to the values of the function at earlier time. DDE is similar to ODE in mathematical treatment, but their evolution involves previous values of the state variables. Therefore it requires the knowledge of not only the present state, but also the state knowledge at a certain historical time for the solution of DDE. A general form of the time-delay differential equation for $x(t) \in \mathbb{R}^n$ is:

$$\frac{d}{dx}x(t) = f(t, x(t), x_t) \tag{4}$$

where $x_t = \{x(\tau) : \tau \le t\}$ represents the trajectory of the solution in the past.

Different types of DDE can be used to address delay of various natures. Equation (5) expresses a system with continuous delay:

$$\frac{d}{dx}x(t) = f(t, x(t), \int_{-\infty}^{0} x(t+\tau)\,d\mu(\tau)) \tag{5}$$

For a system with discrete delay, the DDE takes the following form:

$$\frac{d}{dx}x(t) = f(t, x(t), \ldots, x(t-\tau_m)) \qquad (6)$$

for $\tau_1 > \tau_2 > \ldots > \tau_m \geq 0$. DDE for a linear system with discrete delay takes following form:

$$\frac{d}{dx}x(t) = A_0 x(t) + A_1 x(t-\tau_1) + \ldots + A_m x(t-\tau_m) \qquad (7)$$

where $A_0, A_1, \ldots, A_m \in \mathbb{R}^{n\times n}$.

B. Control Systems and Observers

Control systems use sensors to determine and control the state of the variables/quantities like movement, heat circulation, temperature, stress, fluid pressure, etc. Sensors have some limitations [16] that may cause systems instability and problems like:

- Usually sensors are expensive that can raise the system cost significantly.
- Sensors itself and required wiring to incorporate in control system may reduce dependability and consistency of a control system.
- Occasionally, it is difficult for sensors to measure faint signal practically due to difficulty in accessing inconsiderate environment or relative movement between sensors and objects.
- Usually sensors bring about considerable errors such as cyclical errors, stochastic errors and limited responsiveness.

Internal states of many systems cannot be observed directly. This raises problems in feedback systems and consequently rules out the possibility of state feedback. A separate system, termed as an observer or an estimator that attempts to duplicate the values of the state vector, can be designed.

Observers are methods that merge with sensed quantities alongside the other information of the control system to generate control signals. Observers can be used to replace or as a supplement to the sensors in a control system. Observer generated control signals can be more precise, cheaper to generate, and more trustworthy than the control signals produced by sensors. Observers provide engineers an attractive option to accumulate new sensors or improve upon current ones [16]. A sample role of observer in a control system is shown in Fig. 1 and the general form of Luenberger observer is shown in Fig. 2.

C. Observers for Time-Delay Systems

System performance is affected by the delay introduced during system operation. An observer can be employed to observe the states of the system to address the time-delay problem. Different approaches like Kronecker canonical form, geometric, inversion approaches, algebraic, singular value decomposition, and generalised inverse techniques can be employed to design observer for systems with time-delay. Several varieties of state observer e.g., unknown input, disturbance decoupled, full-order, minimal-order, functional, etc. have been systematically investigated in [17]. Most frequently used observer systems are Luendberger observer, Kalman filter, and adaptive observer.

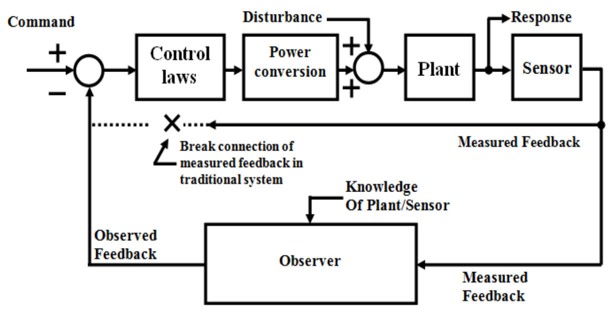

Fig. 1. Observer and Control System [16]

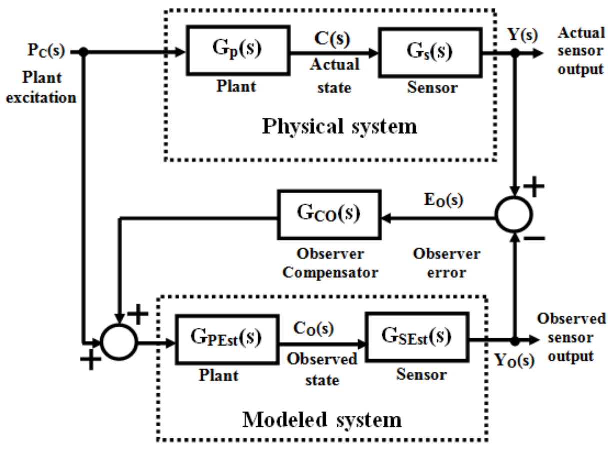

Fig. 2.General form of Luenberger Observer [16]

Kalman filter is often a recursive estimator for stochastic techniques, which usually consists of noise and processes with known parameters. It has a correction factor to insure stability of the observer through prediction and update. In *predict* step, it estimates current state variables and their vagueness. In *update* step, the current state variables are used to accurately enhance the estimation as it arrives at a new state estimate.

The ways of using a Kalman filter to design observer for time-delay systems were shown in [18]. The methodologies for designing robust observer for time-delay systems were described in [19], [20]. Time-delay system with memoryless state observer was presented in [21]. In [22], an observer for linear time-delay systems was designed using coordinate transformations. An observer for linear time-delay systems with unknown input was designed in [23]. Geometric design of observers for linear time-delay systems was proposed by Perdon and Anderlucci in [24]. Hou, Busawon, and Saif had discussed observer design techniques based on triangular form generated by injective map in [25]. Discontinuous observers for non-linear time-delay systems were designed in [26] and a neural observer was designed using coordinate transformation in [27]. Presently, network-based observer design is an emerging topic because of the constraint that all the required system components cannot be accommodated at same place in many systems. These components add further transmission delay by using network medium for communicating.

D. Load Balancing Techniques

In parallel computing, a group of processing units communicates and collaborates to solve big problems competently. Creating parallel programs requires first breaking the computation task into smaller chunks and distributing the chunks to the processors; this is called partitioning. The optimisation objective of partitioning is to stabilise the workload among processing units and to minimise the inter process communication needs. Some processors can become idle if the loads are not balanced properly among the processors; therefore efficiency of the parallel processing systems are dependent on the efficiency of the load balancing algorithms, i.e., how effectively the load balancing algorithm can evenly balance loads among the processors. Usually, a load balancing algorithm comprises of three stages – a) Information collection, b) Decision-Making and c) Data migration. The broad classification of load balancing were discussed in [28], shown in Fig. 3. This framework is suitable for explaining as well as classifying prevailing load balancing techniques, simplifying the task of finding an appropriate load balancing approach.

III. Linear Time-Delay System To Study Load Balancing Performance

A. System Description

In many applications, in order to increase throughput and reliability, computer clusters are used instead of a single computer where a number of readily available computing nodes are available to facilitate parallel computing. In this paper, a computer cluster comprising of *n* nodes is considered which was first used in [1].

In the model, all the nodes can communicate with each other and at the very beginning of operation, all nodes are provided with equal work load. With the passage of time, the workloads on different processing units become uneven due to the varied processing capability of the processing units. To balance loads among nodes, each node in the cluster sends its load information $q_j(t)$ (queue length) to other nodes of the cluster. Due to time-delay node i obtains information sent by node j delayed by a certain period of time τ_{ij}; i.e., $q_j(t)$ is received as $q_j(t-\tau_{ij})$. After receiving information from all other nodes, each node locally calculates and determines the average amount of load using a simple estimator $\frac{\sum_{j=1}^{n} q_j(t-\tau_{ij})}{n}$, where $\tau_{ii}=0$. Then each node compares its own queue size against locally calculated queue size to make decision about task sharing. If the size of the queue of a node is greater than the average queue size then the node will send a portion of its tasks to other nodes. The scenario is shown in Fig. 4.

In the figure, the horizontal line indicates the average load predicted by processing unit 1 and the vertical bars represent loads for each node. As seen in Fig.4, node 1 has more load than the average estimated load of the network; therefore this node will decide to share its load with other nodes. In this example, node 3, 5, 6, and 8 are the nodes with fewer loads than average load in the network. Again, the sending tasks also add further delay in the course of its execution. As a result, node j receives tasks sent from node i with a time-delay of h_{ij}.

The load balancing procedure determines the frequency of performing load balancing operations. A smart load balancing algorithm should consider that each processing unit has delayed value of queue size and tasks send from one processing unit to other is received with a certain time delay.

B. Mathematical Model of the System

The mathematical model of the above system can be described as [1]:

$$\frac{dx_i(t)}{dt} = \lambda_i - \mu_i + u_i(t) - \sum_{j=1}^{n} p_{ij} \frac{t_{pi}}{t_{pj}} u_j(t-h_{ij}) \tag{8}$$

$$u_i(t) = -K_i y_i(t) \tag{9}$$

$$y_i(t) = x_i(t) - \frac{\sum_{j=1}^{n} x_j(t-\tau_{ij})}{n} \tag{10}$$

where, n is the number of processing unit in the cluster; $x_i(t)$ is the estimated waiting time faced by a task placed into the queue of node i; λ_i is the rate of increase of waiting time at node i and μ_i is the rate of reduction in waiting time at node i.

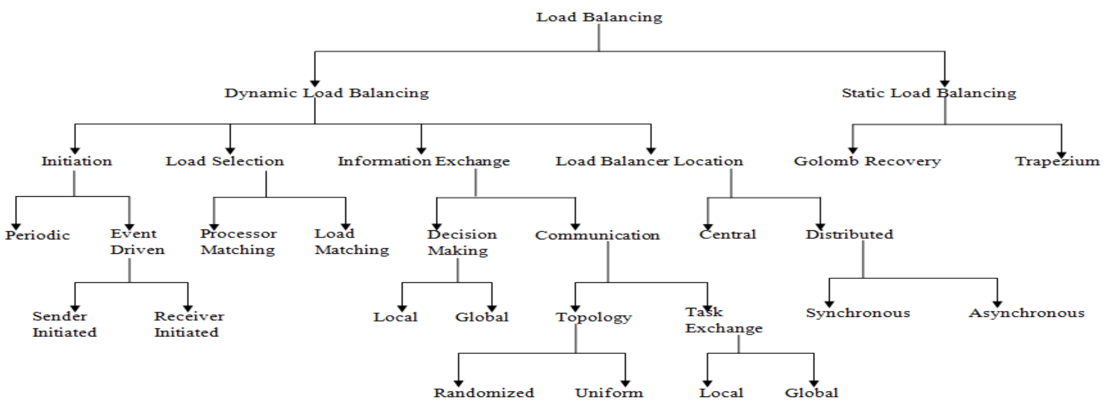

Fig. 3. Classification of Load Balancing Algorithms

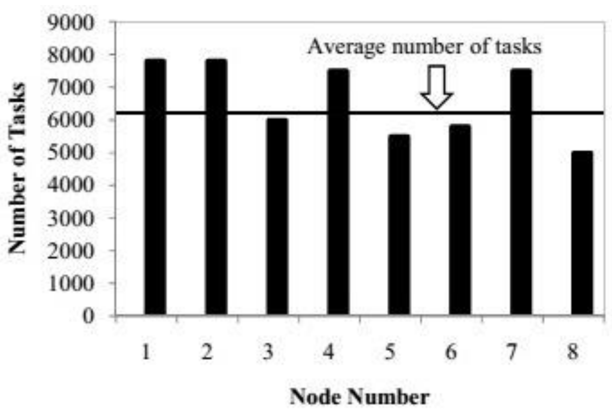

Fig. 4. General example of load balancing process

$u_i(t)$ is the transmission rate of the tasks from processing unit *i* at time instance *t* to other processing units; $p_{ij}u_i(t)$ is the transmission rate of waiting time from node j to node i at time instance t where $p_{ij} \geq 0, p_{jj} = 0$, and $\sum_{j=1}^{n} p_{ij} = 1; -p_{ij}u_i(t-h_{ij})$ is the rate of transmission of the estimated waiting time from processing unit j by (to) processing unit i at time instance t where h_{ij} is the time delay for the load transfer from processing unit j to unit i and $h_{ii} = 0$; and τ_{ij} is the time delay for communicating queue size from node j to node i.

In this paper, for simplicity, we have considered three computing nodes instead of n nodes generalised in the above equations. Hence, the equation for subsystem 1 in the cluster can be described as:

$$\dot{X}_1 = \lambda_1 - \mu_1 + u_1(t) - p_{11}u_1(t-h_{11}) - p_{12}u_2(t-h_{12})$$
$$-p_{13}u_3(t-h_{13})$$
$$= \lambda_1 - \mu_1 + u_1(t) - p_{12}u_2(t-h_{12}) - p_{13}u_3(t-h_{13})$$
$$= \lambda_1 - \mu_1 - K_1\left[X_1 - \frac{1}{3}(X_1 + X_2(t-\tau_{12}) + X_3(t-\tau_{13}))\right]$$
$$+p_{12}K_2\left[2X_2 - \frac{1}{3}(X_1(t-\tau_{21}-h_{12}) + X_3(t-\tau_{23}-h_{12}))\right]$$
$$+p_{13}K_3\left[2X_3 - \frac{1}{3}(X_1(t-\tau_{31}-h_{13}) + X_2(t-\tau_{32}-h_{13}))\right]$$

Now if we consider the delay experienced by nodes during sharing load information as $\tau_{ij} = \tau$ and the delay to share tasks as $h_{ij} = 2\tau$ then the above equation takes following form:

$$\dot{X}_1 = \lambda_1 - \mu_1 - K_1\left[X_1 - \frac{1}{3}(X_1 + X_2(t-\tau) + X_3(t-\tau))\right]$$
$$+p_{12}K_2\left[2X_2 - \frac{1}{3}(X_1(t-3\tau) + X_3(t-3\tau))\right]$$
$$+p_{13}K_3\left[2X_3 - \frac{1}{3}(X_1(t-3\tau) + X_2(t-3\tau))\right]$$
$$= \lambda_1 - \mu_1 + \frac{2}{3}\begin{bmatrix}-K_1 & p_{12}K_2 & p_{13}K_3\end{bmatrix}\begin{bmatrix}X_1\\X_2\\X_3\end{bmatrix}$$
$$+\frac{1}{3}\begin{bmatrix}0 & K_1 & K_1\end{bmatrix}\begin{bmatrix}X_1-\tau\\X_2-\tau\\X_3-\tau\end{bmatrix}$$
$$-\frac{1}{3}\begin{bmatrix}p_{12}K_2 + p_{13}K_3 & p_{13}K_3 & p_{12}K_2\end{bmatrix}\begin{bmatrix}X_1-3\tau\\X_2-3\tau\\X_3-3\tau\end{bmatrix}$$

Similarly, equations for subsystem 2 and 3 can be developed. After generalisation, the equations for all three subsystems can be written as:

$$\dot{X}_1 = \lambda_1 - \mu_1 + a_{1,0}X(t) + a_{1,1}X(t-\tau) + a_{1,3}X(t-3\tau) \quad (11)$$

$$\dot{X}_2 = \lambda_2 - \mu_2 + a_{2,0}X(t) + a_{2,1}X(t-\tau) + a_{2,3}X(t-3\tau) \quad (12)$$

$$\dot{X}_3 = \lambda_3 - \mu_3 + a_{3,0}X(t) + a_{3,1}X(t-\tau) + a_{3,3}X(t-3\tau) \quad (13)$$

Now the generalised equation for the system can be derived from (11), (12), and (13) as:

$$\dot{X} = \lambda - \mu + A_0 X(t) + A_1 X(t-\tau) + A_3 X(t-3\tau) \quad (14)$$

where $A_0 = \begin{bmatrix} a_{1,0} \\ a_{2,0} \\ a_{3,0} \end{bmatrix}, A_1 = \begin{bmatrix} a_{1,1} \\ a_{2,1} \\ a_{3,1} \end{bmatrix}, \text{ and } A_3 = \begin{bmatrix} a_{1,3} \\ a_{2,3} \\ a_{3,3} \end{bmatrix}.$

Equation (14) can again be simplified in the following form:

$$\dot{X} = \lambda - \mu + A(d)X(t) \quad (15)$$

where, $A(d)$ is a polynomial matrix in the time-delay operators $d = \{d_i\}$.

$$A(d) = A_0 + d_1 A_1 + d_3 A3$$
$$d_1 A_1 X(t) = A_1 X(t-\tau), \text{ and}$$
$$d_3 A_3 X(t) = A_1 X(t-3\tau).$$

C. SIMULINK Block Diagram and Simulation Results

As the whole system consists of three identical subsystems, therefore, block diagram of each subsystem is built first using Matlab SIMULINK. The block diagram of subsystem 1 is shown in Fig. 5 and the block diagram of the whole system is shown in Fig. 6.

The experimental procedures to determine different delay values were shown in [29], [30]. In this paper, the system is simulated by setting the initial condition as $x_1(0) = 0.85, x_2(0) = 0.63, \text{and } x_3(0) = 0.5; \lambda_1 = 3\mu_1, \lambda_2 = 0,$ $\lambda_3 = 0; \mu_1 = \mu_2 = \mu_3 = 1$. Processing gain for all the subsystems is thought as equal, i.e., $K_1 = K_2 = K_3 = K$ and all other conditions are same as in [5].

Fig.7 and 8 show the experimental responses as *queue length vs. time* with $K_1 = K_2 = K_3 = 1000$ and $K_1 = K_2 = K_3 = 2000$ respectively. On the other hand, Fig. 9 and 10 show the responses as *(queue length−local average queue length) vs. time* with $K_1 = K_2 = K_3 = 1000$ and $K_1 = K_2 = K_3 = 2000$ respectively.

It is observed from the simulation outcome that the oscillatory behaviour of system responses increased as the gain increased. At the same time we can see that, responses in Fig.8 and 10 with the higher gain oscillate more and die out slowly compared to responses in Fig.7 and 9 with less gain.

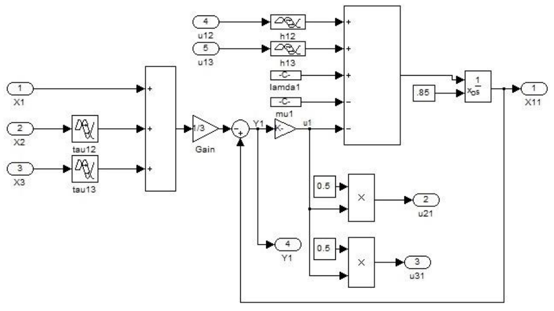

Fig. 5. Simulation Block Diagram of Subsystem 1

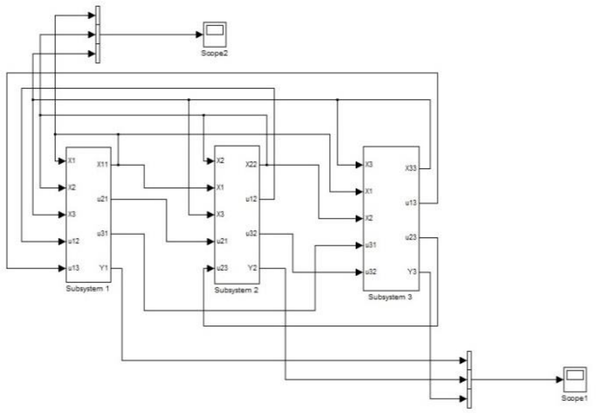

Fig. 6. Simulation Block Diagram of Linear Time-Delay System

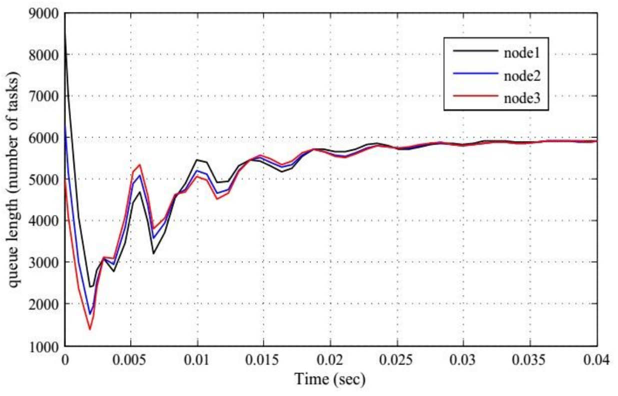

Fig. 7. Experimental responses as queue length vs. time with K1 = K2 = K3 = 1000

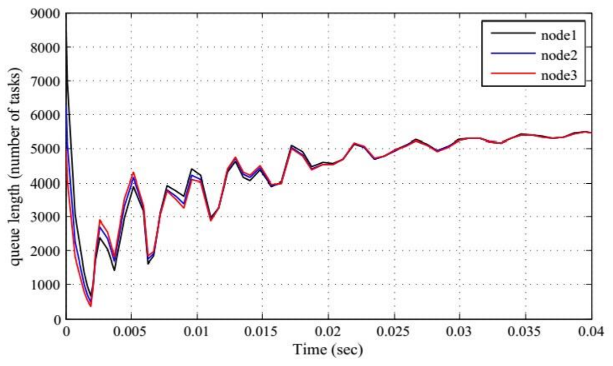

Fig. 8. Experimental responses as queue length vs. time with K1 = K2 = K3 = 2000

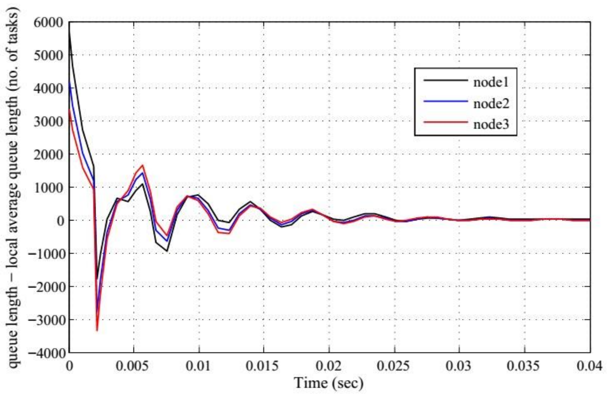

Fig. 9. Experimental responses as (queue length−local average queue length) vs. time with K1 = K2 = K3 = 1000

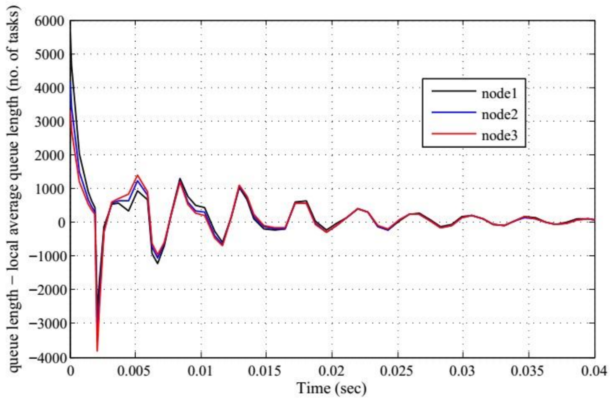

Fig. 10. Experimental responses as (queue length−local average queue length) vs. time with K1 = K2 = K3 = 2000

This unexpected behaviour is due to the delay experienced by the subsystems during communication. If delays are set to zero, then responses with higher gain will die out faster as expected. The oscillatory behaviour of the output responses depend on the linear control law (load balancing algorithm). The system experiences time delay whenever any transmission takes place among nodes. In the above system, transmission of task depends on the value of $u_i(t) = -K_i y_i(t)$ as decided by the linear control law. According to load balancing algorithm, if at the i-th node, $-K_i y_i(t) < 0$, i.e., $y_i(t) > 0$ then i-th node sends some of its tasks to other nodes. On the other hand, if $y_i(t) < 0$, i.e., $u_i(t) = -K_i y_i(t) > 0$ then the node will not send tasks to other nodes rather the node is instantly takes waiting time (tasks) from other nodes.

IV. Observers for Linear Time-delay Systems

In the previous section, a linear time-delay system is modelled and simulated to observe the effect of time delay on system performance as speed of load balancing operation and oscillatory behaviour of responses. In this section, observers of the linear time-delay system are designed considering the effect of time-delay in the system performance. The idea of an observer is that by merging calculated feedback signals with information of the modules of the control system, the behaviour of the system can be identified with higher accuracy as compared to utilising the feedback signal alone [16]. System performance can be improved by using an observer with skill and the reliability of the system can also be increased.

A. Mathematical Model of Observer

Let a time delay system be defined as follows:

$$\dot{X}(t) = A(d)X(t) + Bu,\ y(t) = C(d)X(t) \qquad (16)$$

where, $X(t) \in \mathbb{R}^n$ is the system state and $y(t) \in \mathbb{R}^m$ is the output of the system at time instant t. $A(d)$ and $C(d)$ are polynomial matrices in the time-delay operators $d = \{d_i\}$. For any i, operator d_i is defined by $d_i X(t) = X(t-\tau)$ with $\tau_i > 0$ which is the time-delay constant. When compared to the standard Luenberger observer, the following system might be considered as an observer for the system described by (16):

$$\dot{\hat{X}}(t) = A(d)\hat{X}(t) + Bu + L(d)(y(t) - C(d)\hat{X}(t)) \qquad (17)$$

where, $\hat{X}(t) \to X(t)$ as $t \to \infty$ for arbitrary initial values of $\hat{X}(t)$ and $L(d)$ is the observer gain.

Now we can derive the definition of our system from (15) as follows:

$$\dot{X}(t) = \lambda - \mu + A(d)X(t)\ and\ y(t) = CX(t))$$

Now by using observer design concept described in (17), the observer for the linear-time delay system considered in this paper can be defined as:

$$\dot{\hat{X}}(t) = A(d)\hat{X}(t) + \lambda - \mu + L(d)(y - C\hat{X}(t)) \qquad (18)$$

B. SIMULINK Block Diagram and Simulation Results for the System with Observer

According to the observer equation and using SIMULINK built in blocks an observer is designed for subsystem 1. The block diagram for the observer of subsystem 1 is shown in Fig. 11. Fig. 12 and 13 show the responses of the system as *queue length vs. time*, and Fig. 14 and 15 show the responses as *(queue length−local average queue length) vs. time* in the presence of observer with system and observer gain K$_1$ = K$_2$ = K$_3$ = 1000, L = 1000, and K$_1$ = K$_2$ = K$_3$ = 2000, L = 2000 respectively.

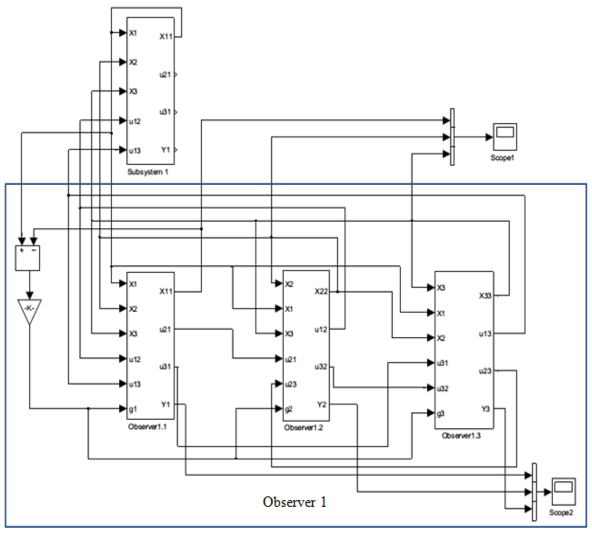

Fig. 11. Block Diagram of Observer 1 of Linear Time-Delay System

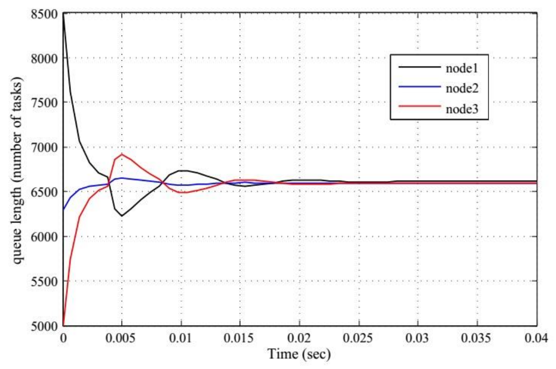

Fig. 12. System responses as queue length vs. time in the presence of observers with K1 = K2 = K3 = 1000 and L=1000

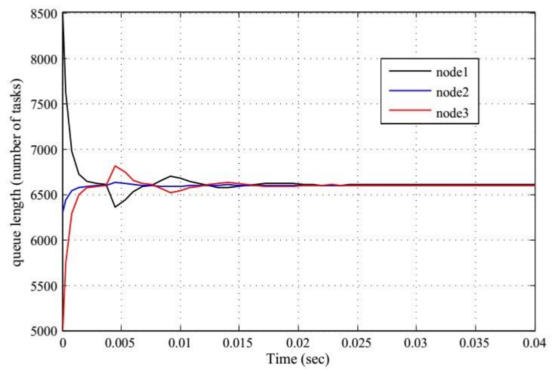

Fig. 13. System responses as queue length vs. time in the presence of observers with K1 = K2 = K3 = 2000 and L=2000

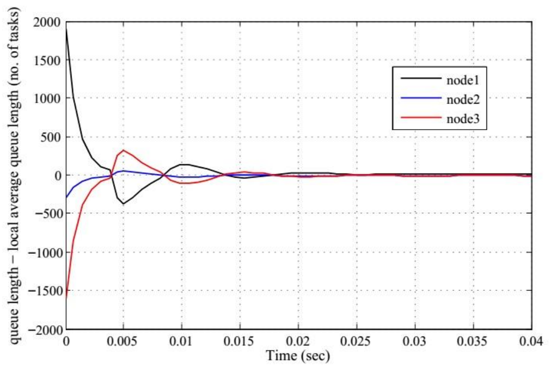

Fig. 14. System responses as (queue length−local average queue length) vs. time in the presence of observers with K1 = K2 = K3 = 1000 and L=1000

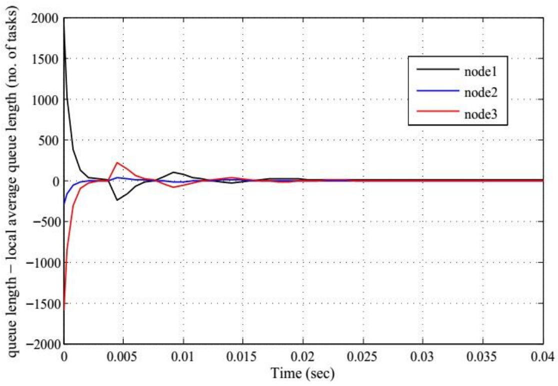

Fig. 15. System responses as (queue length−local average queue length) vs. time in the presence of observers with K1 = K2 = K3 = 2000 and L=2000

V. Discussion

One important issue in designing linear time-delay system is determining whether the system model is stable in the presence of oscillation in the system responses. It is well established that the presence of delays in various states of the system has an enormous impact on the stability of the system. Even if the system is stable, still system performance can be sluggish. For example, if the system is stable but oscillates too much while transferring tasks back and forth among subsystems rather than processing tasks, then system performance will degrade and the system will unnecessarily waste resources. Considering all possible issues with regard to stability and performance of the system, a linear time-delay system is modelled to observe the stability of the load balancing performance.

The system experiences delays during sending waiting time (queue size) and tasks from one subsystem to another. For simplicity, delays in sending queue size and tasks are considered as constant, but in real life usually delays are not constant. Initially, all the subsystems are fed with an equal quantity of load. Then subsystems experience creating and reducing waiting time. But, since the rate of reduction of the waiting time for all subsystems are we deliberately set as equal, i.e., $\mu_1 = \mu_2 = \mu_3 = 1$ and generation of waiting time as unequal, i.e., $\lambda_1 = 3\mu_1, \lambda_2 = 0, \lambda_3 = 0$ therefore loads become imbalanced within few moments. To balance loads among the nodes, each node shares load information with other nodes. As communications to share waiting time take place among nodes, all the nodes experience delays. As a result, subsystems receive queue size (expected waiting time) and tasks delayed by a certain amount of time.

The linear control law $u_i(t) = -K_i y_i(t)$ is used for load balancing, where $u_i(t)$ is the reduction rate of waiting time $x_i(t)$, and evaluated as per unit time. Here, the system gain K denotes the reduction rate of the waiting time of the system. According to the control law, within Δt time period between consecutive executions of the load balancing algorithm, a fraction of the queue is removed resulting into reduced waiting time. According to the linear control law, with higher system gains, the waiting time reduces quickly. In the simulation, it is observed that with higher gain the system responses oscillate more and die out slowly. This unexpected system behaviour is due to the delays experienced during communication. If we can compensate the effect of delays on system performance, then the system responses will die out fast as expected with higher gain. As delays are unavoidable, therefore the system gain is chosen in a way that the system responses oscillate moderately and system performance are mildly affected. To avoid the effect of time delay during the exchange of waiting time, observers are designed for all subsystems. Observer of each subsystem can estimate some important information, and thus enabling each subsystem to work independently. As a result, communication between nodes to exchange waiting time is no more necessary; therefore no delay is experienced by the system due to communication.

VI. Conclusion

In this paper, a linear time-delay system is modelled to analyse stability and performance of the system in performing load balancing operation and observers of all subsystems are designed based on the analysis. It is assumed that the system consists of three symmetric subsystems and all the delays experienced by the subsystems are constant. In principle, system performance is considered as less oscillation in the responses and high throughput that increases as the system gain increases. Stability is another issue which requires to be considered along with system performance. In the modelled system, to perform load balancing operation each subsystem has to communicate with other subsystems to get information about the amount of load on other subsystems. This result in the system experiencing delays due to mutual communications between subsystems and these delays limit the value of the system gain.

It is observed that, the system responses oscillate more with higher gain in comparison to lower the gain. More oscillation in system responses means the system waste resources by passing tasks back and forth among subsystems rather processing tasks which results in degraded system performance. This unexpected behaviour of the system is due to the delays experienced by the subsystems while communicating with each other. After analysing system responses, observer of the system is designed keeping the effect of time-delay on system performance in mind. As observers are designed considering all the issues that affect system performance, therefore expected system performance, i.e., the load is balanced quickly with higher gain, is obtained from the system with the observer.

In this experiment, all the subsystems of the system were considered as symmetric and all the delays experienced by the system as constant. But in real life, it is not necessary for all the subsystems of a system to be symmetric and delays to be constant. Delays depend on different physical properties such as the availability and bandwidth of network, execution time of programs etc. Therefore, in future we plan to model linear time-delay system and design observers of the system considering variable delays and asymmetry of the subsystems. A processor of a subsystem uses a considerable amount of time to perform load balancing other than processing tasks which is not considered in this work. In future, the time required by the processors of the subsystems to perform load balancing operation can be taken into account together with the time required to process tasks.

References

[1] C. T. Abdallah, N. Alluri, J. D. Birdwell, J. Chiasson, V. Chupryna, Z. Tang, and T. Wang, "Linear time delay model for studying load balancing instabilities in parallel computations," *Int. J. Syst. Sci.*, vol. 34, no. 10–11, pp. 563–573, 2003.

[2] D. A. Patterson and J. L. Hennessy, *Computer Organization & Design: The Hardware/Software Interface*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993.

[3] J. P. Richard, "Time-delay systems: an overview of some recent advances and open problems," *Automatica*, vol. 39, no. 10, pp. 1667–1694, 2003.

[4] J. K. Hale and S. V. Lunel, "Stability and control of feedback systems with time delays," *Int. J. Syst. Sci.*, vol. 34, no. 8–9, pp. 497–504, 2003.

[5] S. Kabir, A. S. M. A. Alam, and T. Azad, "Performance Evaluation of Parallel-processing Networked System with Linear Time Delay," in *13th Australasian Symposium on Parallel and Distributed Computing*, 2015.

[6] E. Fridman and U. Shaked, "A descriptor system approach to H_∞ control of linear time-delay systems," *IEEE Trans. Automat. Contr.*, vol. 47, no. 2, pp. 253–270, 2002.

[7] Y. Xia and Y. Jia, "Robust stability functionals of state delayed systems with polytopic type uncertainties via parameter dependent lyapunov functions," *Int. J. Control*, vol. 76, no. 16–17, pp. 1427–1434, 2002.

[8] Y. He, M. Wu, J. H. She, and G. P. Liu, "Parameter-dependent lyapunov functional for stability of time-delay systems with polytopic-type uncertainties," *IEEE Trans. Automat. Contr.*, vol. 49, no. 5, pp. 828–832, 2004.

[9] Y. He, Q. G. Wang, C. Lin, and M. Wu, "Delay-range-dependent stability for systems with time-varying delay," *Automatica*, vol. 43, no. 2, pp. 371–376, 2007.

[10] C. Lin, Q. G. Wang, and T. H. Lee, "A less conservative robust stability test for linear uncertain timedelay systems," *IEEE Trans. Automat. Contr.*, vol. 51, no. 1, pp. 87–91, 2006.

[11] V. L. Kharitonov, "Robust stability analysis of time delay systems: A survey," *Annu. Rev. Control*, vol. 23, pp. 185–196, 1999.

[12] X. M. Zhang, M. Wu, J. H. She, and Y. He, "Delay-dependent stabilization of linear systems with time-varying state and input delays," *Automatica*, vol. 41, no. 8, pp. 1405–1412, 2005.

[13] C. Hua, X. Guan, and P. Shi, "Robust backstepping control for a class of time delayed systems," *IEEE Trans. Automat. Contr.*, vol. 50, no. 6, pp. 894–899, 2005.

[14] H. Gao, T. Chen, and J. Lam, "A new delay system approach to network-based control," *Automatica*, vol. 44, no. 1, pp. 39–52, 2008.

[15] J. W. Ko, W. I. Lee, and P. Park, "Delayed system approach to the stability of networked control systems," in *Proceedings of the international multiconference of engineers and computer scientists*, 2011, pp. 772–774.

[16] G. Ellis and G. H. Ellis, *Observers in Control Systems: A Practical Guide*, Electronic. Academic Press, 2002.

[17] Z. Wang, D. Goodall, and K. Burnham, "On designing observers for time-delay systems with nonlinear disturbances," *Int. J. Control*, vol. 75, no. 11, pp. 803–811, 2002.

[18] X. Lu, H. Zhang, W. Wang, and K. L. Teo, "Kalman filtering for multiple time-delay systems," *Automatica*, vol. 41, no. 8, pp. 1455–1461, 2005.

[19] Z. Wang, F. Yang, D. W. Ho, and X. Liu, "Robust H_∞ filtering for stochastic time-delay systems with missing measurements," *IEEE Trans. Signal Process.*, vol. 54, no. 7, pp. 2579–2587, 2006.

[20] Z. Wang, B. Huang, and H. Unbehauen, "Robust H∞ observer design of linear time-delay systems with

parametric uncertainty," Syst. Control Lett., vol. 42, no. 4, pp. 303–312, 2001.

[21] H. Trinh and M. Aldeen, "A memoryless state observer for discrete time-delay systems," *IEEE Trans. Automat. Contr.*, vol. 42, no. 11, pp. 1572–1577, 1997.

[22] M. Hou, P. Zitek, and R. J. Patton, "An observer design for linear time-delay systems," *IEEE Trans. Automat. Contr.*, vol. 47, no. 1, pp. 121–125, 2002.

[23] Y. M. Fu, G. R. Duan, and S. M. Song, "Design of unknown input observer for linear time-delay systems," *Int. J. Control. Autom. Syst.*, vol. 2, no. 4, pp. 530–535, 2004.

[24] A. M. Perdon and M. Anderlucci, "Geometric design of observers for linear time-delay systems," in *6th IFAC Workshop on Time Delay Systems*, 2006, pp. 120–125.

[25] M. Hou, K. Busawon, and M. Saif, "Observer design based on triangular form generated by injective map," *IEEE Trans. Automat. Contr.*, vol. 45, no. 7, pp. 1350–1355, 2000.

[26] A. J. Koshkouei and K. J. Burnham, "Discontinuous observers for non-linear time-delay systems," *Int. J. Syst. Sci.*, vol. 40, pp. 383–392, 2009.

[27] A. Delgado, M. Hou, and C. Kambhampati, "Neural observer by coordinate transformation," *IEEE Proc. Control Theory Appl.*, vol. 152, no. 6, pp. 698–706, 2005.

[28] Z. Khan, R. Singh, J. Alam, and S. Saxena, "Classification of load balancing conditions for parallel and distributed systems," *Int. J. Comput. Sci. Issues*, vol. 8, no. 5, pp. 411–419, 2011.

[29] P. Dasgupta, "Performance Evaluation of Fast Ethernet, ATM and Myrinet Under PVM," University of Tennessee, Knoxville, 2001.

[30] P. Dasgupta, J. D. Birdwell, and T. Wang, "Timing and congestion studies under pvm," in *Tenth SIAM Conference on Parallel Processing for Scientific Computation*, 2001.

Authors' Profiles

Sohag Kabir is a postgraduate research student in Department of Computer Science, University of Hull, UK. He received his MSc degree in Embedded Systems from University of Hull, UK in 2012 and BSc degree in Computer Science and Engineering from Military Institute of Science and Technology (MIST), Dhaka, Bangladesh in 2010. His research interests include embedded systems, parallel computing, information theory, image processing, model-based safety assessment, and probabilistic risk and safety analysis.

A S M Ashraful Alam is a postgraduate research student in Department of Computer Science in University of Otago, New Zealand. He received an EMBA in Finance from University of Dhaka, Bangladesh in 2013 and BSc in Computer Science and Engineering from MIST, Dhaka, Bangladesh in 2011. Prior to beginning research in Otago, he worked in Machine Readable Passports and Visa project in Department of Immigration and Passports in Bangladesh. His research interests include tiny system security, information theory, systems integration, ICT innovation, complex system modelling, cross-disciplinary research on technical education.

Tanzima Azad is a Graduate student in Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology (BUET), Dhaka, Bangladesh. She received her BSc degree in Computer Science and Engineering from Military Institute of Science and Technology (MIST), Dhaka, Bangladesh in 2010. Her research areas include parallel processing, cluster computing, Crowdsourcing, Crowd DB, and Location Privacy.

How to cite this paper: Sohag Kabir, A S M Ashraful Alam, Tanzima Azad,"Evaluation of Load Balancing Performance of Parallel Processing Linear Time-Delay Systems", IJIEEB, vol.7, no.3, pp.1-10, 2015. DOI: 10.5815/ijieeb.2015.03.01